: N° d'ordre
: N° de série

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

---

UNIVERSITE ECHAHID HAMMA LAKHDAR - EL OUED
FACULTÉ DES SCIENCES EXACTES
Département D'Informatique

Mémoire de Fin D'étude
Présenté pour l'obtention du Diplôme de

MASTER ACADEMIQUE

Domaine : **Mathématique et Informatique**
Filière : **Informatique**
Spécialité : **Systèmes Distribués et Intelligence Artificielle**

Présenté par :

- **Choungara salah**
- **Zebdi mohammed**

Thème

Un nouveau schéma de chiffrement pour sécuriser les messages dans les réseaux de capteurs

Soutenue le 22-09- 2022 Devant le jury:

M.	Khelaifa Abdenacer	MAA	Président
M.	Yagoub Mohammed	MCB	Rapporteur
M.	Kertiou Ismail	MCB	Encadreur
M	Kara Mostefa		Co-encadreur

Année Universitaire : 2021/2022

Remerciements

Avant tout, je remercie sincèrement le Bon Dieu, le Tout-Puissant, qui nous guide vers le droit chemin, de m'avoir permis de suivre le chemin de la recherche et de m'avoir donné le courage et la patience nécessaires pour accomplir ce travail. Ma sincère gratitude et mes remerciements vont d'abord et avant tout à mon directeur de thèse, le professeur Kertiou Ismail, qui m'a permis de faire mon travail. Ce traité doit beaucoup à ses précieux conseils, sa rigueur, son calme et sa volonté. J'adresse également mes sincères remerciements aux membres du jury qui ont accepté d'évaluer mon travail, le président du jury, M. Khalifa Abdnasser, et les candidats, M. Yagoub Mohammed. Je voudrais avoir pitié de l'âme pure de mes parents, mes chers parents, pour avoir encadré mon potentiel depuis l'enfance,Je remercie ma femme et mes enfants ainsi que toute ma famille et mes amis, en particulier M. kara Mustafa, qui n'a ménagé aucun effort pour m'aider, et mes proches pour leur soutien inconditionnel et continu. Enfin, je remercie tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail, et tous ceux qui m'ont accompagné et soutenu. Merci à tous. Mes cadeaux spéciaux vont particulièrement à ma femme et à mes deux fils. Que Dieu les protège pour nous.

Résumé :

La synchronisation d'horloge et la sécurité des données sont essentielles pour l'utilisation de n'importe quel réseau dans un domaine particulier, notamment les réseaux de capteurs sans fil. Parmi les exemples et applications réelles et critiques il y a le problème de la synchronisation d'horloge dans les réseaux **R**CSF. Le sujet de notre mémoire est dans ce cadre, ce travail offre une solution au problème de la synchronisation, notamment la protection des horloges dans les réseaux de capteurs sans fil. Alors que de nombreuses solutions proposées dans la littérature nécessitent plus d'espace de stockage ou plus de temps d'exécution, et donc, plus de consommation d'énergie, l'objectif de ce manuscrit est de proposer une technique de codage légère, efficace et applicable dans des environnements à faibles capacités telles que les capteurs . Le protocole proposé consiste à une technique de chiffrement, une méthode de division de nœuds et une méthode de calcul d'horloge finale.

La technique de codage assure que les messages d'horloge échangés entre les nœuds sont protégés contre les manipulations. Les nœuds du réseau sont divisés en groupes sachant qu'il y ait des éléments communs entre chacun des deux groupes afin d'unifier l'horloge générale du réseau. La façon de calculer ou de sélectionner l'horloge au niveau de chaque nœud est d'utiliser simplement la médiane.

L'analyse prouve que notre technique offre une protection solide contre de nombreuses attaques bien connues. Les simulations sont faites par le langage Python, ce qui montre l'efficacité du protocole proposé pour résoudre le problème de synchronisation d'horloge dans les réseaux de capteurs.

les mots-clés

synchronisation d'horloge , réseaux capteur sans fil, cryptographie

Summary

Clock synchronization and data security are essential for the use of any network in a particular domain, especially wireless sensor networks. Among the real and critical examples and applications is the problem of clock synchronization in RCSF networks. The subject of our thesis is in this context, this work offers a solution to the problem of synchronization, in particular the protection of clocks in wireless sensor networks. While many solutions proposed in the literature require more storage space or more execution time, and therefore, more energy consumption, the objective of this manuscript is to propose a lightweight, efficient coding technique. and applicable in low capacity environments such as sensors. The proposed protocol consists of an encryption technique, a node division method and a final clock calculation method.

The encryption technique ensures that clock messages exchanged between nodes are protected against manipulation. The nodes of the network are divided into groups knowing that there are common elements between each of the two groups in order to unify the general clock of the network. The way to calculate or select the clock at each node is to simply use the median.

The analysis proves that our technique offers strong protection against many well-known attacks. The simulations are made by the Python language, which shows the effectiveness of the proposed protocol to solve the problem of clock synchronization in sensor networks.

Key-words

clock synchronization, wireless sensor networks, cryptography

ملخص

الطبيعة التعاونية لشبكات الحساسات الاسلكية تجعل الحاجة إلى وجود مفهوم مشترك للوقت في الشبكة شرطا أساسية لمعظم التطبيقات. هذا المفهوم يمكن تجسيده بتوظيف بروتكول التزامن الوقتي الذي يأخذ بعين الاعتبار مختلف قيود شبكات المستشعرات الاسلكية مثل : الطاقة المحدودة، مساحة الذاكرة المحدودة وعطل أجهزة الحساسات الاسلكية.

موضوع أطروحتنا في هذا السياق ، يقدم هذا العمل حلاً لمشكلة التزامن ، ولا سيما حماية الساعات في شبكات الاستشعار اللاسلكية. في حين أن العديد من الحلول المقترحة سابقا تتطلب مساحة تخزين أكبر و وقتًا أطول للتنفيذ ، وبالتالي ، المزيد من استهلاك الطاقة ، إن الهدف من هذه الاطروحة هو اقتراح تقنية تشفير خفيفة الوزن وفعالة وقابلة للتطبيق في البيئات منخفضة السعة مثل أجهزة الاستشعار. يتكون البروتوكول المقترح من تقنية تشفير وطريقة تقسيم العقدة وطريقة حساب الساعة النهائية.

تضمن تقنية التشفير حماية رسائل الساعة المتبادلة بين العقد من التلاعب. سنعتمد تقسيم الشبكة إلى مجموعات مع مشاركة عناصر بين المجموعات بمفهوم الوزن من أجل توحيد الساعة العامة للشبكة بشكل افضل. طريقة حساب أو تحديد الساعة في كل عقدة هي ببساطة استخدام حساب المتوسط القيم للعقد المتصلة في نفس المجموعة .

يثبت التحليل أن أسلوبنا يوفر حماية قوية ضد العديد من الهجمات المعروفة. يتم إجراء عمليات المحاكاة بواسطة لغة Python، مما يدل على فعالية البروتوكول المقترح لحل مشكلة مزامنة الساعة في شبكات الاستشعار.

الكلمات المفتاحية

مزامنة الساعة، شبكات الاستشعار اللاسلكية، التشفير

Table of Contents

List of Tables v

List of Figures vi

Acknowledgments vii

Abstract viii

1 Introduction 1

2 Généralités sur les RCSF 4

2.1 Introduction 4

2.2 Réseaux sans fil 4

2.3 Réseaux de capteurs sans-fil (RCSF) 5

2.4 Clustering 8

2.5 Primitives cryptographiques 10

2.6 Conclusion 11

3 La synchronisation d'horloge dans les WSNs 12

3.1 Introduction 12

3.2 Besoin de la synchronisation d'horloge dans les WSNs 12

3.3 Protocoles de Synchronisation d'horloge dans les WSNs 15

3.4 Evaluation qualitative 19
3.5 Techniques de chiffrement 20
3.6 Conclusion 21

4 Un nouveau schéma de chiffrement pour la synchronisation d'horloge dans les RCSF 23

4.1 Introduction 23
4.2 Description de la proposition 23
4.3 Performance de la technique proposée 28
4.4 Clustering 29
4.5 Conclusion 30

5 Un nouveau schéma de chiffrement pour la synchronisation d'horloge dans les RCSF 31

5.1 Introduction 31
5.2 Implémentation 31
5.3 Performances 36
5.4 Conclusion 38

6 Conclusion 40

Bibliography 42

List of Tables

List of Figures

2.1 Caractéristiques techniques des capteurs 6
2.2 Composants d'un capteur sans fil 7
2.3 Domaines d'application des RCSF 8
2.4 Division de système en groupes 9

3.1 Division de système en groupes 20

4.1 Valeurs régulières et valeurs aberrantes 24
4.2 Protocole Proposé 25
4.3 Fonctions principales 26
4.4 Algorithme de chiffrement 26
4.5 Algorithme de déchiffrement 27
4.6 Algorithme de synchronisation 28

5.1 Convergence de l'horloge système avec groupes de 20 nœuds 32
5.2 Convergence de l'horloge système avec groupes de 30 nœuds 33
5.3 Convergence de l'horloge système avec groupes de 30 nœuds 34
5.4 Trois groupes simultanément avec des nœuds communs 35
5.5 Convergence de l'horloge système avec 3 groupes (nbr com = 2) 36
5.6 Convergence de l'horloge système avec 3 groupes (nbr com = 5) 37
5.7 comparaison des techniques de chiffrement 39

Acknowledgments

Thanks to ...

Abstract

With the advancement of technology and the abundance of information, it has become impossible to do without cloud services that enable data backup and availability, which involves access anytime, and from anywhere. The collective use of cloud data has become increasingly important in the security domain, mainly in the encryption domain for exchanging or storing data. Nowadays, cloud computing offers a suitable platform for users to use data. Nevertheless, the cloud can be an untrusted part that may expose decrypted data to abuse. Cryptography helps solve many computer problems, including cloud entities' communication. In computer cryptography, researchers are constantly working to find solutions to various challenges in order to preserve data and systems. Homomorphic encryption is one aspect of autonomy that provides safe use of the cloud. This work offers powerful and effective solutions represented by homomorphic encryption techniques and key exchange protocols.

Besides that, other problems with cloud entities cannot be addressed by homomorphic encryption. Cloud computing suffers from many problems such as lack of transparency and trust as well as centralization of control. The blockchain has powerful features like decentralization that enable it to add several features to cloud computing including autonomy, security, trustworthiness, and transparency. Therefore, we propose several consensus algorithms that are the core of the secure blockchain.

Résumé

Avec les progrès de la technologie et l'abondance d'informations, il est devenu impossible de se passer des services cloud qui permettent la sauvegarde et la disponibilité des données, ce qui implique un accès à tout moment et de n'importe où. L'utilisation collective des données cloud est devenue de plus en plus importante dans le domaine de la sécurité, principalement dans le domaine du chiffrement pour échanger ou stocker des données. De nos jours, le cloud computing offre une plate-forme appropriée aux utilisateurs pour utiliser les données. Néanmoins, le cloud peut être une partie non fiable qui peut exposer les données décryptées à des abus. La cryptographie aide à résoudre de nombreux problèmes informatiques, y compris la communication des entités cloud. En cryptographie informatique, les chercheurs travaillent constamment à trouver des solutions à divers défis afin de préserver les données et les systèmes. Le chiffrement homomorphe est un aspect de l'autonomie qui permet une utilisation sûre du cloud. Ce travail propose des solutions puissantes et efficaces représentées par des techniques de chiffrement homomorphe et un protocole d'échange de clés.

En plus de cela, d'autres problèmes avec les entités cloud ne peuvent pas être résolus par le chiffrement homomorphe. Le cloud computing souffre de nombreux problèmes tels que le manque de transparence et de confiance ainsi que la centralisation du contrôle. La blockchain possède des fonctionnalités puissantes comme la décentralisation qui lui permettent d'ajouter plusieurs fonctionnalités au cloud computing, notamment l'autonomie, la sécurité, la fiabilité et la transparence. Par conséquent, nous proposons plusieurs algorithmes de consensus qui sont les cœurs de la blockchain sécurisée.

1 Introduction

La grande évolution dans les domaines de la micro-électronique, de la micromécanique, et des techniques de communication sans fil, ont permis de produire avec un coût raisonnable des petits dispositifs de détection et de communication. Ces dispositifs sont connus comme des noeuds capteurs. Ils ont la capacité de collecter et transmettre des données environnementales vers un point centralisé appelé la Station de Base (SB) ou le puits. L'ensemble des noeuds capteurs forme un réseau de Capteurs sans fil (RCSF). Ce type de réseau est largement utilisé, et fait l'objet de plusieurs recherches scientifiques. Il constitue une solution efficace dans une grande variété d'applications comme les applications militaires, environnementales, domotiques, industrielles, etc. Parmi les caractéristiques de RCSF, le déploiement de ses noeuds dans des zones ouvertes d'une manière aléatoire ou déterministe, le fonctionnement autonome de ses noeuds capteurs. L'échange des données entre des noeuds capteurs se fait sans utiliser une infrastructure réseau préexistante et fixe ou une administration centralisée. Chaque noeud de réseau communique directement avec l'autre noeud qui se trouve dans son protée radio. La communication avec les noeuds distants ou hors portée radio se fait par l'intermédiaire des autres noeuds (communication multi-sauts). Les architectures de communications dans RCSF sont généralement classées selon deux catégories : communication plate et communication basée sur les clusters. La communication basée sur les clusters est une manière efficace de réduire la consommation d'énergie totale du réseau sans fil. L'idée consiste à former des groupes (clusters) de noeuds capteurs et d'utiliser parfois les CHs (Cluster-Heads) élus comme routeurs. Chaque CH collecte les données à partir de tous les noeuds capteurs qui appartenant à leur cluster, agrège les données rassemblées et les transmet directement vers station de base (SB).

Problématique

Les nœuds capteurs sont généralement déployés dans des zones non surveillées, la plupart des applications des RCSF nécessitent un haut niveau de sécurité pour fournir les exigences de sécurité de base et rendre ces applications invulnérables aux différentes attaques, empêchant un intrus de perturber le bien fonctionnement du réseau en prenant le contrôle des noeuds de capteurs. En plus, il est connu que les RCSF sont faciles à attaquer en raison de la nature du médium qui permet relativement facilement intrus d'espionner, d'altérer ou d'injecter des données dans le réseau. Objectif Dans ce travail, nous nous intéressons aux problèmes de sécurité des communications dans les réseaux de capteurs et synchronisation d'heur dans groupe des capteurs en particulier aux communications basées sur les clusters. Pour cela, nous proposons une solution de sécuriser des communications dans RCSF, qui offrent une bonne protection en tenant compte des caractéristiques limitées des capteurs. L'objectif est donc, d'assurer les services de sécurité les plus importants. Ainsi la proposition peut être robuste face aux attaques comme la manipulation de données. Notre proposition est basée sur un algorithme de chiffrement/déchiffrement des données et un algorithme de synchronisation dé d'hologe dans un groupe de capteurs.

Organisation du mémoire

Ce mémoire est organisé de la manière suivante :

Chapitre I : introduction aux réseaux de capteurs sans fil qui sont un type spécial de réseaux ad hoc et au clustering dans RCSF et concepts généraux de cryptographie.

Chapitre II : dans ce chapitre, nous abordons la problématique de la synchronisation d'horloge dans les réseaux de capteurs et exigences de la synchronisation dans les WSNs et autres travaux de synchronisation d'horloge, en expliquant les différentes vulnérabilités d'un réseau de capteurs, les différents types d'attaques qui visent ce réseau ainsi que les exigences de sécurités existantes et proposées pour faire face à ces menaces. Ensuite, nous présentons les différentes primitives cryptographiques utilisées dans les RCSF.

Chapitre II : dans ce chapitre, nous allons expliquer le protocole développé pour synchroniser l'horloge et les techniques de chiffrement et de déchiffrement, avec une explication de la réponse aux attaques de toutes sortes utilisant l'algorithme proposé.

Chapitre IV: dans ce chapitre nous exposons l'implémentation et l'évaluation de notre

proposition. Le système d'exploitation utilisé. Il consiste une programmation entière en langage Paython. Conclusion générale : Elle résume notre proposition et nos perspectives de recherches.

2 Généralités sur les RCSF

2.1 Introduction

Les progrès technologiques réalisés dans les communications sans fil et dans la microélectronique ont conduit à la naissance d'un nouveau genre de réseaux Ad-Hoc appelé Réseau de Capteurs sans Fil (RCSF). Ce type de réseau consiste généralement en un grand nombre de nœuds (nœuds capteurs). Ces nœuds peuvent être déployés dans des endroits géographiques en vue de collecter et transmettre des données vers un point de collecte appelé Sink ou Station de Base (SB). Le déploiement des nœuds capteurs se fait par manière aléatoire (avion) ou déterministe (manuelle). [1] Généralement, les RCSF peuvent être utilisés dans différents domaines tels que le militaire, l'environnementale, la médicale et la surveillance et le contrôle industriel.

2.2 Réseaux sans fil

Nous passerons en revue la définition du réseau de capteurs sans fil et de ses composants.

2.2.1 Definitions

Un réseau sans fil est un réseau informatique ou numérisé qui connecte différents postes ou systèmes entre eux par ondes radio. Il peut être associé à un réseau de télécommunications pour réaliser des interconnexions entre nœuds. La norme la plus utilisée actuellement pour les

réseaux sans fil est la norme IEEE802.11, mieux connus sous le nom de Wi-Fi. Le rayonnement géographique des ondes est relativement limité étant donnée la faible puissance d'émission des solutions matérielles actuelles. Pour cette raison, les réseaux sans fil se sont avant tout développés comme réseaux internes, propres à un bâtiment, soit comme réseau d'entreprises, soit comme réseau domestique. Néanmoins, des projets de réalisation de réseaux à grande échelle ont vu le jour, notamment le WiMAX.

Les éléments constitutifs d'un RCSF

Un réseau de capteurs sans fil se constitue principalement de plusieurs nœuds capteurs un nœud Sink et un centre de traitement de données.

Nœuds : ce sont des capteurs, qui répondent aux exigences de l'application pour laquelle ils ont été conçus du point de vue de leurs architectures, et de leur dispersion géographique.

Sink ou puits : c'est un nœud important du réseau. Son rôle diffère de celui des autres nœuds du réseau car sa tâche est de collecter les données qui proviennent des autres nœuds du réseau c'est pour cela qu'il doit être toujours actif car la réception des données se fait de manière aléatoire, et pour ce fait son énergie doit être illimitée.

Centre de traitement des données : Toutes les données collectées par le Sinky seront envoyées, ce centre va regrouper toutes les informations et les triées afin de relever celles qui seront exploitables.

2.3 Réseaux de capteurs sans-fil (RCSF)

Revenir à la définition ainsi qu'aux caractéristiques des capteurs sans fil (1).

C'est un système qui sert à, sous forme de signal souvent électrique, un phénomène physique afin de le représenter. Les capteurs sont des petits appareils dotés d'une batterie, capables de communiquer entre eux et de détecter des évènements s'ils se trouvent à l'intérieur de leur rayon de perception. Un capteur est un petit appareil doté de mécanismes lui permettant de relever des informations sur son environnement. La nature de ces informations varie très largement selon l'utilisation qui est faite du capteur : ce dernier peut tout aussi bien faire des

Propriétés	TmoteSky	WiSMote	MICAZ	TelosB
Microcontrôleur	MSP430 F1611	TI MSP430F5437x	ATmega 128L	TI MSP430
Fréquence d'horloge	3.9 MHz	16 MHz	16 MHz	8 MHz
RAM (Ko)	10	16	4	10
ROM (Ko)	48	256	128	48
Radio	CC2420	CC2520	CC2420	CC2420
Batterie	2.1 - 3.6V	2.1 - 3.6V	2.7 - 3.3V	1.8 - 3.6 V

Figure 2.1: Caractéristiques techniques des capteurs

relevés de température, d'humidité ou d'intensité lumineuse. Un capteur possède également le matériel nécessaire pour effectuer des communications sans-fil par ondes radio.

Il existe plusieurs modèles commercialisés des capteurs sans fil, nous citons : TmoteSky et WiSMote, MICAZ, TINYNODE, TELOSB, Le Tableau 2.1 illustré les principales caractéristiques techniques de quelques capteurs (2; 3).

2.3.1 Définition d'un RCSF

Le réseau de capteurs sans fil (RCSF), "Wireless Sensor Network (WSN)" est considéré comme un type spécial de réseau ad hoc. Le RCSF est constitué d'un grand nombre de nœuds capteurs répartis sur une zone donnée de manière aléatoire (avion, missile) ou déterministe (manuel, robotique). Les nœuds de capteurs communiquent entre eux via des liaisons radio pour le partage d'informations et le traitement coopératif.

Chaque nœud communique directement avec les autres nœuds situés dans sa portée de transmission. La communication avec le nœud distant ou hors de portée de transmission se fait par l'intermédiaire d'autres nœuds qui acheminent les données vers la destination, ce processus est assuré par un protocole de routage (4).

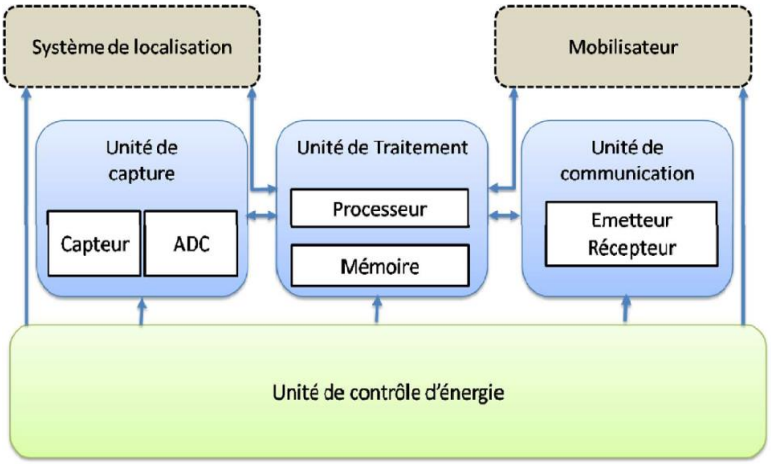


Figure 2.2: Composants d'un capteur sans fil

Un nœud de capteur se compose de quatre unités principales: (I) l'unité de détection ou captage, (II) l'unité de traitement, (III) l'unité de transmission et (IV) la batterie, comme illustré à la (Figure I.1), ces quatre unités sont brièvement définies comme montre Figure 2.2.

2.3.2 Domaines d'application

La miniaturisation, la facilité du déploiement, l'adaptation et la communication sans fil sont des facteurs ayant permis de généraliser l'utilisation des RCSF. De nos jours, ce type de réseaux est très utilisé dans divers domaines parmi lesquels nous pouvons citer: Domaine militaire : c'est une technique très utilisée pour des besoins militaires comme le suivi des troupes dans les champs de bataille. Domaine médical : les RCSF peuvent être utilisés pour assurer une surveillance permanente des organes vitaux de l'être humain (la détection de cancer, surveillance de glycémie, température corporelle, . . . etc.) grâce à des micro-capteurs qui peuvent être implantés sous la peau. Application industrielle : l'application de micro-capteurs dans le domaine industriel permet de contrôler l'environnement dans les entreprises manu facturières, la gestion d'inventaire, de la télésurveillance après-vente, . . . Etc. L'image (Figure 2.3) suivante représente les domaines d'exploitation et d'application du réseau de capteurs (5).

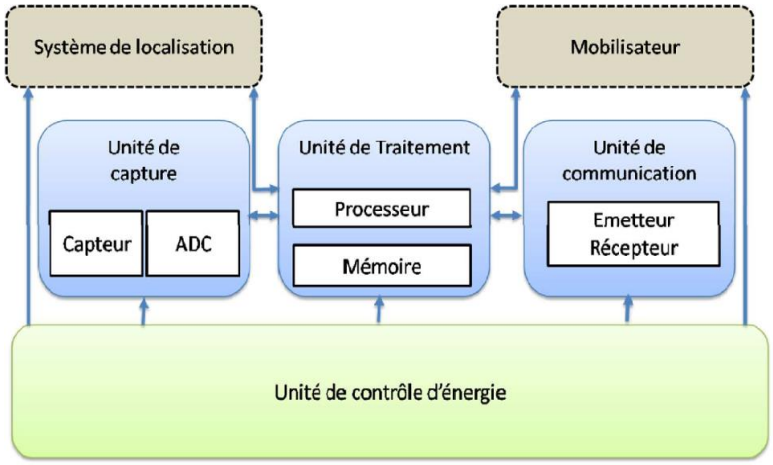


Figure 2.3: Domaines d'application des RCSF

2.4 Clustering

Nous aborderons la définition de la clustering, les étapes de clustering, et les méthodes existantes.

L'approche de Clustering consiste à partitionner le réseau en un certain nombre de clusters, et être plus homogène selon une métrique spécifique ou une combinaison de métriques, et former une topologie virtuelle. Les clusters sont en général identifiés par un nœud particulier appelé Cluster-Head. Ce dernier permet la coordination entre les membres de son cluster, d'agréger leurs données collectées à travers des liens radio et de les transmettre à la station de base (Figure 2.4).

Etapes du clustering

Afin de hiérarchiser ou de répartir les données faisant partie d'un réseau, le processus de clustering s'effectue en trois étapes principales : remière étape : la préparation des données, Deuxième étape : lasélection de l'algorithme de clustering et Troisième étape : la validation et interprétation des résultats. Plusieurs algorithmes de clustering existent et sont utilisés, selon le choix des critères considérés pour l'organisation des données.

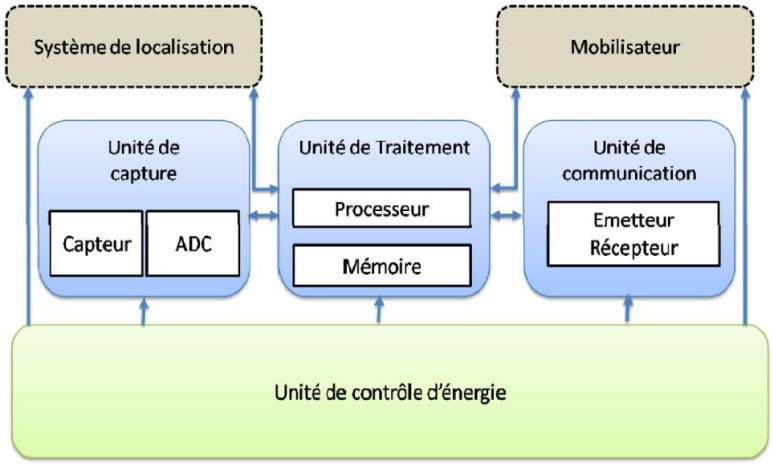


Figure 2.4: Division de système en groupes

2.4.1 Méthodes de Clustering de base

Il existe environ 30 méthodes de clustering dans la littérature, et leurs variantes sont incluses dans différentes familles. On distingue les grandes familles suivantes de méthodes de clustering (6).

Les méthodes hiérarchiques

La base du clustering hiérarchique est de construire une hiérarchie de clusters ou un arbre de cluster, également connu sous le nom de dendrogramme. Chaque nœud contient son sous-cluster et les nœuds frères divisent les objets contenus dans son nœud parent. Cette approche peut explorer les données à différents niveaux de granularité.

Les méthodes par partitionnement

Une façon typique de diviser la méthode consiste à utiliser une méthode itérative. Produire une classification par partitionnement revient à produire plusieurs classes non vides en déplaçant des éléments d'un cluster à un autre, à partir d'une partition initiale (le nombre de classes ou de clusters est généralement prédéfini) (7).

Les méthodes à base de densité

Le clustering basé sur la densité trouve des clusters de formes arbitraires dans des bases de données spatiales bruitées. La densité ici peut être définie comme le nombre de points dans un rayon spécifié. La technique de clustering basée sur la densité comprend principalement trois technologies : DBSCAN (clustering spatial basé sur la densité d'application avec bruit), OPTICS (points de commande pour identifier la structure de clustering), DENCLUE (DENsity CLUstering.

Les méthodes basées sur un modèle

Les méthodes de clustering basées sur des modèles tentent d'optimiser l'ajustement entre les données et un modèle mathématique. Ces méthodes reposent généralement sur l'hypothèse que les données sont générées à partir d'un mélange de distributions de probabilité sous-jacentes. Les méthodes de clustering basées sur des modèles suivent deux approches principales : les méthodes statistiques (probabilistes) et les méthodes de réseau de neurones.

2.5 Primitives cryptographiques

utilisées dans les RCSF Plusieurs mécanismes basés généralement sur la notion de cryptographie, sont mis en place afin de répondre à la question de la sécurité dans les RCSF. Le chiffrement et le déchiffrement des messages sont effectués par des algorithmes cryptographiques. Ces algorithmes sont souvent basés sur des problèmes mathématiques complexes et difficiles à résoudre. Ensuite, nous introduisons les différentes primitives cryptographiques utilisées dans les réseaux de capteurs sans fil.

2.5.1 La cryptographie

Le mot cryptographie est composé des mots grecs: crypto signifie caché et graphy qui signifie écrire. La cryptographie est définie comme étant une science permettant de protéger une communication et d'assurer que l'information contenue dans un message n'est révélée qu'au destinataire de ce message. On distingue deux familles de cryptographie : la cryptographie symétrique et la cryptographie asymétrique.

Cryptographie symétrique: La même clé est utilisée entre deux nœuds communicants pour chiffrer et déchiffrer les données à l'aide d'un algorithme de chiffrement symétrique. Cryptographie asymétrique: Dans la cryptographique asymétrique (ou la cryptographie à clé publique) Comme le montre l'image ci-dessous (Figure I.8), la clé de chiffrement et la clé de déchiffrement sont différentes. Une des clés appelée clé publique (qui est diffusée) utilisée généralement pour chiffrer le message. Tandis que l'autre clé appelée clé privée (gardée secrète), permet de déchiffrer le message crypté.

2.6 Conclusion

Dans ce chapitre, nous avons essayé de donner quelques généralités sur les réseaux de capteurs sans fil que nous avons considéré comme un cas particulier de réseaux Ad-hoc. Pour cela, nous avons décrit les principaux concepts liés aux réseaux de capteurs sans fil tels que : l'architecture, les domaines d'application, les topologies, et nous avons présenté un état de l'art qui détaille les attaques menaçant les réseaux en général et les RCSF en particulier. Pour faire face à ces attaques, nous avons présenté une synthèse bibliographique des systèmes de cryptographie et des mécanismes qui peuvent sécuriser les réseaux filaires. L'adaptation de ces derniers aux RCSF représente de grands challenges.

3 La synchronisation d'horloge dans les WSNs

3.1 Introduction

Dans ce chapitre, nous présenterons en premier lieu le besoin de la synchronisation d'horloge dans les WSNs. Ensuite, nous allons introduire les concepts de base de la synchronisation d'horloge (8; 9). Nous étudierons également quelques protocoles de synchronisation d'horloge dédiés spécialement aux WSNs, c'est-à-dire, qu'ils prennent en considération la miniaturisation des microcapteurs, et leurs contraintes de conception. Et enfin, nous terminerons par une conclusion.

3.2 Besoin de la synchronisation d'horloge dans les WSNs

La synchronisation d'horloge (10; 9) est un service qui vise à construire une notion du temps commune dans le réseau, ou encore à la construction d'un convertisseur du temps qui permet à chaque nœud d'estimer les valeurs d'horloges générées dans les différents nœuds du réseau (8). Ceci représente un chalenge important et primordial pour les applications collaboratives, ainsi que pour certains protocoles de base Par exemple, les micro-capteurs collaborent entre eux pour effectuer une tâche de détection complexe tel que la fusion de données qui sert à regrouper les données collectées par les différents nœuds en un seul résultat significatif. Ceci peut être utilisé pour tracer le chemin d'un véhicule, où les nœuds captent la position et le temps de détection du véhicule et les envoient à une station de base, qui à son tour combine toutes les informations

pour tracer le chemin complet. II est évident que si les nœuds ne sont pas synchronisés, le chemin sera inexact.

L'implémentation du mécanisme dit duty-cycling pour économiser la consommation d'énergie est un autre exemple; les micro-capteurs peuvent être mis en veille (éteindre le module de communication sans fil) à des moments appropriés, et de se réveiller lorsque c'est nécessaire. Les nœuds doivent alors passer en veille, et se réveillent à des heures coordonnées. Ce mécanisme de coordination doit assurer que l'interface radio d'un nœud ne soit pas éteinte lorsqu'il est censé recevoir certaines données. Se la exige une synchronisation précise entre les micro-capteurs.

3.2.1 Modèle d'Horloge

L'horloge est définie comme étant un ensemble de composants matériels et logiciels utilisés pour fournir un temps exact, stable et fiable. L'oscillateur est le composant principal dans l'horloge. Chaque oscillateur est défini par une fréquence exprimée en Hertz. Les horloges utilisées dans les composants électroniques modernes sont caractérisées par deux paramètres très importants:

La précision : c'est la différence entre les fréquences d'oscillations théoriques être elles. Cet écart est appelé erreur de fréquence, il est donné par le fabricant de l'oscillateur, et il est généralement de l'ordre de 10-5 pour les oscillateurs utilisés dans les composants électroniques des miro-capteurs modernes.

a stabilité : c'est la tendance de l'oscillateur à garder la même fréquence avec le temps, l'instabilité de l'oscillateur peut être à court terme et due aux facteurs d'environnement telle que les variations de la température, la pression la tension d'alimentation, etc., ou à long terme et due à des effets plus subtils tels que le vieillissement de l'oscillateur.

La conséquence directe de l'instabilité de l'oscillateur, est le skew. Ces derniers représentent la fréquence d'une horloge à un instant t Celle-ci peut changer par rapport à la fréquence initiale d'où son appellation skew ou déviation. La déviation continuelle des fréquences entraîne un décalage offset entre la valeur donnée par une horloge et le temps réel donné par une référence du temps telle que l'UTC. La différence entre les fréquences de deux nœuds du réseau est

appelée skew relative (relative skew). De même, la différence entre les valeurs d'horloges données par ces deux nœuds est appelée l'offset relatif (relative offset). Toutefois ; chaque nœud n1, soit peut estimer à n'importe quel instant t la valeur d'horloge C2 générée dans un autre nœud (n2) du réseau en utilisant sa propre horloge Cl.

3.2.2 Exigences de la Synchronisation d'horloge dans les WSNs

Beaucoup de travaux de recherche ont été consacrés pour la conception des nouveaux algorithmes de synchronisation spécialement adaptés aux réseaux de capteurs sans fil. Ces nouveaux algorithmes doivent prendre en considération plusieurs facteurs ou exigences (11; 12). Ces derniers peuvent être considérés comme des indicateurs pour évaluer les schémas désynchronisation dédiés aux WSNs. Ci-dessous, nous citons ces facteurs.

Efficacité Energétique Energy Efficiency : L'économie d'énergie est l'une des problématiques majeures dans les WSNs. En effet, la recharge des sources d'énergie est souvent trop coûteuse, et parfois impossible. Par conséquent, tous les protocoles conçus pour les WSNs, y compris les protocoles de synchronisation d'horloge, doivent minimiser au maximum la consommation énergétique.

Passage à l'Echelle Scalability : La plupart des applications des réseaux de capteurs sans fil nécessitent le déploiement d'un grand nombre de nœuds-capteurs. De ce fait, les algorithmes de synchronisation doivent s'adapter aux réseaux de forte densité ou à l'augmentation du nombre de nœuds.

Précision : Le besoin de la précision peut varier considérablement selon le but de l'application. Pour certaines applications, un simple ordre entre les événements et les messages peut suffire (un temps logique (13)), alors que pour d'autres, l'exigence de la précision désynchronisation peut être de l'ordre de quelques microsecondes, en raison de leur couplage étroit avec le monde physique.

Robustesse : Un réseau de capteurs est généralement laissé sans surveillance pendant des longues périodes de fonctionnement, éventuellement, dans des environnements hostiles. En cas dépanné ou de défaillance d'un petit nombre de nœuds, l'algorithme de synchronisation doit rester valide et fonctionnel pour le reste du réseau.

3.2.3 Sources d'Erreurs

Toutes, les méthodes de synchronisation d'horloge dans les WSNs comptent sur l'échange de messages entre les nœuds. La cause majeure d'erreur dans les protocoles désynchronisation provient de l'estimation non déterministe de la latence des messages échangés. Quand un nœud génère une estampille pour la synchronisation, le paquet portant cet estampillage fera face à une quantité variable de retard jusqu'à son arrivée à la destination prévue. Ce retard empêche le récepteur de comparer, avec exactitude, sa valeur d'horloge locale avec celle de l'expéditeur, comme montre là (Figure II.1) Ces sources d'erreurs peuvent être divisées en quatre catégories différentes (14).

Temps d'Envoi: C'est le temps nécessaire pour la construction d'un paquet de synchronisation. Il comprend la latence du système d'exploitation (par exemple, les changements du contexte), et le temps écoulé pour transférer le message à l'interface réseau.

Temps d'Accès : Il s'agit du temps d'attente pour accéder au canal de transmission (contention). Chaque paquet transmit est confronté à des retards dans la couche de contrôle d'accès au médium (MAC). Ce temps dépend du protocole MAC utilisé. De plus, les retransmissions dues aux problèmes d'interférence ou de collision font augmenter ce temps d'accès.

Temps de Propagation : C'est le temps nécessaire pour la transmission du message entre les interfaces réseau de l'émetteur et de récepteur, dans les réseaux sans fil où la portée est limitée à quelques dizaines de mètres, le temps de propagation est négligeable quand il s'agit du même domaine de diffusion (un seul saut).

Temps de Réception : Il s'agit du temps nécessaire à l'interface réseau du récepteur pour la réception du message et la notification du système d'exploitation de son arrivée. C'est typiquement le temps nécessaire pour générer un signal de réception du message. Il contient aussi, le temps écoulé dans le transfert de message à la couche application.

3.3 Protocoles de Synchronisation d'horloge dans les WSNs

Comme mentionné auparavant, les protocoles de synchronisation d'horloge dédiés aux réseaux filaires ne sont pas appropriés aux WSNs à cause des contraintes matérielles de ces derniers.

Nous présenterons dans ce qui suit, quelques protocoles de synchronisation d'horloge proposés spécialement pour les réseaux de capteurs sans fil.

Timing-sync Protocol for Sensor Networks (TPSN)

TPSN (15) est un protocole de synchronisation d'horloge destiné au WSNS Il est basé sur l'approche sender-receiver. Ce protocole vise à fournir une synchronisation d'horloge suivant le modèle Always-on, c'est-à-dire, chaque nœud maintient une horloge synchronisée à un nœud référence dans le réseau. De ce fait, TPSN permet d'établir une notion du temps commun pour tous les nœuds du réseau, ceci en se basant sur la correction des horloges. Le protocole TPSN suit l'approche maitresse/esclave et il est adapté aux réseaux multi hop de forte densité. Il appartient à l'ensemble des protocoles de synchronisation décentralisés, cars le nœud racine peut-être choisi par l'application d'un algorithme d'élection du chef leader élection algorithme TPSN utilise uniquement les liens bidirectionnels, afin de synchroniser les nœuds.

TPSN fonctionne en deux phases, la phase de découverte de niveau et la phase de synchronisation, dans la deuxième phase, chaque nœud de niveau i doit être capable de communiquer au moins avec un seul nœud de niveau i-1. À la fin de la phase de synchronisation, tous les nœuds du réseau vont être synchronisés avec le nœud racine.

Au déploiement du réseau, le nœud racine est autoattribué au niveau 0, et il initie la phase de découverte du niveau par la diffusion d'un paquet level discovery à ses voisins qui appartiennent à son domaine de diffusion. Le paquet envoyé contient l'identifiant, et le niveau de l'expéditeur. À la réception du paquet level discovery par les voisins du nœud racine, chacun doit s'autoattribuer à un niveau, ce dernier doit être égal au niveau reçu plus 1. Après avoir établi leur propre niveau, chaque récepteur diffuse un paquet level discovery qui contient son propre niveau et son identifiant pour assurer l'appartenance d'un nœud à un seul niveau, chaque nœud doit négliger le paquet level discovery après la première réception. À la fin de l'exécution de cette phase, une structure hiérarchique sera créée. Là, Peut-être le résultat après l'exécution de la phase de découverte du niveau.

Chaque nœud doit être une partie de la topologie hiérarchique, afin qu'il puisse être synchronisé avec le nœud racine. Cependant, il se peut qu'un nœud ne soit pas attribué à un niveau à cause de l'un des scénarios suivants. Premièrement, un nœud peut rejoindre le réseau après

l'exécution de la première phase. Ainsi, même si ce nœud est présent dès le déploiement du réseau, il peut rater la réception du paquet level discovery à cause des collisions. Pour remédier à ces deux situations. Chaque nœud déployé doit attendre un certain temps. S'il ne reçoit pas le paquet level discovery dans ce délai, il diffuse un message level request A la réception de ce message par ses voisins, ces derniers répondent en envoyant leurs propres niveaux. Enfin, le nœud s'attribue au niveau reçu.

Après l'exécution de la première phase, le nœud racine déclenche la phase de synchronisation en diffusant le message (time sync) A la réception de ce message parle nœuds du niveau (1) chaque nœud doit enregistrer sa valeur d'horloge dans une variable T1, attendre un certain temps, puis envoyer un paquet (synchronization pulse) vers le nœud racine. Ce paquet contient le niveau de l'émetteur, et sa valeur d'horloge lors de l'envoi de ce paquet Le temps d'attente entre la réception de (time sync) et renvoi de (synchronization pulse) permet d'éviter les collisions. À la réception du paquet (synchronization pulse), le nœud racine doit enregistrer sa valeur d'horloge dans une variable T2, attendre un certain temps, réenregistrer la valeur d'horloge dans T3, puis envoyé un accusé de réception au nœud de niveau (1) (l'émetteur du paquet (synchronization pulse)). Cet accusé contient en plus du niveau du nœud racine, les trois valeurs d'horloges : T1, T2 et T3. À la réception de l'accusé par les nœuds du niveau (1), chacun d'eux enregistre sa valeur d'horloge locale dans la variable T4. Nous allons présenter dans la section ultérieure, comment les nœuds de niveau (1) utilisent ces variables pour ajuster leur pour d'horloges. Après la synchronisation des nœuds du niveau (1) avec la racine, chaque nœud du niveau (i) doit se synchroniser aux nœuds du niveau (i-1). Les nœuds du niveau (i) doivent attendre un certain temps, avant d'envoyer le paquet (synchronization pulse) au nœud du niveau (i-1). Ce temps permet de s'assurer que les nœuds de niveau (i-l) le sont terminés l'exécution de la phase de synchronisation. à noter, qu'un nœud ne renvoie pas un accusé du paquet (synchronization pulse), sauf s'il est déjà synchronisé avec les nœuds du niveau (i-1). Ceci garantie que les nœuds des différents niveaux ajustent leurs horloges par rapport à celle du nœud racine. Le processus de synchronisation des nœuds des niveaux (i), avec ceux du niveau (i-1) est pareil à celui de synchronisation des nœuds du niveau (1) avec la racine.

Pour prendre en considération le problème du drift, le protocole TPSN peut utiliser l'un des estimateurs décrit en détail dans (16).

Flooding Time Synchronization Protocol (FTSP)

FTSP (17) est un protocole de synchronisation d'horloge fondé sur l'approche Sender/Receiver. Il organise une arborescence ad-hoc de synchronisation selon laquelle une racine est sélectionnée comme une source du temps. Ce protocole utilise la technique unidirectionnelle pour synchroniser les nœuds du réseau. La phase initiale de FTSP consiste à choisir le nœud racine, qui représente une source du temps pour tout le réseau. Le processus d'élection du nœud racine doit sélectionner le nœud qui possède l'identifiant le plus petit L'élection du nœud racine est suivi par une fenêtre de synchronisation, qui s'écoule pendant un intervalle du temps (i) le choix de cet intervalle est dicté par les exigences de précision de l'application développée.

La racine diffuse un paquet (synchronisation) qui sera reçu par tous les nœuds voisins. Le paquet doit contenir trois champs, timestamp, rootID et le champ seqNum. Le champ d'horodatage représente le temps de l'expéditeur lors de renvois du paquet, le rootID représente l'identificateur de la racine, et le seqNum est un numéro de séquence incrémenté uniquement par la racine au début de chaque tour de synchronisation. À la réception du paquet de synchronisation, chaque nœud doit calculer les paramètres de synchronisation, pour déterminer son décalage par rapport à l'expéditeur, corriger son horloge, et diffuser un message (synchronisation) Ainsi, le temps est effectivement inondé à travers le réseau.

Comme TSYNC (18) le mécanisme de base de fonctionnement il utilise les nœuds multi canaux tels que chaque nœud possède deux canaux pour contrôle et horloge. Tous les nœuds utilisent le même canal de contrôle mais le canal d'horloge est unique pour chaque nœud. Ce dernier pour gérer le trafic de l'ensemble de réseaux utilise deux protocoles le premier HRTS (Hierarchy Referencing Time Synchronization) qui est utilisée pour la synchronisation de l'ensemble de réseaux l'autre ITR (Individuel-based Time request) qui permet à chaque nœud de se synchroniser à la demande. Chaque nœud qui veut être synchronisé envoie un message de demande de RTI à son parent et cela est répété jusqu'à ce que le message de demande atteint la station de base et cette dernière renvoie son horloge à travers le canal d'horloge au noeud demandé.

Scalable Lightweight Time Synchronization Protocol for Wireless Sensor Network: SLTP (19) proposés s'appuie sur la méthode de regroupement (Clustring). Ce dernier fonctionne en deux phases, la première phase correspond à la configuration pour le réseau statique et dy-

namique dans lequel détermine le chef de chaque groupe de nœuds ; la deuxième phase permet au réseau de se synchroniser après la sélection des chefs des groupes

3.4 Evaluation qualitative

Nous parlons des critères les plus importants qui sont utilisés pour vérifier la crédibilité d'un protocole donné.

Efficacité énergétique C'est une exigence implicite dans la plupart des réseaux sans fil dans lesquelles cette obligation doit varier et être exécutée en fonction de la demande. Par exemple, dans le cas des réseaux de capteurs, cette exigence est stricte, ce qui oblige les nœuds d'à dormir le plus souvent possible et limitant fortement l'énergie disponible par la synchronisation et d'autres taches. La raison principale derrière cette contrainte de l'énergie est la petite taille des piles des capteurs ; ce qui limite la quantité de l'énergie stockée et produite.

Précision

La précision est la mesure de la façon dont le temps maintenu au sein du réseau est confirmé à l'heure normale. En d'autres termes, il s'agit d'une mesure de la précision de la synchronisation. Un protocole avec une grande précision garantit ainsi une haute précision dans le cas d'une précision absolue. Cela signifie que le temps synchronisé dans le réseau ne s'écarte pas beaucoup du repère extérieur (par exemple UTC). Dans le cas d'une précision chapitre III Proposition d'une approche d'économie d'énergie dans le réseau de capteurs relatifs signifie quand un ensemble des nœuds synchronisé est considéré l'écart maximum d'horloge d'un nœud dont l'ensemble est relativement petit.

Évolutivité (Scalability)

La portée d'un réseau est l'étendue géographique des nœuds qui sont synchronisés et l'exhaustivité de la couverture dans cette région. En général, la portée d'un réseau peut être éloignée en augmentant le nombre de nœuds dans le réseau.

Complexité globale

Elle est considérée comme une combinaison de la complexité algorithmique telle que la tolérance aux pannes, et les frais généraux de communication.

	Caractéristiques dépendantes de l'application				
Protocoles	**Précision**	**Consommation d'énergie**	**Complexité global**	**évolutivité**	**Tolérance aux pannes**
TPSN	Elevé	Moyenne	Faible	Bonne	Oui
DMTS	Elevé	Elevé	Elevé	Bonne	Oui
FTSP	Elevé	Faible	Faible	Bonne	Oui
LTS	Moyenne	Moyenne	Moyenne	Bonne	Oui
TDP	Elevé	Moyenne	Elevé	Bonne	Oui
TSRT	Elevé	Elevé	Moyenne	Bonne	Oui
SLTP	Elevé	Moyenne	Elevé	Bonne	Oui

Figure 3.1: Division de système en groupes

Tolérance aux pannes

La tolérance aux pannes joue un rôle important car un support sans fil est une source d'erreurs. Les manques de fiabilité de la remise des messages dans un milieu sans fil peuvent avoir des effets dévastateurs sur les protocoles de synchronisation car cette dernière nécessite des échanges de messages.

On retiendra du tableau (Tableau 3.1) précédent, qui consiste à comparer certains protocoles sur plusieurs critères liés à la sécurité (20; 21), à précision, consommation d'énergie (22; 23) et complexité globale, évolutivité tolérance aux pannes il nous apparaît que le second protocole est le meilleur en général.

3.5 Techniques de chiffrement

Pour montrer l'efficacité de notre technique, on va présenter brièvement quelques autres techniques de chiffrement et faire une comparaison en matière de temps de chiffrement et de déchiffrement. Dans (24), les auteurs ont présenté un schéma de chiffrement homomorphe asymétrique (25; 26) qui repose sur la conversion des nombres de base en une base donnée de

telle manière que le chiffrement entièrement homomorphe et avec niveau (LFHE) est effectué. Cette technique légère est facile à mettre en œuvre et applicable dans de nombreux domaines (27; 28), notamment ceux qui nécessitent rapidité et sécurité (29; 30).

Dans (31), une nouvelle variante du cryptosystème d'ADN est proposée pour sécuriser les données d'origine dans les nucléotides d'ADN, offrant un plus grand espace de stockage, des frais généraux réduits et des opérations dynamiques. L'importance de l'ADN est incorporée dans le nouveau système cryptographique d'ADN proposé, qui crypte les données transférées entre le propriétaire des données et l'utilisateur des données dans le cloud. Le cryptosystème El Gamal amélioré est le cryptosystème asymétrique proposé utilisé pour résoudre les problèmes de gestion des clés dans le cloud, en transférant en toute sécurité le fichier de clé entre le propriétaire des données et l'utilisateur des données. Dans (32), les auteurs ont décrit un schéma de chiffrement sur les entiers. Le principal attrait de ce schéma est sa simplicité conceptuelle. Cette simplicité se fait au détriment d'une taille de clé publique en $O(\lambda 10)$ qui est trop grande pour tout système pratique. Ils ont réduit la taille de la clé publique à $O(\lambda 7)$. En chiffrant avec une forme quadratique dans les éléments de clé publique, au lieu d'une forme linéaire. Le schéma est sémantiquement sécurisé, basé sur une variante plus forte du problème approximatif-GCD.

Dans (33), l'auteur a proposé un schéma de chiffrement symétrique basé sur un anneau polynomial, qui est fondamentalement quelque peu homomorphe et rendu entièrement homomorphe à l'aide d'une procédure de rafraîchissement.

3.6 Conclusion

Parmi les nombreuses difficultés dans la construction d'un réseau de capteurs, un enjeu essentiel est fourni : c'est la synchronisation d'horloge entre l'ensemble de réseaux et la limite des ressources. Dans ce chapitre, nous avons mis le point sur les protocoles les plus utilisés pour la synchronisation en essayant de les étudier en détail et après l'étude de différents algorithmes, nous avons basé notre travail sur une étude comparative concernant des points faibles et fort basés sûrs de différents facteurs y compris la précision, l'exactitude, le coût et la complexité. Cette étude guidera les chercheurs dans l'intégration des fonctionnalités de la solution de divers

protocoles comme elle permettra de créer un succès de synchronisation de temps pour leurs applications. chapitre III présente une proposition d'une approche de synchronisation d'horloge. D'une part, d'économie d'énergie dans le réseau de capteurs par un algorithme léger. D'autre part, améliorer la sécurité dans le réseau de capteurs avec une protection efficace contre les attaques.

4 Un nouveau schéma de chiffrement pour la synchronisation d'horloge dans les RCSF

4.1 Introduction

Dans ce chapitre, nous allons détailler notre proposition concernant la synchronisation d'horloge qui consiste à régler toutes les horloges des nœuds du réseau à une variation uniforme, car une telle synchronisation temporelle absolue ne peut être réalisée naturellement. Nous allons aussi décrire les méthodes de division l'ensemble de nœuds dans un réseau de capteurs. Et enfin, nous terminerons par une conclusion.

Dans le domaine de la synchronisation et de la sécurité, le gros problème auquel sont confrontés les chercheurs est celui des valeurs aberrantes comme le montre l'image ci-dessus (Figure 4.1), c'est l'occasion d'aboutir à des solutions proches de la réalité.

4.2 Description de la proposition

De point de vue de participation de nœuds dans le processus de synchronisation, les protocoles sont divisés en deux types principaux. Le premier type est basé sur un (des) nœud (s) de référence où l'horloge réseau global est synchronisée via ses messages envoyés aux autres nœuds. Ce type est vulnérable à de nombreuses attaques ; en outre, si le nœud de référence échoue, alors le système perdra la synchronisation. Le deuxième type est basé sur le consensus (34; 22), c'est-à-dire que tous les nœuds participent à l'opération de la synchronisation (35).

Dans notre travail, nous définissons un schéma pour chiffrer les messages de synchroni-

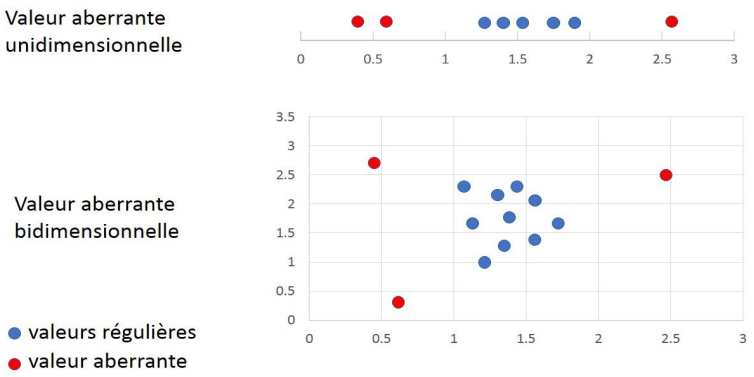


Figure 4.1: Valeurs régulières et valeurs aberrantes

sation et un mécanisme de consensus pour choisir une valeur d'horloge. Le schéma proposé est dû aux faibles capacités des capteurs, nous avons choisi une technique légère où l'on multiplie le message en clair par une clé, puis ajouter aléatoirement au résultat une petite valeur, cette valeur ajoutée s'appelle un indicateur; l'indicateur est une valeur parmi un ensemble de valeurs prédéfinies (indicateurs) par tous les nœuds. D'une part, l'indicateur permet de masquer le chiffrement du message afin qu'il ne soit pas vulnérable aux attaques de déchiffrement externes. D'autre part, il permet de détecter rapidement si le message a été chiffré par un nœud honnête ou par un attaquant.

Le problème des protocoles de synchronisation dans divers domaines qui utilisent des capteurs en général est divisé en deux parties principales en matière d'efficacité et de sécurité afin que nous puissions les évaluer dans un domaine spécifique. L'image ci-dessus (Figure 4.2) montre le protocole proposé et les types de messages envoyés et reçus par un nœud pour la synchronisation, le chiffrement, le déchiffrement et les messages anormaux.

On a proposé un schéma linéaire simple et facile à implémenter. Généralement, presque toutes les techniques de chiffrement se basent sur des opérations exponentielles pour atteindre un niveau élevé de confidentialité. Notre schéma de chiffrement est lainier et symétrique I.e.

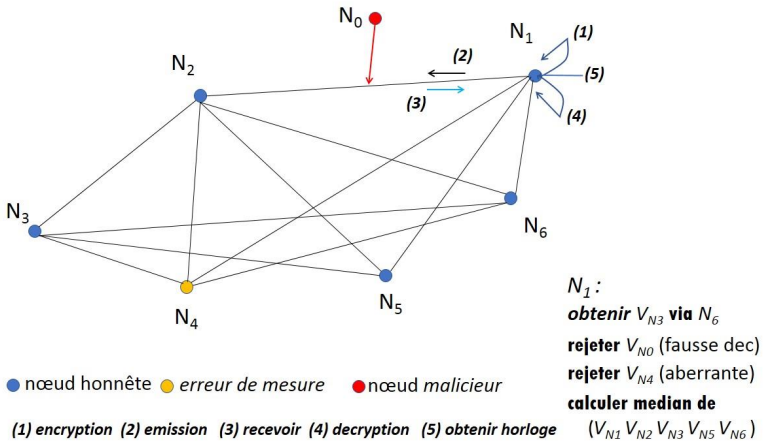


Figure 4.2: Protocole Proposé

que la clé de chiffrement est la même que la clé de déchiffrement. Nous considérons les primitives suivantes M, C, K, X, Enc(), Dec() et le tableau ci-dessous (Tableau III.1) montre leurs significations.

La technique proposée peut être définie dans le Tableau 4.3. Où : 'c' est le texte chiffré (ciphertext), 'm' est le message d'horloge à chiffrer (plaintext), 'k' est la clé de chiffrement, 'xi' est une valeur de vérification choisie aléatoirement pour chaque message parmi un ensemble de valeurs X afin de faire un bruit de chiffrement.

Algorithme de chiffrement

L'algorithme de chiffrement est représenté par les instructions décrites ci-dessous (Tableau 4.4) permets de chiffrer les messages de manière efficace et légère.

Algorithme de déchiffrement

L'algorithme de déchiffrement est représenté par les instructions décrites ci-dessous (Tableau 4.5) permets de déchiffrer les messages de manière efficace, sécurisée et légère.

Le rôle de la valeur secrète x_i est la vérification si le texte chiffré 'c' est une valeur réelle i.e. qu'elle provient d'un nœud honnête (capteur) ou s'il est une valeur manipulée. Nous avons $c \mod k = x_i$, donc si $c \mod k \neq \alpha$ où $\alpha \notin X$ ça implique que 'c' n'est pas une donnée

Fonction	Description
KeyGen	Donne une clé secrète.
Enc(m)	Le message **m** est chiffré à l'aide de la clé secrète **k** et l'indicateur $\mathbf{x_i}$ $\mathbf{c = m * k + x_i}$ avec $\mathbf{x_i}$ est aléatoirement dans **X** et $\mathbf{x_i < m}$
Dec(c)	Le texte chiffré est déchiffré à l'aide de la clé secrète; $\mathbf{m = c \div k}$

Figure 4.3: Fonctions principales

Algorithme III.1 de chiffrement
1. Début
2. Requérir : m, k, X
3. Assurer : c
4. Fonction Enc{}
5. x ← choisir au hasard de X
6. c ← m * k + x
7. Retourner c
8. FinFonction
9. Fin

Figure 4.4: Algorithme de chiffrement

Algorithme III.2: de déchiffrement

```
1.  Début
2.  Requérir : c, k, X
3.  Assurer : m
4.  Fonction Dec{}
5.      m ← c ÷ k
6.      x ← c – (m * k)
7.      Si  x ∈ X
8.          Retourner m
9.      Sinon
10.         Retourner erreur
11. FinFonction
12. Fin
```

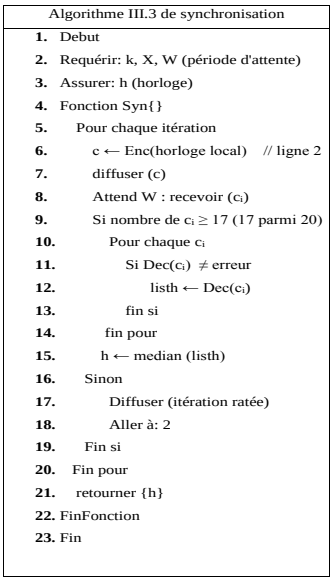
Figure 4.5: Algorithme de déchiffrement

réelle. Par exemple, si le message a été manipulé entre émetteur A et récepteur B, B va calculer $c \mod k$ ça va donner α, Le nœud B vérifiera cette valeur si elle est la même qui a été diffusé entre les nœuds ou non ; si oui, ‘c’ est une donnée réelle et B va l’accepter, sinon, ‘c’ est une fausse donnée ; I.e. il y a un élément externe qui a envoyé ce texte chiffré ou que ce message a été modifié en raison d’une erreur matérielle.

4.2.1 Synchronisation

Après avoir diffusé son horloge locale dans le réseau, chaque nœud doit attendre un temps prédéterminé ; puis le nœud déchiffre toutes les valeurs reçues des autres nœuds en utilisant k et X. Ici, nous utilisons la médiane pour sélectionner l’une des valeurs d’horloge décryptées ; en même temps, l’utilisation de la médiane exclut facilement les valeurs aberrantes sans nécessiter de nombreux calculs. Ainsi, l’utilisation de la médiane ne retarde pas la synchronisation d’horloge. De même, le temps d’attente ne retarde pas la synchronisation car ce temps est inclus dans la durée globale du processus de synchronisation.

L’algorithme (Algorithme 4.6) nous permet de synchroniser un certain nombre de capteurs en prenant un temps aléatoire au début et d’utiliser la synchronisation en calculant le temps



Algorithme III.3 de synchronisation

```
1.  Debut
2.  Requérir: k, X, W (période d'attente)
3.  Assurer: h (horloge)
4.  Fonction Syn{}
5.    Pour chaque itération
6.      c ← Enc(horloge local)   // ligne 2
7.      diffuser (c)
8.      Attend W : recevoir (cᵢ)
9.      Si nombre de cᵢ ≥ 17 (17 parmi 20)
10.       Pour chaque cᵢ
11.         Si Dec(cᵢ) ≠ erreur
12.             listh ← Dec(cᵢ)
13.         fin si
14.       fin pour
15.      h ← median (listh)
16.   Sinon
17.       Diffuser (itération ratée)
18.       Aller à: 2
19.   Fin si
20. Fin pour
21.  retourner {h}
22. FinFonction
23. Fin
```

Figure 4.6: Algorithme de synchronisation

moyen et en tenant compte des erreurs causées par les valeurs aberrantes de manière efficace, sûre et légère.

4.3 Performance de la technique proposée

Dans les réseaux de capteurs sans fil, la synchronisation de l'horloge ne peut pas être réalisée si le système est attaqué ou s'il y a des pannes de réseau. Le pirate peut empêcher la synchronisation elle-même, comme l'attaque par déni de service (Dos); où l'attaquant brouille les canaux de communication afin que les messages n'atteignent pas leurs destinations. L'attaque peut également entraîner un échec de convergence d'horloge, c'est-à-dire augmenter l'erreur de synchronisation dans le système, par exemple, une fausse attaque par injection de données. Il existe également des attaques qui peuvent être évitées par des techniques cryptographiques comme nous l'avons fait dans le protocole proposé.

comme l'attaque par manipulation de messages; notre protocole résiste à ce type d'attaque par le schéma de chiffrement proposé. La technique symétrique proposée contient deux types de clés secrètes, la première est 'k', la seconde est x_i. Dans l'attaque par fausse injection de

données où l'adversaire injecte de fausses données dans le réseau. Notre approche résiste à cette attaque. La première étape est le cryptage des messages par le schéma proposé ; les données erronées seront facilement détectées lors de leur décryptage car il est impossible pour un attaquant de prédire la valeur secrète x_i.

Dans une attaque par manipulation du message, si un adversaire a pu accéder à un message et qu'il l'a manipulé, la nouvelle valeur de ce message sera rejetée par le nœud. Il est facile de générer une valeur aléatoire $m \times k$ dont le déchiffrement $(m \times k/k)$ va donner une valeur 'm' quelconque qui peut être accepté. Mais lors de l'utilisation d'une deuxième clé secrète x_i ça va nous donner un échange sécurisé (36; 37); car dans l'opération de déchiffrement ($c \mod k = x_i$), le récepteur va trouver que $c \mod k = \alpha \neq x_i$.

Dos attaque : L'approche proposée est basée sur le consensus et non sur les nœuds de référence. Les protocoles de synchronisation d'horloge basée sur le consensus ne sont pas vulnérables aux attaques par déni de service.

Delay attaque : dans ce type d'attaque, l'adversaire veut retarder la transmission des messages d'horloge afin d'amplifier l'écart entre les valeurs d'horloge. Dans notre protocole, le nœud calculera l'horloge s'il a reçu au moins 15 valeurs dans la période déterminée ; par conséquent, la synchronisation sera réalisée avec un retard de 5 valeurs.

Schéma de cryptage : dans le cryptage de base, nous multiplions m par k. Ce schéma est vulnérable à plusieurs attaques, comme une attaque de texte claire connue où l'attaquant ayant le message d'origine et le texte chiffré correspondant (Enc(m), m), l'attaquant peut facilement obtenir la clé secrète k en le divisant sur (m). Dans le schéma proposé, $c = m \times k + x$; si l'attaquant possède m, cela ne donne rien à l'attaquant et il ne peut obtenir ni k ni x. Même si l'attaquant possède m et m', cela ne l'aidera en rien car les données chiffrées par la multiplication sont protégées par l'ajout d'un indicateur x, et donc ni des deux clés k ni l'indicateur x peuvent être extraits.

4.4 Clustering

Dans le travail présenté, on se préoccupera pas de diviser le nombre total de capteurs (section 4 chapitre I) , mais nous allons expérimenter le bug de plusieurs groupes différents en matière de

nombre d'individus pour comparer les résultats en matière de nombre affectant le temps de calcul de la convergence au sein de chaque groupe en calculant le temps moyen de chaque capteur et en choisissant la sécurité des messages et en essayant de faire passer le nombre d'individus de 20 à 30 puis 50, et on détermine quel nombre est plus approprié, ce qui donne le moins de temps de convergence, et ainsi nous avons économisé la quantité d'énergie consommée grâce au meilleur choix des critères de santé de base et grâce à la procédure d'assemblage avec équilibrage de charge approuvé, que cette dernière consiste à construire. Il consommera moins d'énergie lors du transfert de données au sein du groupe, et il peut conserver l'énergie pour le pilotage entre les groupes et réaliser une synchronisation proche de la réalité.

4.5 Conclusion

Dans ce chapitre, nous avons présenté un protocole de synchronisation d'horloge consistant en un schéma de chiffrement efficace et léger qui utilise des indicateurs pour détecter les messages manipulés par un nœud malveillant. Dans lequel également, nous nous sommes basés sur la sélection de médiane pour déterminer une valeur synchronisée d'horloge dans un groupe de nœuds. Nous avons présenté l'analyse de sécurité de l'approche proposée qui montre sa robustesse contre certaines des attaques les plus célèbres. Le prochain chapitre présentera l'implémentation et la performance de la technique proposée.

5 Un nouveau schéma de chiffrement pour la synchronisation d'horloge dans les RCSF

5.1 Introduction

Ce chapitre présente le détail de l'implémentation, à partir de l'environnement exploité passant par les différentes expérimentations qui montrent l'utilisation de différents nombres d'éléments dans chaque groupe par test. Puis nous exposerons les performances du protocole proposé par la présentation de sa scalabilité, sa complexité de calcul et sa tolérance aux pannes. À la fin du chapitre nous montrerons le temps de chiffrement et de déchiffrement de notre technique.

5.2 Implémentation

Dans notre implémentation, nous avons utilisé un PC de marque Dell, avec un processeur multi- core I3, horloges de fréquence de 2.40 GHZ et de mémoire vive (RAM) de 4 GO. Pour développer la technique, nous avons utilisé le langage Python. Python est un langage de programmation portable, dynamique, extensible, gratuit, de syntaxe très simple, code plus court que C ou Java, multi thread, orienté objet, évolutif.

Python est un langage de programmation de haut niveau interprété et orienté objet. Il est très sollicité par une large communauté de développeurs et de programmeurs. Python est un langage simple et facile à apprendre. Les bibliothèques de python sont disponibles pour la majorité des plateformes et peuvent être redistribuées gratuitement.

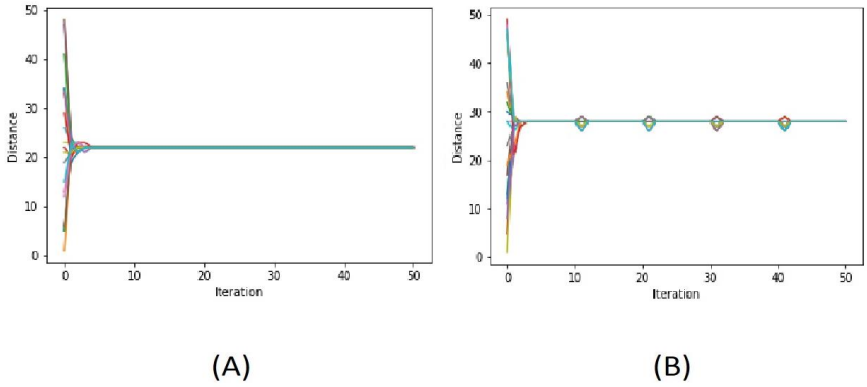


Figure 5.1: Convergence de l'horloge système avec groupes de 20 nœuds

5.2.1 Avec des groupes de 20 éléments

Nous avons créé un groupe contenant 20 nœuds, les valeurs initiales de l'horloge sont générées aléatoirement parmi l'intervalle (0.01,0.5), les nombres d'itérations sont égaux à 50 ; à chaque itération, on a supposé que chaque nœud ne recevra que 17 valeurs choisies aléatoirement parmi les 20 valeurs générées par ses voisins, dont une valeur a été manipulée par un attaquant, donc le nœud la rejettera à cause d'un faux déchiffrement, et que deux valeurs ont été perdu dans le réseau. Dans la (Figure 5.1) A, le test se fait sans générer d'erreurs d'offset pendant toutes les itérations, on remarque que le système a convergé a après 2 ou 3 itérations. Par contre, l'expérience illustrée par là (Figure 5.1) B se fait avec la présence d'erreurs d'offset toutes les 10 itérations ; la valeur d'erreur a été générée aléatoirement dans la plage (-0,002, +0,002). Nous remarquons que le système a convergé à nouveau après 2 ou 3 itérations toutes les 10 itérations, puis il reste stable jusqu'à ce que la prochaine erreur se produise.

5.2.2 Avec des groupes de 30 éléments

Nous avons créé un groupe contenant 30 nœuds, les valeurs initiales de l'horloge sont aussi générées aléatoirement parmi l'intervalle (0.01,0.5), les nombres d'itérations sont toujours

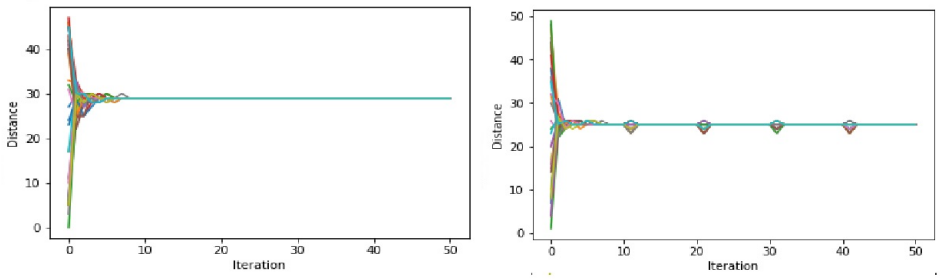


Figure 5.2: Convergence de l'horloge système avec groupes de 30 nœuds

égaux à 50 ; cette fois-ci , à chaque itération, on a supposé que chaque nœud ne recevra que 25 valeurs choisies aléatoirement parmi les 29 valeurs générées par ses voisins, dont deux valeurs ont été manipulée par un attaquant, donc le nœud les rejettera à cause d'un faux déchiffrement, et que trois valeurs ont été perdu dans le réseau. Dans la (Figure 5.2) A, le test se fait sans générer d'erreurs d'offset pendant toutes les itérations, on remarque que le système a convergé après 8 itérations. L'expérience illustrée par là (Figure 5.2) B se fait avec la présence d'erreurs d'offset toutes les 10 itérations ; la valeur d'erreur a été générée aléatoirement dans la plage (-0,002, +0,002). Nous remarquons que le système a convergé à nouveau après 2 ou 3 itérations toutes les 10 itérations, puis il reste stable jusqu'à ce que la prochaine erreur se produise.

5.2.3 Avec des groupes de 50 éléments

Dans le test illustré par là (Figure I.6), on a créé maintenant un groupe contenant 50 nœuds, les valeurs initiales de l'horloge sont générées toujours aléatoirement parmi l'intervalle (0.01,0.5), le nombre d'itérations est similaire aux deux tests précédents (50 itérations) ; à chaque itération, on a supposé que chaque nœud ne recevra que 40 valeurs, ces valeurs ont été choisies aléatoirement parmi les 49 valeurs générées par ses voisins, dont trois valeurs ont été manipulées

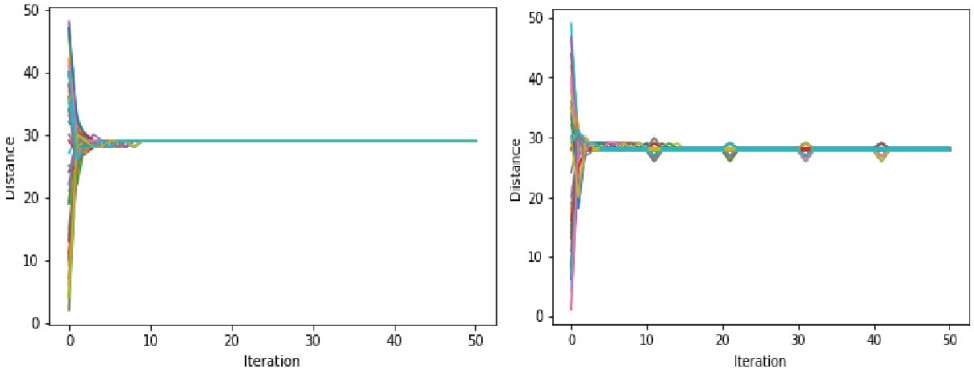


Figure 5.3: Convergence de l'horloge système avec groupes de 30 nœuds

par un attaquant, donc le nœud les rejettera à cause d'un faux déchiffrement, sept valeurs ont été perdu ou ont fait un retard d'échange. Dans la (Figure 5.3) A, le test se fait sans générer d'erreurs d'offset pendant toutes les itérations. Dans ce test, le système a convergé après 10 itérations. Dans l'expérience illustrée par là (Figure 5.3) B qui se fait avec la présence d'erreurs d'offset toutes les 10 itérations (la valeur d'erreur a été toujours générée aléatoirement parmi la plage (-0,002, +0,002)). Nous remarquons que le système a convergé à nouveau après 2 ou 3 itérations toutes les 10 itérations, puis il reste stable jusqu'à ce que la prochaine erreur se produise.

5.2.4 Discussion

D'après les trois expériences précédentes, la technique de sélection d'horloge se base sur le médian et en choisissant un nombre de valeurs aléatoires inférieures au nombre total des valeurs générées par le groupe, en tenant compte des valeurs manipulées par l'attaquant et des autres perdus dans le réseau. Nous remarquons qu'après l'augmentation du nombre d'éléments dans chaque groupe de nœuds 20, puis 30, puis 50, les expériences nous donnent des convergences de 5, 8, 10 itérations respectivement.

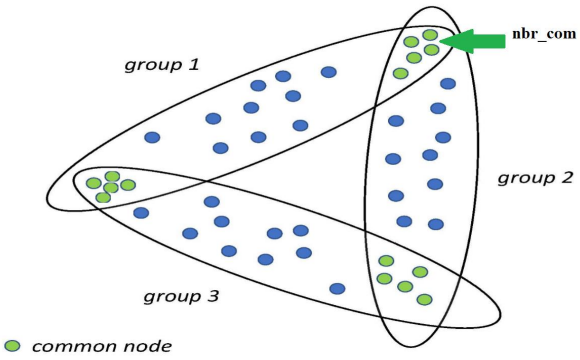


Figure 5.4: Trois groupes simultanément avec des nœuds communs

C'est un résultat logique car si le nombre d'éléments augmente, le système a besoin d'un plus grand nombre d'itérations pour que ces éléments atteignent la même valeur d'horloge. Dans le choix de la taille du groupe, on doit éviter les petites tailles car ça augmentera le nombre de groupes dans le réseau dont chaqu'un d'eux va converger vers sa propre horloge ; et les grandes tailles augmenteront le nombre d'envois de messages dans le système, et par conséquent, le taux de perte.

5.2.5 Unification de l'horloge globale du système

Dans ce point, nous avons fait des expérimentations sur un système de 60 capteurs divisés en trois groupes, 20 éléments chaqu'un. On a exécuté l'algorithme de synchronisation sur les trois groupes simultanément. Par ces tests, on a pris quelques nœuds communs dans chaque ensemble (Figure IV.5) dont le but d'unifier l'horloge globale du système, ou bien de minimiser l'écart de l'horloge dans le réseau.

La figure (Figure 5.5) montre le résultat de la première expérimentation qui prend 2 nœuds communs (nbr com = 2). Nous remarquons qu'après presque 10 itérations le premier groupe à converger vers l'horloge h = 27, le deuxième vers l'horloge h = 30, le troisième vers l'horloge

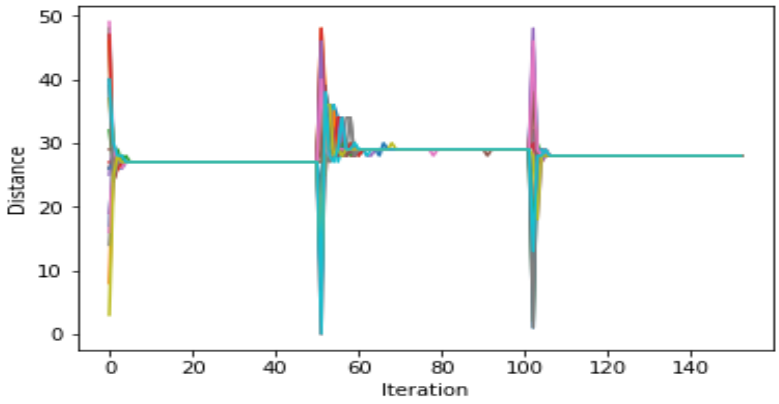


Figure 5.5: Convergence de l'horloge système avec 3 groupes (nbr com = 2)

h = 29. Sachant que chaque nœud dans le système a commencé par une horloge aléatoire dans l'intervalle (0, 50). L'exécution nous a donné des valeurs d'horloge très proche I.e. avec un écart (erreur) très petit dont e = 1/50.

La figure (Figure 5.6) montre le résultat de la deuxième expérimentation qui prend 5 nœuds communs. Nous remarquons qu'après presque 10 itérations tous les nœuds du réseau ont convergé vers l'horloge h = 24. Sachant que chaque nœud dans le système a commencé par une horloge aléatoire dans l'intervalle (0, 50). L'exécution nous a donné des valeurs d'horloge identiques I.e. avec un écart (erreur) égale à 0.

Ça montre l'efficacité du modèle proposé pour synchroniser l'horloge globale d'un système avec un niveau élevé de sécurité.

5.3 Performances

Dans cette partie nous présenterons les performances du protocole proposé en matière de scalabilité, complexité de calcul et tolérance aux pannes.

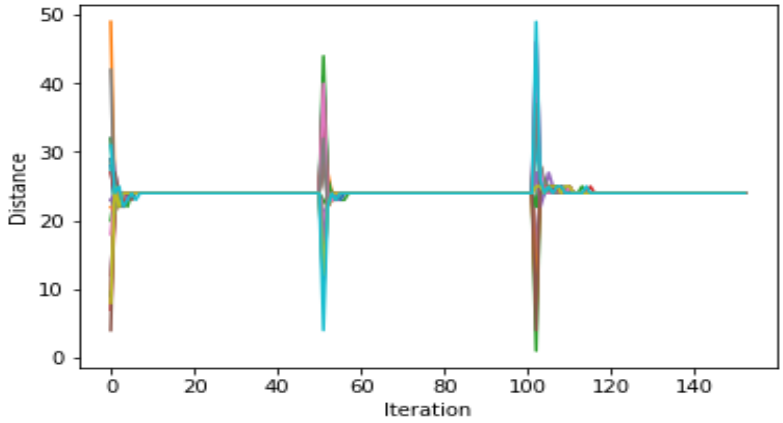


Figure 5.6: Convergence de l'horloge système avec 3 groupes (nbr com = 5)

5.3.1 Scalabilité

Scalabilité indique la possibilité d'augmenter le nombre de nœuds dans le réseau. Dans notre protocole, nous avons divisé le réseau en groupes, chaque groupe contient 20 nœuds, les nœuds d'un même groupe partagent la même clé de chiffrement et le même ensemble d'indicateurs. De cette façon, on évite l'augmentation des opérations arithmétiques et de la taille de stockage (clé et indicateurs) au cas où le nombre de nœuds dans le réseau soit augmenté ; par conséquent, le protocole proposé ne sera pas affecté par le nombre de nœuds qui composent le système.

Le problème est que chaque groupe convergera vers sa propre valeur médiane, et donc le système ne convergera pas vers une horloge commune, ce qui signifie que l'erreur de synchronisation sera agrandie. Pour résoudre ce problème, nous proposons de placer au moins un nœud commun (CNd) entre tous les deux groupes. C'est-à-dire que ce CNd possède la clé de chiffrement de chaque groupe (respectivement, les indicateurs de chaque groupe). Le CNd participera au processus de synchronisation d'horloge du deuxième groupe par l'horloge calculée dans le premier groupe. Bien sûr, si le nombre de nœuds communs entre chaque paire de groupes augmente, la précision de l'horloge système augmentera aussi. Si le CNd échoue, l'erreur de synchronisation sera importante.

5.3.2 Complexité de calcul

Dans les deux algorithmes: Enc et Eec, nous avons un nombre limité d'opérations élémentaires quelle que soit la taille de l'entrée m; par conséquent, les algorithmes de chiffrement et de déchiffrement contiennent une complexité égale à O(1). Dans l'approche proposée, nous avons limité le nombre de voisins par 20 dans chaque groupe de nœuds (un nœud + 19 voisins), puis le calcul de la médiane se fait dans un ensemble fini quel que soit le nombre total de nœuds du réseau. La complexité totale de l'algorithme de synchronisation dans chaque groupe est constante et égale à O(1). Chaque nœud doit exécuter Enc et Dec (avec O(1) comme complexité), donc si nous n'avons n nœuds alors la complexité de l'ensemble du processus est donnée par O(n).

5.3.3 Tolérance aux pannes

Lorsque le système continue à fonctionner, de manière éventuellement moins fiable, en cas de panne partielle ; c'est-à-dire qu'un ou plusieurs de ses éléments ne fonctionne plus correctement. Dans le protocole proposé, chaque nœud attendra pendant une durée précise ; s'il a reçu au moins 15 valeurs, il calcule l'horloge. Par conséquent, la panne (ou l'absence) de 5 nœuds n'affecteront pas le processus de synchronisation.

5.3.4 Chiffrement et déchiffrement

Pour la mise en œuvre de la technique proposée, nous avons utilisé un PC doté d'un processeur Intel(R) Core(TM) i3-3110M CPU 2.40 GHz, et 4 Go de RAM. La taille d'un message chiffré est de 16 bits. Le (Tableau 5.7) présente une comparaison des résultats de processus de chiffrement et de déchiffrement.

5.4 Conclusion

Après avoir présenté la description de notre protocole dans le chapitre précédent, ce chapitre a montré l'implémentation des deux algorithmes qui consistent le processus de consensus pour la synchronisation et la technique de chiffrement. On a aussi discuté les expérimentations faites

Technique	Enc (ms)	Dec (ms)
proposition	0.007	0.007
[41]	0.07	11.95
[42]	11.91	17.67
[43]	47	15
[44]	50	10
[45]	255	493

Figure 5.7: comparaison des techniques de chiffrement

avec une démonstration de performance de ce protocole. Finalement, pour bien mettre en valeur l'efficacité de notre schéma, on a fait une comparaison avec quelques autres cryptosystèmes.

6 Conclusion

La synchronisation d'horloge sécurisée dans les réseaux de capteurs est un élément important pour différentes applications aux réseaux de capteurs. Récemment, ce problème est largement étudié dans les réseaux de capteurs sans fil. En outre, les contraintes des réseaux de capteurs sont des facteurs essentiels pour construire un protocole de synchronisation d'horloge sécurisée. Parmi les solutions proposées, il y a des approches qui ont un coût de communication élevé à cause du nombre de messages échangés ou une grande consommation d'énergie à cause des protocoles de chiffrement complexe utilisés. À travers notre étude de différentes approches de synchronisation d'horloge, nous avons vu qu'il existe plusieurs façons de synchroniser l'horloge dans les réseaux de capteurs. Ces méthodes ont leurs inconvénients et leurs avantages. Dans l'étude présentée, nous nous sommes concentrés sur la réalisation des meilleures performances par rapport aux autres techniques.

Notre mémoire est constitué de quatre chapitres. Dans le premier chapitre, nous avons donné des généralités sur les réseaux de capteurs sans fil en décrivant les principaux concepts liés aux eux tels que : l'architecture, les domaines d'application, les topologies, et en présentant un état de l'art. Dans le deuxième chapitre, nous avons mis le point sur les protocoles les plus utilisés pour la synchronisation d'horloge et on les a étudiés de façon détaillée. Aussi, nous avons fait une étude comparative concernant les points faibles et fort basés sûrs de différents facteurs y compris la précision, l'exactitude, le coût et la complexité.

Pour garantir une approche plus solide, plus rapide et plus rentable. Nous avons défini dans le troisième chapitre un modèle de synchronisation d'horloge en utilisant un algorithme qui

fonctionne sur les capteurs individuels et calcule le temps moyen entre eux de façon sécurisée. Le schéma de chiffrement utilise une liste d'indicateurs pour détecter les messages manipulés par un nœud malveillant et en se basant sur la sélection médiane pour déterminer une valeur synchronisée d'horloge dans un groupe de nœuds. Ensuite, nous avons examiné la robustesse de la méthode proposée pour un ensemble bien connu d'attaques tel que l'attaque par fausse injection de données, attaque par manipulation du message, Dos attaque et Delay attaqué.

Dans le dernier chapitre nousavons présentés l'implémentation du modèle proposé en exposant deux algorithmes, le premier est le processus de consensus pour la synchronisation et le deuxième est la technique de chiffrement. On a aussi effectué des expérimentations afin de démontrer la performance de notre modèle tel que la scalabilité, la complexité de calcul et la tolérance aux pannes. Le modèle proposé a montré une grande efficacité dans la synchronisation d'horloge et une grande rapidité dans l'exécution, ce qui le rend adapté à une utilisation dans des environnements à faible efficacité telle que les réseaux de capteurs.

Nous avons expérimenté la mise à l'échelle de notre modèle en utilisant plusieurs nœuds communs aux deux groupes ; et nous avons introduit la notion de poids, qui consiste à attacher un poids à chaque nœud commun pour que sa montre ait un grand impact dans chaque groupe et soit un grand succès, car la montre est synchronisée par notre expérience jusqu'à 60 capteurs simultanément et nous espérons donner les mêmes résultats en grand nombre en l'essayant avec des nombres se chiffrant par milliers.

Bibliography

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[2] C. Duran-Faundez, "Transmission d'images sur les réseaux de capteurs sans fil sous la contrainte de l'énergie," Ph.D. dissertation, Université Henri Poincaré-Nancy I, 2009.

[3] W. Znaidi, "Quelques propositions de solutions pour la sécurité des réseaux de capteurs sans fil," Ph.D. dissertation, Lyon, INSA, 2010.

[4] Q. Zhao and L. Tong, "Distributed opportunistic transmission for wireless sensor networks," in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3. IEEE, 2004, pp. iii–833.

[5] I. Khemapech, I. Duncan, and A. Miller, "A survey of wireless sensor networks technology," in *6th Annual Postgraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting*, vol. 13. Citeseer, 2005.

[6] A. Manjeshwar and D. P. Agrawal, "Apteen: A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks," in *Parallel and distributed processing symposium, international*, vol. 3. IEEE Computer Society, 2002, pp. 0195b–0195b.

[7] M. Hadjila, H. Guyennet, and M. Feham, "Energy-efficient in wireless sensor networks using fuzzy c-means clustering approach," *International Journal of Sensors and Sensor Networks*, vol. 1, no. 2, pp. 21–26, 2013.

[8] M. Kara, A. Laouid, A. Bounceur, and M. Hammoudeh, "Secure clock synchronization protocol in wireless sensor networks," 2023.

[9] M. Kara, "A lightweight clock synchronization technique for wireless sensor networks: A randomization-based secure approach," 2023.

[10] A. Habib, A. Laouid, and M. Kara, "Secure consensus clock synchronization in wireless sensor networks," in *2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP)*. IEEE, 2021, pp. 1–6.

[11] M. Kara, A. Bounceur, M. Hammoudeh, N. Hamza, and A. Laouid, "A lightweight leader election algorithm for iot: Cloud storage use case," 2023.

[12] M. Kara, A. Laouid, A. Bounceur, and O. Aldabbas, "Arabic opinion mining using machine learning techniques: Algerian dialect as a case of study," 2023.

[13] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," in *Concurrency: the Works of Leslie Lamport*, 2019, pp. 179–196.

[14] B. Sundararaman, U. Buy, and A. D. Kshemkalyani, "Clock synchronization for wireless sensor networks: a survey," *Ad hoc networks*, vol. 3, no. 3, pp. 281–323, 2005.

[15] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, 2003, pp. 138–149.

[16] K.-L. Noh, Q. M. Chaudhari, E. Serpedin, and B. W. Suter, "Novel clock phase offset and skew estimation using two-way timing message exchanges for wireless sensor networks," *IEEE transactions on communications*, vol. 55, no. 4, pp. 766–777, 2007.

[17] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi, "The flooding time synchronization protocol," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, 2004, pp. 39–49.

[18] H. Dai and R. Han, "Tsync: a lightweight bidirectional time synchronization service for wireless sensor networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 8, no. 1, pp. 125–139, 2004.

[19] S. N. Gelyan, A. N. Eghbali, L. Roustapoor, S. Abadi, and M. Dehghan, "Sltp: scalable lightweight time synchronization protocol for wireless sensor network," *Lecture Notes in Computer Science*, vol. 4864, p. 536, 2007.

[20] M. Kara, "Safe control of autonomous cloud entities in distributed systems," 2023.

[21] S. S. Guia, A. Laouid, M. Kara, and M. Hammoudeh, "Tuberculosis detection using chest x-ray image classification by deep learning," 2023.

[22] M. Kara, A. Laouid, M. Hammoudeh, M. Alshaikh, and A. Bounceur, "Proof of chance: A lightweight consensus algorithm for the internet of things," *IEEE Transactions on Industrial Informatics*, 2022.

[23] M. Kara, A. Laouid, and M. Hammoudeh, "An efficient multi-signature scheme for blockchain," *Cryptology ePrint Archive*, 2023.

[24] M. Kara, A. Laouid, R. Euler, M. A. Yagoub, A. Bounceur, M. Hammoudeh, and S. Medileh, "A homomorphic digit fragmentation encryption scheme based on the polynomial reconstruction problem," in *The 4th International Conference on Future Networks and Distributed Systems (ICFNDS)*, 2020, pp. 1–6.

[25] M. Kara, A. Laouid, M. A. Yagoub, R. Euler, S. Medileh, M. Hammoudeh, A. Eleyan, and A. Bounceur, "A fully homomorphic encryption based on magic number fragmentation and el-gamal encryption: Smart healthcare use case," *Expert Systems*, vol. 39, no. 5, p. e12767, 2022.

[26] K. Chait, A. Laouid, L. Laouamer, and M. Kara, "A multi-key based lightweight additive homomorphic encryption scheme," in *2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP)*. IEEE, 2021, pp. 1–6.

[27] M. E. Kahla, M. Beggas, A. Laouid, M. Kara, and M. AlShaikh, "Asymmetric image encryption based on twin message fusion," in *2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP)*. IEEE, 2021, pp. 1–5.

[28] M. Kara, A. Laouid, A. Bounceur, M. Hammoudeh, M. Alshaikh, and R. Kebache, "Semi-decentralized model for drone collaboration on secure measurement of positions," in *The 5th International Conference on Future Networks & Distributed Systems*, 2021, pp. 64–69.

[29] M. Kara, "The secure management of autonomous cloud entities in a distributed system," Ph.D. dissertation, University Of Eloued, 2022.

[30] M. Kara, A. Laouid, A. d. Omer, M. Hammoudeh, and B. Ahcene, "One digit checksum for data integrity verification of cloud-executed homomorphic encryption operations," *Cryptology ePrint Archive*, 2023.

[31] M. Thangavel and P. Varalakshmi, "Enhanced dna and elgamal cryptosystem for secure data storage and retrieval in cloud," *Cluster Computing*, vol. 21, pp. 1411–1437, 2018.

[32] J.-S. Coron, A. Mandal, D. Naccache, and M. Tibouchi, "Fully homomorphic encryption over the integers with shorter public keys," in *Advances in Cryptology–CRYPTO 2011: 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings 31*. Springer, 2011, pp. 487–504.

[33] S. Dasgupta and S. Pal, "Design of a polynomial ring based symmetric homomorphic encryption scheme," *Perspectives in Science*, vol. 8, pp. 692–695, 2016.

[34] M. Kara, A. Laouid, M. AlShaikh, M. Hammoudeh, A. Bounceur, R. Euler, A. Amamra, and B. Laouid, "A compute and wait in pow (cw-pow) consensus algorithm for preserving energy consumption," *Applied Sciences*, vol. 11, no. 15, p. 6750, 2021.

[35] M. Kara, A. Laouid, A. Bounceur, F. Lalem, M. AlShaikh, R. Kebache, and Z. Sayah, "A novel delegated proof of work consensus protocol," in *2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP)*. IEEE, 2021, pp. 1–7.

[36] M. Kara, A. Laouid, M. AlShaikh, A. Bounceur, and M. Hammoudeh, "Secure key exchange against man-in-the-middle attack: Modified diffie-hellman protocol," *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika*, vol. 7, no. 3, pp. 380–387, 2021.

[37] M. KARA, A. LAOUID, A. BOUNCEUR, M. HAMMOUDEH, and M. ALSHAIKH, "Perfect confidentiality through unconditionally secure homomorphic encryption using otp with a single pre-shared key," *Journal of Information Science and Engineering*, vol. 39, no. 1, pp. 183–195, 2023.