

An Ultra Hierarchical Clustering-Based Secure Aggregation Protocol for Wireless Sensor Networks

Sébastien Faye, *Jean Frédéric Myoupo
*Laboratoire MIS, UFR Sciences, Université de Picardie Jules
33 rue Saint Leu 80039 Amiens, France*
{sebastien.faye@etud.u-picardie.fr, jean-frederic.myoupo@u-picardie.fr}

Abstract

In wireless sensor networks (WSNs), security and economy of total energy are two important and necessary aspects to consider. On the one hand, security helps to ensure that such a network is not subject to attacks that involve reading, modification or destruction of information. On the other hand, aggregation of data allows sensors to work together to combine messages to transmit, and in this way to obtain lower transmission costs, which increases efficiency and saves energy. In this paper, we superpose two types of clustered architectures of WSNs to yield ultra-hierarchical structures. A secure aggregation protocol for sensor networks is derived from this hierarchical structure. Our protocol is designed to prevent the majority of internal and external attacks of small and medium-sized proportions and to allow – unlike most previous protocols on the subject – an aggregation fully realized by the sensors, with a low use of the base station.

Keywords: *Wireless sensor networks, Clustering, Data aggregation, Security.*

1. Introduction

Wireless sensor networks (WSNs) are from the family of mobile ad hoc networks (MANET) but have additional features and constraints: typically they consist of a wide range of sensors and have a limited energy capacity. Each sensor is powered from a battery, non-rechargeable and non-replaceable ([2]), and has a low capacity in terms of memory, calculation (CPU), and transmission range. Each sensor is able to harvest a set of data in a certain environment, and is able to transmit the data in a multi-hop way to reach a base station (BS), which may act as an instructor for the network and may communicate the collected data to outside of the WSN. In such networks, security is a crucial point that needs to be studied and discussed. In fact, WSNs have many constraints, including first the communication medium, which is wireless: today it is very easy to read, intercept and even modify data that are transmitted wirelessly, and it is easy to undercut an entire network. Let us add to this the application of context sensors, which are usually deployed in hostile environments. Thus, there is a need for secure protocols to be used to ensure authentication, confidentiality of trade, data integrity and network availability. In the literature, several solutions of security protocols have been proposed, namely, TinySec [14] and μ TESLA [17]. These protocols provide authentication for the packets sent from a BS to all nodes (broadcast). Ultimately, good security should be able to counter both external and internal attacks.

1.1. Previous work

Among the main protocols dealing with the aggregation [1, 9, 15, 19], the protocol in [19] is not based on a cluster structure. This protocol has two main phases. The first phase is aggregation itself, which is conducted by at least network sensor called an aggregator, which forwards the aggregated results to the BS. The second phase is verification, which is necessary to protect against malicious aggregators. In this phase the BS checks the veracity of each sensor by initiating an interactive phase with the leaf nodes of the network. One solution that can be effective on small networks, but that can easily get into weaknesses when solving the scalability problem of dense networks, is that the base station carries too many checks, and node aggregators have too much work to do. Another protocol that

does not use the cluster structure is the protocol of Hu et al. [13]. This protocol is based on a binary tree to perform the aggregation. The aggregation is done from level to level, until arriving at the final data at the BS. This protocol uses an ingenious concept of hierarchy, but it only allows for the compromise of one aggregator, a solution that is too inefficient. Some aggregation protocols [6, 10, 18] aim to deal with *stealth attacks*, where the adversary's goal is to influence the aggregation result without being detected. Consistent with such a goal, these protocols can only raise an alarm when the aggregation result is corrupted. Recently the authors in [7, 12, 22] proposed protocols that detect malicious in order to diminish their attacking capability. Sen in [20] presents an energy-efficient aggregation algorithm for WSNs that is secure and robust against malicious insider attack by any compromised or faulty node in the network. All these papers assumed that some sensors can be trusted and they focus on detecting malicious sensors.

Another aggregation protocol using the notion of the cluster is attributed to Zhu et al. [24]. This protocol protects the network against various internal and external attacks, and allows the aggregation of data in a cluster. In particular, it provides an authentication service between pairs of non-neighboring nodes, which allows earlier detection of illegitimate packets: in a given cluster, each node collects a set of information. The cluster-head is responsible for the aggregation of the information and forwards the results – based on established keys – from node to node until the base station is reached. And the BS has to check the validity of this cluster aggregated data. This protocol provides very interesting results in terms of safety, but nevertheless, it has some weaknesses, notably its heavy complexity (A very large number of keys: the number of shared keys between nodes to ensure security is a consequence (A very large number of keys)).

The protocol SAPC in [3] is based on the clustered architecture protocol of Sun et al. [21]. The aggregation process is carried out in each cluster by all of its members, using a voting procedure, as in [13]. Next, each cluster head sends the aggregated data of its cluster to the BS. The BS aggregates all data it receives and checks its veracity. Some problems with other protocols mentioned here are resolved as well. A serious drawback of SAPC is that the distributed resources of the WSN are less used. It is disadvantageous not to use the distributed resources of the WSN to perform the aggregation mechanism. The WSN spreads the use of energy, so as not to overload the base station, and ultimately wins lifetime and efficiency. Moreover, the authors claim that sending all data to the BS for aggregation is the only way to avoid wrong aggregated data since cluster heads cannot be trusted. Moreover in the case of a huge number of cluster heads, SAPC turns out to be a data collection protocol with a light aggregation process.

1.2. Our contribution

In this paper, SAPC is significantly improved. To reach this goal we superpose two clustered architectures. First, the clustered WSN of Sun et al. [21] is used as the cluster of level 1. A variant of the method for defining virtual architectures in [23] is developed here to produce clusters of level 2 and higher. We show that the aggregation can be achieved in each cluster of level 1 and validated by a voting mechanism until the highest level that has only one node. This last node sends the final aggregated data to the BS. The BS does not have to carry out any aggregation process. Our protocol is secure, i.e., capable of subverting a majority of external and internal attacks that might occur. A major advantage provided by our approach is that it allows data aggregation to be fully realized by the sensors, and the base station is used for its most important value, which is less demanding than in some related work where the base station is considered all-powerful, just as with the SAP. Our solution therefore utilizes greater participation of the distributed resources in the WSN to increase its effectiveness in certain applications. Here, cluster heads are not trusted as in SAPC; however, we show that there is a solution that avoids the use of the BS as an aggregator. Here only the BS can be trusted contrary to the common assumption in the previous work [H. Chan, A. Perrig, and D. Song.], [K. Frikken and J. Dougherty.], [S. Nath, H. Yu, and H. Chan.], [P. Haghani, P. Papadimitratos, M. Poturalski, K. Aberer,, G. Taban and V. Gligor , B. B. Chen, H. Yu.], [Sen].

The rest of this paper is organized as follows: Section 2 gives a detailed description of our secure protocol. Analysis of the protocol's security is performed in Section 3. Simulation results are presented in Section 4, and conclusions are given in Section 5.

2. Secure Aggregation Protocol

Notations. To clarify the text, we use the notations below. Some notations are not specified here but are placed directly in the document:

BS \leftarrow base station.

CH \leftarrow cluster head.

D \leftarrow a message from a sensor, from the BS or an aggregated result.

CH^N \leftarrow cluster head of level N.

W^N{v} \leftarrow all nodes in the cluster of level N of node v.

ID_u \leftarrow identifier of sensor u.

W^N{v} \leftarrow all nodes in the cluster of level N of node v without v.

W^N \leftarrow all nodes of level N

a,b \leftarrow a is concatenated to b.

*W^N \leftarrow the set of all sensors of level N which are not cluster heads

K{n/u} \leftarrow a unique directional key chain of size n +1 generated by the sensor u.

K_{u,v} \leftarrow a secret key shared by u and v

K_{bs,u} \leftarrow a secret key shared by the BS and a sensor u.

K1 \leftarrow a secret key generated by the BS. The corresponding encryption function is denoted K1()

K2 \leftarrow a secret key generated by the BS. The corresponding encryption function is denoted K2()

MACK (M) \leftarrow an authentication message of 8 bytes generated over M using key K.

H \leftarrow a unique directional hash (μ TESLA).

2.1. Assumptions

We assume that:

1. The option of adding and deleting nodes from the WSN (for fault tolerance) is allowed but is rare.

2. We assume that the BS is a trusted entity (i.e., it cannot be compromised). It has a stronger capacity than the other nodes, and its transmission range can cover the entire network.

3. Once the nodes are deployed, they self-organize into clusters following the training that we describe below. Our first clustering level is the protocol in [21], which is based on disjoint cliques formed by the graph representing the WSN. The latter is secured in case of failure; for example, if a cluster head (CH) fails, simply redo an election for a new CH.

4. We assume that each node shares a secret key with the BS, which is loaded before deployment. Also, each node is loaded with the necessary hardware cryptographic key established with its neighbors (using mechanisms as proposed in [4-8]).

5. Each sensor knows its neighbors and has a unique identifier (ID), uniquely identifiable through defined keys. A message sent by a normal node can be received correctly by all of its neighbors (one hop). All messages exchanged between two nodes are authenticated using the shared key between nodes. Each node can generate a public key based on a signature (a realistic assumption from the literature [11], [16]). Messages (broadcast) are authenticated through a combination of signatures and the protocol μ TESLA, in which the signature is used for non-repudiation of data. μ TESLA is used for efficient broadcast authentication. The clocks of nodes are synchronized, as required by μ TESLA. Keys distributed by the various nodes and by the BS are authenticated (using μ TESLA or a certificate to ensure the authenticity of a key received).

2.2. Objectives in Terms of Security

Attacks can be divided into two main types. On the one hand, external attacks are caused by nodes that are outside of the network. These attacks do not possess the cryptographic material required to understand the messages exchanged. Here the protection using authentication techniques is usually enough to ward off most attacks apart from attacks of interference (jamming) attempted on the whole network [21]. On the other hand, internal attacks aimed at undermining the nodes of the network can bypass the established protocol in order to obtain information, or can arrive, in our case, to provide a

false aggregated result to the BS. The simplest case would be for an attacker to compromise a CH directly so that erroneous results would be provided to a higher level of the hierarchy. A more complicated case to be addressed would be an attack that compromises a high proportion of nodes, causing the nodes to transmit wrong information, or to take control of some of the clusters.

The primary objective of our protocol is to secure CH nodes, which are the primary source of the attacks. We pay little attention to the case of compromised basic sensors (non-CH nodes): it is sufficient to apply a voting mechanism or monitoring (such as the protocol of Zhu et al [24]). We thus assume that an attacker could compromise some nodes and/or all of the CH nodes (as appropriate), without a resultant loss of the network. Our protocol guarantees that if the CHs are compromised, the final validated aggregate is still transmitted to the BS. This is enabled by using a voting mechanism at each level that involves all of the nodes belonging to the same cluster of level 1 and also by the regular distribution of keys by the BS so that a result obtained from passing through a compromised CH is not understandable by the BS.

2.3. Ultra Hierarchical Cluster Formation

2.3.1. Phase 1: Initialization

This phase occurs before network deployment. The BS first generates a chain of keys $K\{n/bs\}$ needed to perform broadcasts to all authenticated sensors – to create the cluster training in level 2 and higher, or possibly for other Operations: alerts, etc. The BS then loads each sensor u with a unique identifier denoted ID_u , a secret key $K_{bs,u}$ shared with itself to ensure some future communications in unicast (to guarantee confidentiality and authentication), and the first key $K\{0/bs\}$ of its key chain, to make broadcasts throughout the network (using protocol μ TESLA: security for authentication). Finally, the BS loads each node u with the necessary hardware cryptographic key established with all of its neighbors to secure communications between two neighboring nodes; two neighboring nodes u and v have a shared key $K_{u,v}$.

2.3.2. Phase 2: Clusters of the First Level

In this phase, we chose the protocol in [21], which is based on disjoint cliques formed by the graph representing the WSN. This choice ensures that within each cluster, each member has access to another member in a single hop, which significantly reduces the energy cost and also ensures that we can perform, for example, voting mechanisms in the aggregation phase. We emphasize once again that the protocol in [21] is secure, guaranteeing against a majority of external and internal attacks. This protocol uses the key $K_{u,v}$, established between two nodes u and v . It also use broadcast authentication with μ TESLA: each node u generates a string of keys $K\{n/u\}$ using its hardware cryptographic key and distributing the first key of the chain $K\{0/u\}$ to its neighbors. Once this set of keys is in place, the protocol is performed according to [21]. Finally, whenever a CH is determined, it sends a message to the BS containing the list of members of its cluster. Hereafter CH^1 will denote a cluster head of level 1.

2.3.3. Phase 3: Recursive Construction of Clusters of Higher levels

Next we rely on a mechanism similar to a virtual architecture [23] to partition our clusters of level 1 into clusters of level 2 or higher. The BS – the only trusted entity in the network – is in charge of this operation. The BS knows the architecture of the clustered network in level 1. It determines – based on the number of nodes – a range coefficient C_p (between 0.1 and 1). $C_p=1$ represents 100% of the distance separating the BS from the most distant sensor in the WSN. This parameter can be determined by successive broadcasts from BS during phase 1. The parameter C_p is used by the BS to create coronas, and it also fixes an angular coefficient C_a (between 1° and 360°). The parameter C_a is used by the BS to create angular sectors. Everything then depends on the wishes of the administrator of the system. We use low C_a and C_p to create many levels, and we use large C_p and C_a to get a low number of levels. Other calculations related to the virtual architecture are not detailed in this document as they are already fully studied [23]. The BS is thus able to partition the network by making broadcasts of different ranges and angles as well as, according to C_p and C_a , to create different coronas and angular sectors. An intersection of a corona and an angular sector is a cluster of level 2, as shown in Figure 1.

Each area so defined by the BS is denoted by a pair of integers (angular sector number, number of corona number). In summary, the BS creates a virtual architecture on the clustered network of level 1, W^1 . Levels $N > 2$ are built in the same way. The process of generating clusters of level N , $N > 1$, is as follows:

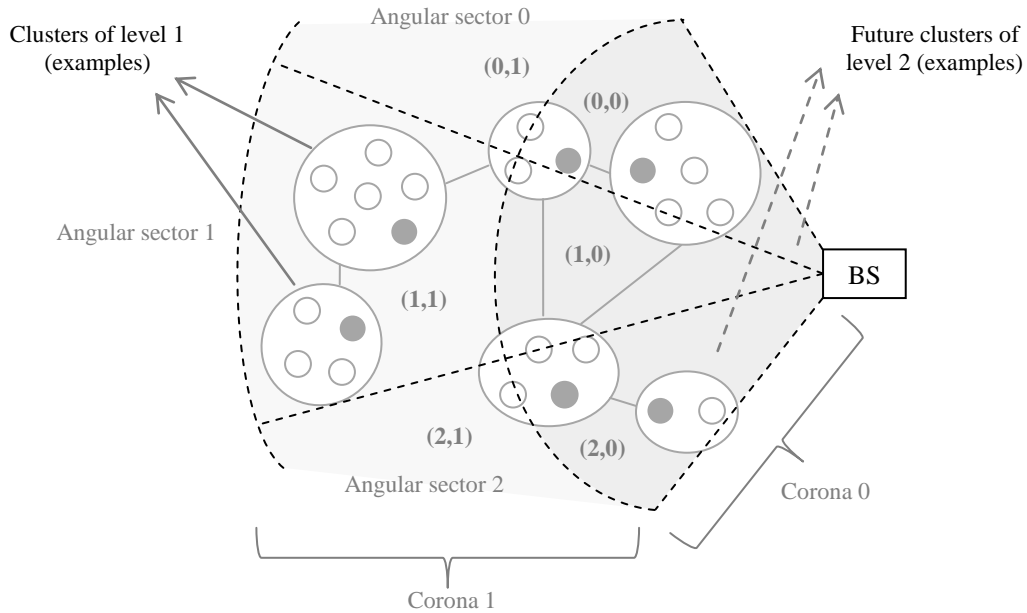


Figure 1. Ultra hierarchical clustered WSN: Virtual architecture of level 2 built by the BS with their clusters of level 2: $C_p = 0.5$ and $C_a = 40^\circ$.

1. BS broadcasts the key $K\{n/bs\}$ to communicate the authenticated pair of integers to nodes of different areas. BS successively broadcasts a message of type (angle and crown) according to C_a and C_b and therefore at different angles and crowns. For each area (angle, crown) defined by BS as a function of C_a and C_b :

$$BS \rightarrow W^1 : Param, MAC_{K\{j/bs\}}(Param), K\{j/bs\}$$

With $K\{j/bs\}$ the current key of the chain key $K\{n/bs\}$, $Param$ is the pair of integers to be broadcast and that corresponds to a certain area according to the BS. W^1 is the set of all nodes in level 1, i.e., in the network.

2. Each node u then receives the message: it first authenticates the key (disclosed) $K\{j/bs\}$ using the key previously stored, $K\{j-1/bs\}$. It uses H , the irreversible hash function it holds, and checks the correspondence $K\{j-1/bs\} = H(K\{j/bs\})$. Once this first step is done, the node verifies the authentication provided by the MAC attached to the message and can update its last known key. This is a simple application of the protocol μ TESLA. Finally, a node becomes aware of the parameters (angle, corona) that affect it. Next each CH^1 communicates the parameters it holds to all members of its cluster of level N using the key $K\{j/CH^1\}$ of the chain $K\{n/CH^1\}$ of the cluster head CH^1 for authentication (broadcast with μ TESLA). This is to put into agreement all members of a cluster of level 1 (in case some members may have a different parameter). This step is done as follows:

$$CH^1 \rightarrow W^1\{CH^1\} : Param, MAC_{K\{j/CH^1\}}(Param), K\{j/CH^1\}$$

3. A node reads the parameter and updates its local value if there is a difference with the value transmitted by the BS, and returns a receipt containing the final value to its CH^1 , which is authenticated by the shared key with the BS: $K_{u,bs}$.

4. Each CH^1 then receives a set of answers that it cannot read or modify. Once all the responses are received, it sends them to the BS together with the signature $K_{ch,bs}$, including the IDs of members who have not responded.

5. Next, the BS receives a message, authenticates it, and then checks one by one all of the receipts from the cluster nodes. If a discrepancy is noticed, or if it does not receive a message from a CH^1 , it takes action (e.g., re-election or banishment of nodes).

6. At this point, clusters of level $N > 1$ are implicitly formed: a cluster of level $N > 1$ is a set of cluster heads of level $N-1$, whose members have the same parameters (angle, corona), as illustrated in Figure 2.

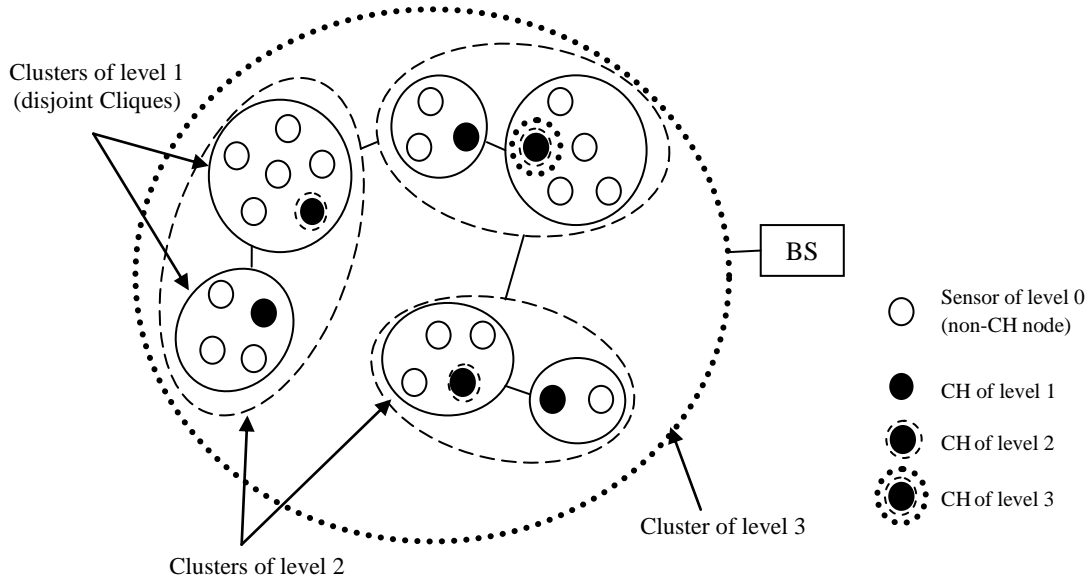


Figure 2. Ultra clusters formation

7. Next, a CH^{N+1} is elected among all CH^N as shown in Figure 2. A cluster-head at level $N + 1$ is therefore also a cluster-head at level $N, N-1, N-2, \dots, 1$. Here we do not detail the procedure for the election itself. This procedure depends on the parameters desired by the user (depending on the available energy of the identifier, etc.).

8. Once an election is made, each newly elected CH^N warns the base station, which restarts the whole procedure at a higher level, by multiplying C_p and C_a by a factor θ to be determined. Cluster formation is complete when the BS sees that there is no more than a cluster-head for level N .

Cluster formation is now complete and, in theory, does not have to be re-run. However it is possible that adjustment operations are performed internally, such as update operations: fault tolerance, adding nodes, banishment mechanisms or reelection in the case of a detection of a malicious sensor.

2.4. Data Aggregation Protocol

Once we establish our ultra multi-level clustered structure, aggregation of data can be performed at various times during the life of the network. This aggregation consists of a set of steps at each level of our hierarchy because CH^1 connects to the BS via CH^2, CH^3, \dots, CH^N . The BS, being the only trusted entity in the network, is used solely for the purpose of securing information that would be passed without having to directly manage or make any aggregation. Clearly, the BS is responsible for disseminating the different keys that enable us to provide security information to be exchanged between all nodes. Our aggregation mechanism is quite different from what is proposed in the literature

due to our second hypothesis; here, the goal is to protect (above all) CHs, which are located at strategic locations and may allow easy access for potential attackers. Imagine a CH being compromised; if the CH is the only carrier of the aggregation of a dataset, then the information could easily be modified without anybody noticing. Thus, we introduce a voting mechanism, similar to the SAPC protocol [3, 24]: in clusters of level 1 it is not the CHs that achieve the aggregation but all of the other members of this cluster, to more easily establish a probability of overall honesty with the members of the cluster and to narrow down the group that is more likely to attack. This solution allows us to ensure a correct result with less than 50% of the nodes compromised in any cluster of level 1, and as well as with all cluster-heads.

2.4.1. Phase 1: Key Distribution

The BS is a trusted entity. The BS generates two private keys, K1 and K2, which it keeps and refers to all nodes in the network; it releases K1 to all nodes that are non-CH nodes, and it releases K2 to all CH nodes. This exchange must be done in a confidential and authenticated way, for example, by using the key $K_{bs,u}$ to provide a key from the BS to a node u . Another solution would be to use a delivery mechanism, such as μ TESLA, allied to an aspect of ensuring confidentiality [17]. We assume here that once the keys are held by different nodes, frauds are detectable: if a malicious CH were to transmit its key K2 to a compromised member of its cluster, the fraudulent transaction would be detected by a third member of the clique (see security analysis in the next section). Finally, we raise the specific case of a cluster at level 1 consisting of a single member. In this case, the CH is informed of the keys K1 and K2 to manage transactions that are usually carried out only by two types of nodes in a cluster.

2.4.2. Phase 2: Recursive: Data Aggregation at Level N

The keys are now known from the various sensors, so aggregation can begin. Initially, the aggregation takes place at level 1, i.e., inside clusters of level 1, and then aggregation continues from level to level. This phase is the same for all levels and is divided into eight steps:

Step 1. For each cluster of level 1: each member, except CH^1 , sends to other members of its cluster, for example, $*W^1$, the data that it holds. This can be expressed as:

$$u \rightarrow *W^1 : \text{Data}, \text{MAC}_{K_{u,j}}(\text{Data}), K_{u,j}$$

$*W^1$ are all sensors of level 1 that are not CHs. $K_{u,j}$ is the key shared by u and each sensor j belonging to $*W^1$.

Step 2. All non-CH nodes carry out the aggregation of data received, and they encrypt the result with the private key K1. In theory in a healthy system, every sensor has the same encrypted value, which we denote as $K1(\text{data})$.

Step 3. Next each node sends this encrypted result to its CH^N by authenticating the message using the key K_{u,CH^N} : for each sensor u belonging to $W^N\{CH^N\}$ we have $u \rightarrow CH^N : K1(\text{Data}) \parallel \text{MAC}_{K_{u,CH^N}}(K1(\text{Data})) \parallel K_{u,CH^N}$

Step 4. Because a CH^N does not have the key K1, it may not reproduce any of the results received, nor understand them. It then makes a voting mechanism, as in [3, 24]. In a healthy system, all of the encrypted values received are equal (in theory), as calculated from sensors with the same set of data to be aggregated. In a network with compromised sensors, the CH^N has to choose the value that occurs most often to have a guarantee of less than 50% of the sensors being compromised.

Step 5. Once the resulting value (as hash) is selected, the CH^N then encrypts the value with its key, K2, to send the chain K2 (K1 (Data)) to the CH^{N+1} if K2 is found, and to the BS otherwise (end of the protocol). This is usually done in multi-hops, and the nodes on the path are not able to understand the message, which is encrypted with a key that they do not possess. Trivially, if $CH^N = CH^{N+1}$, then the node does not need to do this but goes directly to Step 8.

Step 6. The message is sent from CH^N to CH^{N+1} , usually in a multi-hop fashion.

Step 7. CH^{N+1} receives the message, and decrypts the message with its key, K2.

Step 8. CH^{N+1} distributes the decrypted message to all members of its cluster of level 1. These members are able to decrypt the message with their key $K1$. This step does not run if the message received in Step 5 is sent to the BS. This step can be expressed as follows:

$$CH^{N+1} \rightarrow W^{N+1} \setminus \{CH^{N+1}\} : K1(Data) || MAC_{K_{u,j}}(K1(Data)) || K_{u,j}$$

where $K_{u,j}$ is the key shared by u and each sensor j belonging to $W^{N+1} \setminus \{CH^{N+1}\}$. Next, go to step 2.

2.4.3. Phase 3: Reception of Data by the BS

This phase begins when a message of the form $K2(K1(Data))$ is owned by CH^{N+1} in our hierarchy, at the end of Step 5. The information is sent to the BS using the key $K_{bs,CH^{N+1}}$ for authentication, as follows:

$$CH^{N+1} \rightarrow BS : K2(K1(Data)) || MAC_{K_{bs,CH^{N+1}}}(K2(K1(Data))) || K_{bs,CH^{N+1}}$$

On receiving the final information sent by CH^{N+1} , the BS is able to decrypt the message with the keys $K1$ and $K2$, which it knows, and is possibly able to detect errors.

3. Security Analysis

First, we must justify the keys mechanism, whereby keys are broadcast from the BS to the CHs and to nodes that are not CHs. Our problem here is to transport information shared by the sensors in $W^N \setminus \{CH^N\}$ to those in $W^{N+1} \setminus \{CH^{N+1}\}$, while going through a CH^N , one or more intermediate nodes and finally the CH^{N+1} . The obvious solution is as follows: each non- CH^N sensor has a key $K1$ that is not known by the CH^N , and there is a key $K2$ shared by the CH^N that is not known by the non- CH^N nodes. Thus, if we consider that a set of non- CH^N nodes perform data aggregation and transmit the result encrypted with $K1$ to their CH^N , then CH^N could not reproduce or understand the information received. Similarly, the CH^N is able to detect if there is a transmission problem, as well as to detect the higher level nodes waiting for a message from this CH^N (in the case where a malicious node wants to send nothing). The second key $K2$, possessed only by the CH^N s, then encrypts the string $K1(D)$ that is received. The result $K2(K1(D))$ is then forwarded to gateway nodes or to intermediate nodes. Their modification, reading or copying is difficult and detectable, until the CH^{N+1} receives the string. CH^{N+1} is then able to distribute the information decrypted by $K1(D)$ to members of its clique.

Obviously, as indicated previously, the situation in which a cluster of level 1 has only one node leads to providing this node with keys $K1$ and $K2$. If it were to be compromised then, regardless of the details, we could not prevent it from sending erroneous information. Thus an election of a CH at the top level must take into account the number of cluster members, so that a compromise does not result in loss (in the worst case scenario) of the final aggregated data from the network.

The principles of key management that we introduce ensure some security for data transmission. However, we must raise two cases before drawing conclusions:

3.1. Case 1

In a clique, a CH as well as a node is compromised. In this case, they may decide to exchange their keys $K1$ and $K2$.

(i). **Solution:** if such an exchange occurs, another member node of the clique should be able to detect the inappropriate exchange, and to take an action.

3.2. Case 2

Message $K2(K1(D))$ is sent from a CH^N to a CH^{N+1} . If the message passes through a CH node and a non- CH node, and both are compromised, then they are able, by exchanging information several times, to obtain D and to retransmit erroneous information.

3.2.1. Simple Solution

If such non-consistent exchanges occur, then detection should be possible by using nearby nodes, which capture that a message of the form «K1(D)» has been sent by a node, which should have sent a message of the form «K2(K1(D))». This is effective for most cases, except for when the message passes through a string consisting of a CH node flanked by two non-CH nodes, with all three being compromised. In this case, we can think of an expiration mechanism: the time for the message to be decrypted, modified and re-encrypted should therefore be noticeable by a higher level entity.

3.2.2. More complex solution, however more efficient

We slightly revise the protocol, and distribute only the key K1 in the first phase. The key K2, usually distributed and shared by the CHs, would be changed by a set of keys to make the authentication end-to-end between a CH^N and a CH^{N+1} . Therefore, the BS should distribute a private key for each CH^N and a set of keys to each CH^{N+1} corresponding to those distributed to the CH^N .

Thus, we can analyze the security of our aggregation protocol in two ways: in the best case, i.e., taking into account the solutions presented ((i), 3.2.1 and 3.2.2), and in the worst case, i.e., without considering these solutions, including (3.2.2), which causes additional distribution costs.

In the best case, the solutions are applied, and they allow us to assert that our protocol offers resistance to external attacks (except in special cases already introduced) and results in a compromise of all CH nodes and a compromise of less than 50% of the aggregators in the clusters of level 1.

In the worst case, the effectiveness in terms of safety is no higher. Here a CH node and a non-CH node, both compromised, can cooperate by exchanging their keys and can put the network into bankruptcy. Nevertheless, protection is ensured in this case against most external attacks (in the same way as above), and against the compromise of all CH nodes or less than 50% of the aggregators in all clusters of level 1.

Therefore, our protocol ensures minimal defense and is effective, with difficult constraints to meet. However, only the base station is trustworthy, and it should not carry out the aggregation process. In addition, we present a basis for security that is possible to improve, for example, by increasing the frequency distribution of the keys.

4. Simulations Results

To measure the effectiveness and to demonstrate the flexibility of our training protocol, we conducted simulations that we describe in Figure 3 of this section. We have successively taken a set of 50, 150 and 300 clusters of level 1, and a set of values for our coefficients C_a and C_p . For each possible case, we made 10 different simulations to evaluate the average location of the base station in the network, the distribution of different sensors, and the average number of clusters of level 2 that are possible to construct. The results are presented in Figure 3, which allows us to have a view of the flexibility provided by our training; everything is a function of C_a and C_p values, which are chosen depending on the application and, in our case, the density of information to be aggregated. As can be seen in the figure, a decrease in the values of C_a and C_p results in an increase in the number of clusters of level 1, which allows us to have control over the number of clusters at level 2. On level 1, very suitable simulations were performed in the original article by Sun et al. ([21]). On higher levels, the number of clusters depends on a coefficient, for instance λ , of C_a and C_p : consider the case where $\lambda=2$, $C_a=180$ and $C_p=0.5$ for level 2. Then, at level 3, we have $C_a=360$ and $C_p=1$, with a single cluster of level 3. Also, we can easily estimate the number of clusters at level 2, according to C_p and C_a and in the case of an ideal situation: the base station is placed at the center of the network and the sensors are properly distributed around. The approximate number of clusters of level 2 would then be limited to the following calculation, taking C_{11} as the number of clusters of level 1:

$$C_{12} = (360/C_a) \times (1/C_p). \text{ If } C_{12} > C_{11} \text{ then } C_{12} = C_{11}$$
$$\text{If } C_{12} > C_{11} \text{ then } C_{12} = C_{11}$$

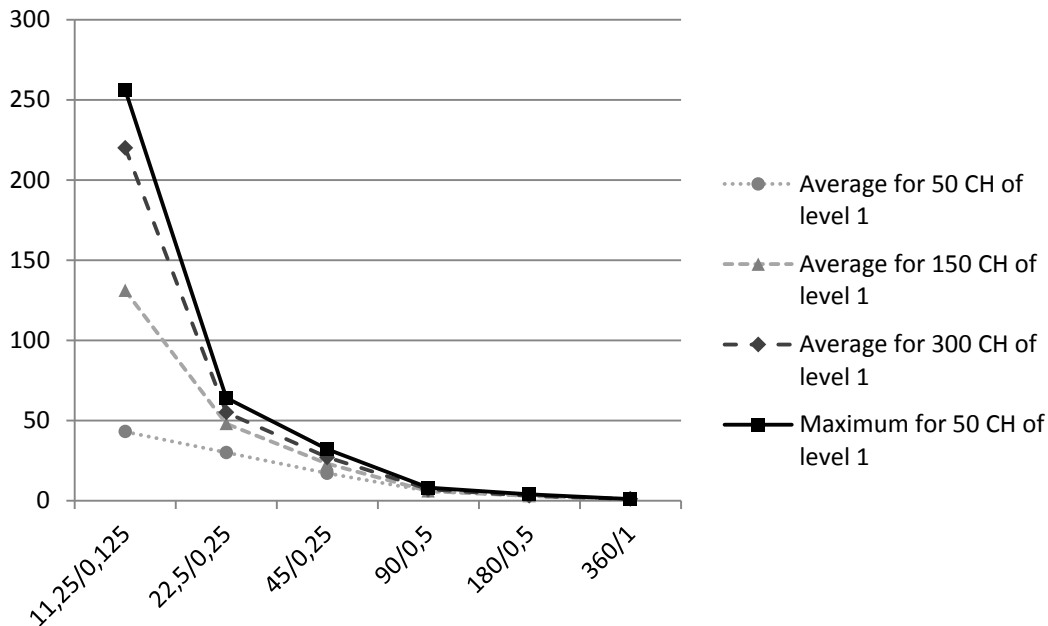


Figure 3. Number of clusters of the second level as a function of C_a and C_p .

5. Conclusions

This paper presents an aggregation protocol in a WSN. We were interested in the case where only the base station could be considered as a trusted entity, and we have proposed a mechanism based on an ultra clustered structure, enabling us to perform distributed aggregation throughout the network. In the end, the base station receives the total aggregate incoming data, and it does not perform additional operations, which results in a capital conservation of its energy. Our approach ultimately improves the life of the network through a more equitable distribution of operations and energy use (which is not the case in the SAPC protocol). If constraints exist, such as in our analysis, this protocol is still effective against most attacks and is a suitable security solution considering the assumptions of trust. One avenue for future research may be to introduce mobility into the network, for example, by assuming a WSN that has both static sensors and movable actuators.

6. References

- [1] J. N. Al-Karaki, R. UI-Mustafa and A. E. Kamal, "Data Aggregation in Wireless Sensor Networks - Exact and Approximate Algorithms". Workshop on High Switching and Routing, pp. 241- 245, April 19-21, 2004.
- [2] I. F. Akyildiz, W. Su and Y. Sankarasubramaniam, "Wireless sensor networks: a survey", Computer Networks (38), pp. 393-422, 2002.
- [3] C. Bekara, M. Laurent-Maknavicius, Kheira Bekara, "SAPC: A Secure Aggregation Protocol for Cluster-Based Wireless Sensor Networks". MSN'2007, pp. 784-798.
- [4] R. Blom, "An Optimal Class of Symmetric Key Generation. Advances in Cryptography", Proc. of EUROCRYPT 84, Lecture Notes in Computer Science, 209, pp. 335-338, 1984.
- [5] C. Blundo, A. D. Santis, A. Herzberg, S. Kuttan, U. Vaccaro and M. Yung, "Perfectly secure key distribution for dynamic conferences", In Proc. of the 12th Annual International Cryptology Conference on Advances in Cryptology, Lecture Notes in Computer Science, vol. 17, Springer-verlag, pp. 471-486, 1992.
- [6] H. Chan, A. Perrig and D. Song, "Random Key Predistribution Schemes for Sensor Networks". In IEEE Symposium on Security and Privacy. pp. 197-213, 2003.

- [7] B. B. Chen, H. Yu, “Secure Aggregation with Malicious Node Revocation in Sensor Networks, ICDCS 2011.
- [8] T. Dimitriou T. and I. Krontiris, “A Localized, Distributed Protocol for Secure Information Exchange in Sensor Networks”, In Proc. Of the 19th IEEE International Parallel and Distributed Processing Symposium, 2005.
- [9] D. Estrin and R. Govindin, “Impact of Network Density on Data Aggregation in Wireless Sensor”, Proceedings of the 22 nd International Conference on Distributed Computing Systems, pp. 457- 458, July 2-5, 2002.
- [10] K. Frikken and J. Dougherty, “An Efficient Integritypreserving Scheme for Hierarchical Sensor Aggregation”, In WiSec, 2008.
- [11] N. Gura, A. Patel, and A. Wander, “Comparing elliptic curve cryptography and RSA on 8-bit CPUs”, In Proceedings of the 2004 Workshop on Cryptographic Hardware and Embedded Systems (CHES), 2004.
- [12] P. Haghani, P. Papadimitratos, M. Poturalski, K. Aberer, and J. Hubaux, “Efficient and robust secure aggregation for sensor networks”, In NPSec, 2007.
- [13] L. Hu and D. Evans, “Secure Aggregation for Wireless Networks”, Proceedings of the 2003 Symposium on Applications and the Internet Workshops, pp. 384- 2003.
- [14] C. Karlof, N. Sastry and D. Wagner, “TinySec: A Link Layer Security Architecture for Wireless Sensor Networks”, SenSys04, November 35, 2004.
- [15] B. Krishnamachari, D. Estrin and S. Wicker, “The Impact of Data Aggregation in Wireless Sensor Network”, Proceedings of the 22nd International Conference on Distributed Computing Systems, pp. 575- 578, July 2-5, 2002.
- [16] D. J. Malan, M. Welsh, and M. D. Smith, “ A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography”, In SECON, October 2004.
- [17] A. Perrig, R. Szewczyk, V. Wen, D. Cullar and J. D. Tygar, “ Spins: Security protocols for sensor networks”, In Proc. of the 7th Annual ACM/IEEE International Conference on Mobile Computing and Networking, pp. 189-199, 2001.
- [18] S. Nath, H. Yu, and H. Chan, “Secure Outsourced Aggregation via One-way Chains”, In ACM SIGMOD, 2009.
- [19] B. Przydatek, D. Song and A. Perrig, “SIA: Secure Information Aggregation in Sensor Networks”. SenSys03, November 5-7, 2003.
- [20] J. Sen, “A Robust and Secure Aggregation Protocol for Wireless Sensor Networks”, In Proceedings of the 6th International Symposium on Electronic Design, Test and Applications (DELTA 2011), Queenstown, New Zealand, 17 - 19 January 2011.
- [21] K. Sun, P. Peng, P. Ning and C. Wang, “Secure distributed cluster formation in wireless sensor networks”, 22nd Annual Computer Security Applications Conference, December 11-15, 2006.
- [22] G. Taban and V. Gligor, “Efficient handling of adversary attacks in aggregation applications”, In ESORICS, 2008.
- [23] A. Wadaa , S. Olariu , L. Wilson , M. Eltoweissy , K. Jones, “Training a wireless sensor network”, Mobile Networks and Applications, v.10 n.1-2, p.151-168, February 2005.
- [24] S. Zhu, S. Setia, S. Jajodia and P. Ning, “An Interleaved Hop-by-Hop Authentication Scheme for Filtering of Injected False Data in Sensor Networks”, Proceedings of the 2004 IEEE Symposium on Security and Privacy, pp. 259-271, May 9-12, 2004.