Original Research Paper

Implementation of Homomorphic Encryption for Wireless Sensor Networks Integrated with Cloud Infrastructure

Christophe Ishimwe Ngabo and Omar El Beqqali

Department of Computer Science, Faculty of Sciences - Sidi Mohamed Ben Abdellah University, Fez, Morocco

Article history
Received: 08-07-2018
Revised: 10-09-2018
Accepted: 7-02-2019

Corresponding Author:
Christophe Ishimwe Ngabo
Department of Computer Science, Faculty of Sciences - Sidi Mohamed Ben Abdellah University, Fez, Morocco
Email: christophe.ngabo@usmba.ac.ma

Abstract: With the current explosion of cloud services, it is easy for everyone to store the prodigious amount of data on remote servers. On the other hand, service providers are given full access to our information, or in the worst case, can be intercepted by malicious people. The solution is to encrypt the data to make it completely secret. But then, we can no longer manipulate these data remotely (retouching the photos, looking for words in a text, performing calculations ...). A new form of cryptography, which is just beginning, promises precisely to offer this security while allowing the encrypted data to be manipulated by the authorized users, it named “Homomorphic Encryption”. This paper presents the implementation of a homomorphic encryption to ensure the confidentiality of aggregated data from wireless sensor networks and also demonstrates how to use it through the cloud infrastructure in order to perform arithmetic operations directly on the encrypted data without decrypting them. After a theoretical comparative study on homomorphic encryption algorithms, we picked out the Domingo-Ferrer’s cryptosystem on the basis of criteria mentioned during the survey. This cryptosystem requires to split the message into several fragments (at least two fragments) before the encryption process. For each implementation case performed with two, three and four fragments of the message, 10,000 messages were sent so that the battery levels of the sensor nodes dropped by 6%, 7% and 8% respectively, which led us to conclude that as long as the number of fragments increases, the power consumption in the network is increasingly significant because of excessive processing and a large amount of data to be transmitted over the wireless network, generated by the Domingo-Ferrer’s cryptosystem.

Keywords: Cloud Computing, Data Aggregation, Wireless Sensor Networks, Homomorphic Encryption

Introduction

Since their inception, wireless communication networks have been increasingly successful in the scientific and industrial communities. Thanks to its various advantages, this technology has been able to establish itself as a key player in the current network architectures.

Nowadays, with the advanced research in the new information and communication technologies, it is very easy to develop small embedded systems of communication, not expensive, comprising at least a power unit, a data collection unit (sensing unit), a processing unit and a transmission unit for mainly surveillance, discovery and detection purposes (Zheng and Jamalipour, 2009). These microsystems (sensor nodes or motes) can integrate a wide range of sensors collecting information from the physical environment for various applications such as military, medical, industrial, natural disasters, etc. (Sohraby *et al*., 2007).

Thanks to these transmission units, these nodes communicate with each other to build a network, often wireless, which can generate thousands of nodes transforming the state of an observed physical quantity (temperature, pressures, humidity, vibration, etc.) into signals that can be converted into codes in order to be processed by computers. This information is routed over the network from one node to another to a collection point called the "Base Station" or the "Sink" that serves as a gateway to devices located on other external networks such as the internet.

The collected data from the Wireless Sensor Network (WSN) can be exploited in real-time by users on the external networks but they also need to be stored for later use, however it is not a simple matter to find enough processing power and storage for this multitude of data. The Cloud infrastructure provides several benefits such as mass storage, the demand for self-service access to the network wide resource consolidation, measurement service, mass scalability, consistency, virtualization, software low-cost, distribution, service orientation and advanced security (Buyya *et al*., 2011). All these qualities are essential to fill the gaps found in the WSN.

Consider a simple example of weather, a city where each neighbourhood has its own weather station, data on rainfall, temperature, humidity, wind, etc., will be stored in the cloud's database to serve statistics for example. Suppose each weather station has a multitude of redundant sensors, in fact we will not use a single temperature sensor for a single station but several and make an arithmetic average to be sent to the cloud. In this example, the sent values are plaintexts, an attacker can intrude into the network and tries to visualize the information that transit, but the big concern is the falsification or unauthorized modification that can occur in the cloud. An effective way of ensuring the confidentiality of data from the sensor network to the system administrator via the cloud is required. Such a system must be composed of four major actors, namely the sensor network, the cloud infrastructure, an administration subsystem and a web service for end users (citizens).

Imagine we decide to encrypt the data from the sensor network with the symmetric algorithms such as AES, whenever the administration wants to perform the calculations (statistics) on the stored data, it must either send a secret key to the cloud for decryption (a potentially dangerous solution because the sending of the key on the network is likely to be intercepted but the administration can benefit the computing power of the cloud) or downloading the encrypted data and performing the statistics locally (solution without risk but the administration machine will have a huge data capacity to process, which requires a powerful machine). In both cases, we do not have the best solution to remedy this problem.

Imagine a solution that does not require a decryption key to perform these operations (statistics, forecasting, etc.) on encrypted data, i.e., the administration will not have to take the risk of sending that decryption key over the network and in addition, he/she will benefit the computing power of the cloud. The administrator's workstation only has to decrypt the results of the statistics using its key that it keeps locally, such solution can be the use of *Homomorphic Encryption* algorithm.

A homomorphic encryption scheme makes possible to perform the arithmetic operations on the encrypted data without decrypting them. Homomorphic encryption also reduces the processing time of initially encrypted data. Homomorphic encryption in the cloud computing ensures data confidentiality and minimizes the risk of attacks (Tebaa *et al*., 2012).

By applying a homomorphic encryption scheme in a system, it can avoid the risk that can occur when exchanging the secret keys. In short, this paper focuses on the use of homomorphic encryption algorithm applied to the distributed system combining together the sensor network, cloud infrastructure and end users.

The homomorphic encryption implemented in this paper has been the subject of a theoretical study on the different symmetric and asymmetric algorithms. The choice of the implemented algorithm was based on the criteria concerning computational difficulties such as exponentiations that are not adequate in the WSNs due to the congestion of the computational resources. The implementation was carried out with the Domingo-Ferrer (2002) encryption scheme on the sensor nodes embedding 16-bit microprocessors, which are moderately sufficient for cryptography calculations. This encryption algorithm requires that the message must be split into several fragments before proceeding to encryption that is why we have performed several experiments with the variable number of the fragments (2, 3 and 4) in order to understand the impact of this message splitting on the state of the battery of each sensor node. The results obtained showed that after sending 10,000 messages the battery levels decreased respectively by 6%, 7% and 8%, which shows that the more the message is fragmented the more the energy of the WSN is slightly diminished but the level of data security (confidentiality) is increasing.

In this paper we started with the introduction on the wireless sensor network, the importance of integrating it with Cloud Computing as well as the security using homomorphic encryption. Then we made a summary about literature and related works to our research topic. The next section focuses on the importance of homomorphic encryption on aggregated data in the WSNs. The fourth section is the security modeling required for aggregated data in the WSN. The fifth section is the implementation of the Domingo-Ferrer algorithm in the WSNs. It is in the sixth section that we presented the results aforementioned and finally we concluded in the seventh section.

Literature Review and Related Works

Cloud Computing permits companies to increase capacity quickly without the need for new infrastructure investment and similarly companies can decrease capacity quickly and efficiently. Cloud Computing is principally designed and promoted to be data centre centric and efficient interaction with the outside world is an area where improved solutions are being sought (Ahmed and Gregory, 2011).

In fact, Cloud Computing offers several advantages such as mass storage, the demand for self-service access to the wide network, resource consolidation, the measurement service, massive scalability, consistency, virtualization, low cost software, distribution, orientation of service and advanced security. All these qualities are essential to fill the gaps found in the WSN (Ngabo and Beqqali, 2016).

As we know, the WSN is designed to collect data in the real world, but the question is what to do with this data when the organisations no longer need it. There are many reasons for the data to be kept including historical, future research and re-analysis at some future point in time. There is a possible linkage between WSN and Cloud Computing and the eventual shift of data into the cloud and over time into the public domain (Ahmed and Gregory, 2011).

By definition (Tebaa *et al*., 2012), we have an homomorphic encryption if from $Enc(m_1)$ and $Enc(m_2)$ it is possible to compute $Enc(f(m_1,m_2))$, where f can be +, x or ⊕ and without using the secret key. According to the operations that allow to access on raw data, we distinguish three types of homomorphic encryption:

- Additive homomorphic encryption: A homomorphic encryption is called additive, if $Enc(m_1 + m_2) = Enc(m_1) \cdot Enc(m_2)$
  e.g.: Pailler cryptosystem
  $Enc(m_1) = g^{m}{}_1 \cdot r_1^N \; mod\, N^2 = c_1$
  $Enc(m_2) = g^{m}{}_2 \cdot r_2^N \; mod\, N^2 = c_2$
  $\Rightarrow Enc(m_1+m_2) = g^{m}{}_1{}^{+m}{}_2 \cdot (r_1 \cdot r_2)^N \; mod\, N^2$
  $= Enc(m_1) \cdot Enc(m_2) = c_1 \cdot c_2$
- Multiplicative homomorphic encryption: A homomorphic encryption is called multiplicative, if $Enc(m_1 \cdot m_2) = Enc(m_1) \cdot Enc(m_2)$
  e.g.: RSA cryptosystem
  $Enc(m_1) = m_1^e \; mod\, n = c_1$
  $Enc(m_2) = m_2^e \; mod\, n = c_2$
  $\Rightarrow Enc(m_1 \cdot m_2) = (m_1 \cdot m_2)^{\,e} \; mod\, n = c_1 \cdot c_2$
- Fully homomorphic encryption: homomorphic encryption is called fully, if it is at time additively and multiplicatively homomorphic.
  E.g.: In 2010, a completely homomorphic encryption scheme (Dijk *et al*., 2010) named DGHV was presented which is an application of Gentry encryption (Gentry, 2009) on integers and whose security is based on the problem of the approximate common divisor.
  Keys generations: r, p and q. Where $r \sim 2^n$, $p \sim 2^{n^2}$, $q \sim 2^{n^2}$, p and q are the prime numbers.
  $Enc(m) = pq + 2r + m = c$
  $Dec(c) = (pq + 2r + m \; mod\, p) \; mod\, 2 = m$
  For two messages m_1 and m_2, let c_1 and c_2 be their ciphertexts respectively:
  - $c_1 + c_2 = (q_1 + q_2)p + 2(r_1 + r_2) + m_1 + m_2$
  - So if $2(r_1 + r_2) + m_1 + m_2 \ll p$, then $((c_1 + c_2) \; mod\, p) \; mod\, 2 = [2(r_1 + r_2) + m_1 + m_2] \; mod\, 2 = m_1 + m_2$. Thus, DGHV realizes the property of additive homomorphic encryption
  - $c_1 \times c_2 = [q_1 q_2 p + (2r_1 + m_1) + (2r_2 + m_2)]p + 2(2r_1 r_2 + r_1 m_1 + r_2 m_1) + m_1 m_2$

  So if $2(2r_1 r_2 + r_1 m_1 + r_2 m_1) + m_1 m_2 \ll p$, then $((c_1 \times c_2) \; mod\, p) \; mod\, 2 = [2(2r_1 r_2 + r_1 m_1 + r_2 m_1) + m_1 m_2] \; mod\, 2 = m_1 m_2$. As a result, DGHV also performs the multiplicative homomorphic encryption.

A faster variant of Pailler's additive homomorphic encryption protocol was derived in 2011 focusing on the security of aggregated data in the WSNs (Wang *et al*., 2011). The idea was to speed up exponentiation in the decryption process. Performance evaluation of the obtained results indicates that the protocol they proposed can accelerate about 49% in encryption and 50% in decryption.

An optimized implementation of the elliptic curve EL Gamal based on additive homomorphic encryption has been presented in order to offer a fast multiplication point by creating a small code beneficial to the memory of the sensor nodes (Ugus *et al*., 2009). The results they obtained show that their implementation is 44% faster compared to the previous best results.

Another homomorphic encryption scheme based on elliptic curves has been proposed to avoid eavesdropping to wireless channels and to ensure energy savings in cluster-based WSNs (Elhoseny *et al*., 2016). Based on the obtained experimental results, the authors promise that their method improve network performance compared to other methods in terms of energy consumption, memory requirement, network overhead and network lifetime.

A homomorphic encryption scheme with a low computation and communication overhead has been proposed to secure compressive data gathering in the WSNs (Xie *et al*., 2017). The authors had the main goal of protecting against traffic analysis and flow tracing in WSNs. By using homomorphic encryption, data can be aggregated to reduce network traffic, but homomorphic encryption functions also increase the size of packets to be sent over the network while increasing power consumption because the consumed power is directly proportional to the amount of transmitted data.

A study was conducted in 2012 (Roy *et al*., 2012) to investigate the effect of increasing packet size for Domingo-Ferrer homomorphic encryption scheme in comparison to a symmetric cryptosystem. The results approved that symmetric encryption outperforms homomorphic encryption for small WSNs, but as the network grows the homomorphic encryption outperforms symmetric encryption.

Another implementation of the Domingo-Ferrer's homomorphic cryptosystem has been simulated on the Mica2 nodes in OMNet ++ (Ertaul and Yang, 2008). During the implementation, the authors considered three different scenarios (different key sizes and fixed message size, different message sizes and fixed key size, different message splits and fixed key size). The obtained results allowed them to conclude that splitting the message into several small messages makes it possible to increase the level of security but by penalizing the long-term network, which must be split at least the message.

Recently, a secure data collection scheme based on compression sensing has been proposed firstly in order to improve the data privacy by the use of the asymmetric semi-homomorphic encryption and secondly to reduce the computational cost by using the sparse compression matrix (Zhang *et al*. 2018b). The asymmetric encryption mechanism reduces the difficulties of encryption secret keys management and distribution, while homomorphic encryption allows the aggregation of encrypted data in the network as well as improving security and load balancing on the network. The sparse compressive matrix reduces the computation and the communication cost by compensating the increasing cost caused by homomorphic encryption. The results that the authors obtained are satisfactory or even better compared with the most related research works.

Almost the same authors (Zhang *et al*., 2018a) as in previous research work have proposed what they called Multi-functiOnal secure Data Aggregation (MODA), which encodes raw data into well-defined vectors to provide value preservation, order preservation and context preservation and thus by building blocks for multifunctional aggregation. The main purpose of this project was to compute efficiently in distributed mode even without worrying about security issues. To do this, they also used homomorphic encryption to enable in-ciphertext aggregation and end-to-end security.

All these related works to this subject concerns only the security inside the WSN, that means once the encrypted data arrived at the base station are decrypted to be processed or visualized but in this paper we want this data to continue to the cloud by being encrypted for storage. In this paper we show how it is possible that a user could perform through the cloud the arithmetic operations on this kind of data (homomorphically encrypted data) without being decrypted in order to increase the privacy level.

Homomorphic Encryption on Aggregated Data in WSN

A sensor network generally consists of a large number of sensor nodes strongly deployed in a sensing region and one or more base stations located within the collection zone. The base stations send requests or commands to the sensor nodes in the sensing zone while the sensor nodes collaborate to accomplish the sensing task and send the collected data to the base station(s). Meanwhile, base stations also serve as gateways to external networks, for example the Internet (cloud infrastructure). The sink node gathers all data from sensor nodes, performs simple processing and sends the relevant information (or processed data) via the Internet to users who have requested it or who use the information (Zheng and Jamalipour, 2009).

To send data to the base station, each sensor node can use a single-hop long distance transmission, which leads to the single-hop network architecture. However, long-distance transmission is costly in terms of energy consumption. In sensor networks, the energy consumed for communication is much higher than the energy required for data collection and computation. Therefore, it is desirable to reduce the amount of traffic and the transmission distance in order to increase energy savings and extend the life of the network.

In this case, short distance communication is highly preferred. In most sensor networks, the sensor nodes are highly deployed and the neighboring nodes are close to one another, which makes it possible to use short-distance communication. In a multi-site communication, a sensor node transmits its sensed data to the base station via one or more intermediate nodes, which can reduce the power consumption for communication.

In a multi-site network, sensor nodes can be clustered, where cluster members send their data to cluster heads, while cluster heads serve as relays to transmit data to the base station. A low energy node can be used to perform the sensing task and send the sensed data to its cluster header at a short distance while a higher energy node can be selected as a cluster head to process data from the cluster members and transmit them to the base station. This process can not only reduce the energy consumption for communication but also balance the traffic load and improve scalability as the network size increases.

In addition, aggregation of data can be performed on cluster heads to reduce the amount of data transmitted to the base station and improve the energy efficiency of the network. For example, if a weather station picks up temperature values with various temperature sensors, it is obvious that the temperature is the same for a given locality, instead of sending each value to the cluster head, the cluster head could perform small calculations on the data as the average in order to reduce network traffic while saving energy and increasing network longevity.

The objective of data aggregation is to combine and generalize the data coming from several sensor nodes in order to reduce the data to be transmitted. The majority of applications using the WSN require a certain level of security, the encryption of the data collected before transmission is preferable.

In the previous example of meteorology, sensor nodes collecting the values of the ambient temperature of the environment encrypt its data before their transmission to the cluster head, as its data come from the same neighborhood it is possible to prepare an aggregation by summing them at the cluster head and the result is sent to the base station as the same time as the number of the nodes sent these encrypted values. In fact, the average of the temperature of the environment is calculated by the base station by summing aggregated data from cluster headers and then it can decrypts this sum in order to be divided by the addition of numbers of all nodes that took these measurements.

Suppose we have three clusters *A*, *B* and *C* (showed on the Fig. 1). Cluster *A* comprises five sensor nodes, three sensor nodes for cluster *B* and four sensors for cluster *C*. In this example, the cluster headers do not collect the temperature values, they serve only to aggregate data from children nodes. Suppose we use an additive homomorphic encryption algorithm, for example the Pailler cryptosystem. The cluster head *A* receives encrypted values A_1, A_2, A_3 and A_4 from children nodes and calculate the output A_s as the result of aggregation on inputs. A_s is sent to the base station with the number 4 corresponding to the number of nodes took the corresponding values. For Pailler cryptosystem, in order to achieve the aggregation on cluster header, it necessary to perform multiplication on inputs:

$A_s = A_1 \cdot A_2 \cdot A_3 \cdot A_4,$
$A_s = g^{(v_{A1}+v_{A2}+v_{A3}+v_{A4})} \cdot (r_{A1} \cdot r_{A2} \cdot r_{A3} \cdot r_{A4})^N \ mod\ N^2,$
where: $A_1 = Enc(v_{A1}) = g^{v_{A1}} \cdot r_{A1}^N \ mod\ N^2,$
$A_2 = Enc(v_{A2}) = g^{v_{A2}} \cdot r_{A2}^N \ mod\ N^2,$
$A_3 = Enc(v_{A3}) = g^{v_{A3}} \cdot r_{A3}^N \ mod\ N^2,$
$A_4 = Enc(v_{A4}) = g^{v_{A4}} \cdot r_{A4}^N \ mod\ N^2$

By using the same algorithm, the cluster header *B* and *C* send to the base station respectively (B_s, 2) and (C_s, 3), where $Bs = g^{(v_{B1}+v_{B2})} \cdot (r_{B1} \cdot r_{B2})^N \ mod\ N^2$ and $C_s = g^{(v_{C1}+v_{C2}+v_{C3})} \cdot (r_{C1} \cdot r_{C2} \cdot r_{C3})^N \ mod\ N^2$. Then, the base station calculates the average by the following formula:

$$Average = \frac{Dec(As \cdot Bs \cdot Cs)}{4+3+2}$$

$$Average = \frac{Dec\left(g^{(v_{A1}+v_{A2}\text{v}+v_{A3}+v_{A4}+v_{B1}+v_{B2}+v_{C1}+v_{C2}+v_{C3})} \cdot \left(r_{A1} \cdot r_{A2} \cdot r_{A3} \cdot r_{A4} \cdot r_{B1} \cdot r_{B2} \cdot r_{C1} \cdot r_{C2} \cdot r_{C3}\right)^N \bmod N^2\right)}{9}$$

$$Average = \frac{v_{A1}+v_{A2}\text{v}+v_{A3}+v_{A4}+v_{B1}+v_{B2}+v_{C1}+v_{C2}+v_{C3)}}{9}$$

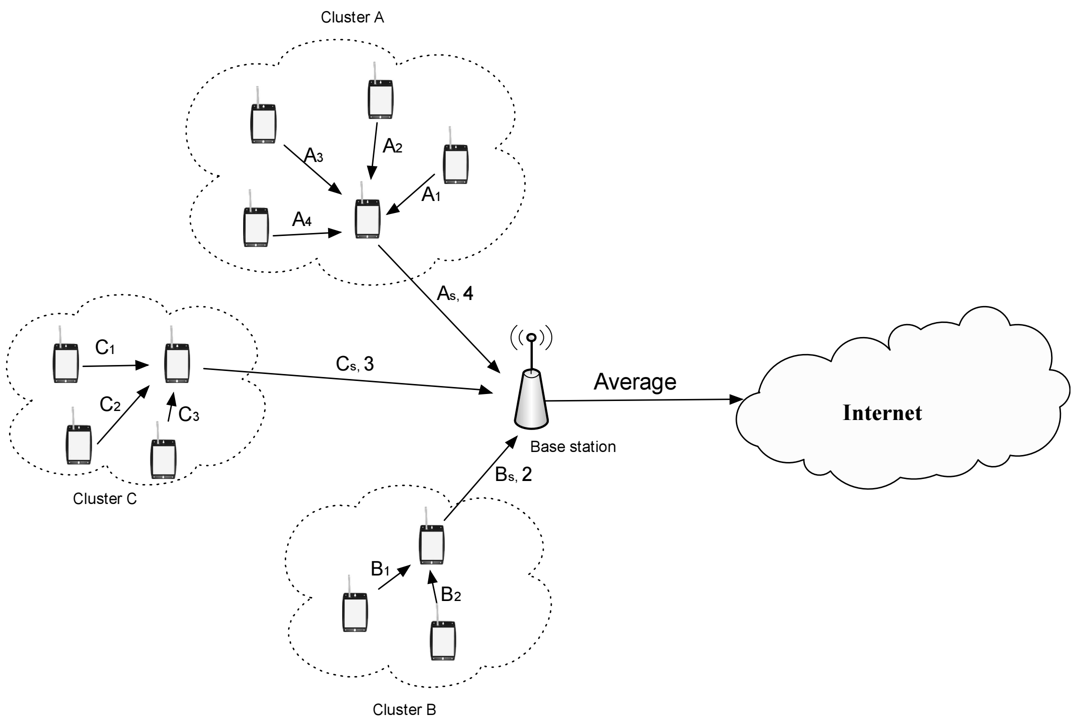

Fig. 1: Homomorphic encryption on aggregated data in the WSN

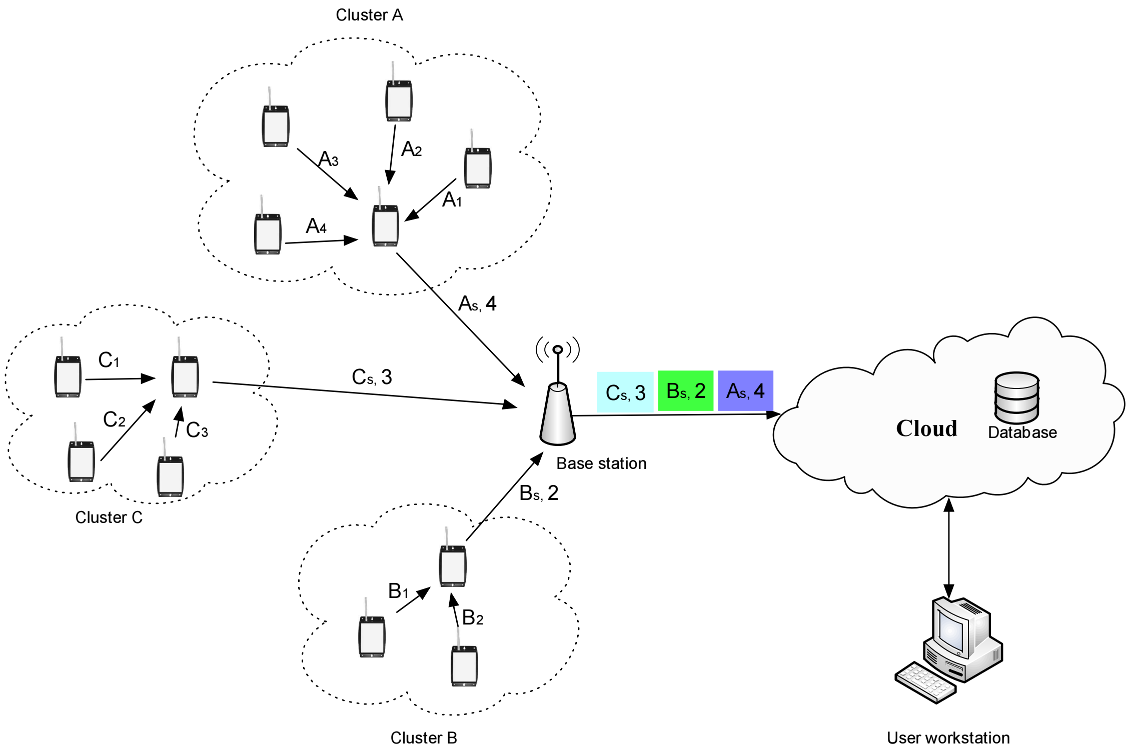

Fig. 2 Integration of cloud computing with WSN

Assume that the clusters *A*, *B* and *C* represent the neighborhoods of a city, the calculated average will be the temperature of a city that can be sent over the Internet, for example on a weather website. Since the base station has already decrypted the aggregated values from each cluster, it seems obviously that the average sent over the Internet will be in plaintext and that the confidentiality of the data is not ensured on the Internet.

Afterwards we will see how we can transmit these encrypted data on the Internet while ensuring its integrity and confidentiality.

Back to our example of weather, we want to keep the data archive in the cloud for future use, from the previous paragraph we can extend the logical architecture by sending immediately encrypted data (A_s, 4), (B_s, 2) and (C_s, 3) in the cloud instead of calculating the average at the base station, the base station will behave as a gateway without perform any processing on the data (check on the Fig. 2 for more clarification).

It is assumed that there is a database in the cloud to accommodate data from WSN. This data is stored according to the date on which it was collected. When a user wants to perform arithmetic operations on the encrypted data, for example calculating the average temperature of the day, he/she just needs to send a request to the cloud, the cloud in turn retrieve the stored data on the indicated date in the query to apply the homomorphism property to the encrypted data. In our previous example, the data is encrypted by the Pailler's algorithm, the cloud will only have to multiply all the entries of the day and the result is an encrypted representing the sum of these entries, in order to calculate the average, the result and the number *n* of the nodes that participated in the sensing are sent to the user. The workstation of the user decrypts this result and divides it by this number n to find the daily average.

Security Model

Previously, we discussed about aggregation of data in WSN and integration of WSN with cloud infrastructure using Pailler as homomorphic encryption algorithm but we can ask ourselves if this algorithm is the best for wireless sensor networks in terms of the resources needed to encrypt, like the sensor nodes' CPUs and the memories containing the programs and the encrypted data.

The majority of WSN platforms are not advanced like traditional computers that we are using in everyday life, the actual wireless sensor nodes embed few kilobits microprocessors ranging from 8 bits up to 32 bits (Johnson *et al*., 2009). So, let's take an example of an 8-bit microprocessor whose temperature value of 22 Celsius is to be encrypted with the Pailler algorithm using the following parameters:

Private key: $(p, q, r) = (5, 7, 12)$
Public key: $(N, g) = (35, 144)$
Encryption: $c = g^m \cdot r^N \; mod \; N^2$

$$c = 144^{22} \cdot 12^{35} \; mod \; 35^2$$
$$c = 144^{22} \cdot 12^{35} \; mod \; 1225$$
$$c = 348$$

At first glance, an 8-bit microprocessor cannot perform this kind of calculations because already the result of encryption is on 9 bits (348 in decimal is equivalent to 101011100 in binary), so there is an overflow what will lead to the bad results. Another remark is related to the power calculations, in spite of which the operands remain within the limit of 8 bits but this microprocessor cannot calculate 144 exponents 22 or 12 exponents 35 because there would also be an overflow. As a result, algorithms using power operations to encrypt the data are not well suited for this kind of device with lower computing power.

Apart from the computing power, another problem is related to energy consumption. To increase the level of security, we have obviously to increase the size of the encryption key. By increasing the size, the ciphertexts become huge and the sensor nodes will be forced to spend more energy for coding, modulation and radio transmissions. A long chain of data leads to more prolonged transmissions while consuming a lot of energy. As a result, the algorithms to be used for encryption must use the minimum possible key size for an acceptable level of security.

Another challenge is the choice of a type of algorithm to implement, if we use a symmetric encryption algorithm we will be pleased to share the secret key with all the sensor nodes of the network. This presents a huge risk as we know that the network's nodes are deployed in a hostile environment where the risk of being intercepted by malicious people is inevitable. If an attacker reaches physically a sensor node, he may search until he gets the secret key that this node shares with the other nodes in the network as well as the base station. The best solution is the use of an asymmetric encryption algorithm that has two keys, one public and the other private.

In this proposed system, only the end user can access the data in plaintext, i.e. the end user must generate asymmetric encryption keys. The private key must be shared with the sensor nodes either when programming the nodes or using public key infrastructures (Holohan and Schukat, 2010). When a node needs to send its collected data, it encrypts them using the public key which it has previously obtained and sends them to the next node or cluster head which in turn applies the aggregation and sends them to the base station. The base station will send the data to the cloud where they will be stored or sent in real-time to the end users. As long as the data are stored in the cloud and encrypted, only the end user can manipulate them without having to decrypt them since they are homomorphically encrypted. The end user retrieves the encrypted results found from the encrypted data and decrypts them using his/her private key.

Previously, we discussed some criteria necessary to pick out the suitable homomorphic encryption algorithm for connecting the WSN with the cloud, then we will later use a comparative Table 1 that will help us to identify a suitable homomorphic encryption algorithm to be implemented in the proposed architecture.

This comparative table above makes it easy to determine the algorithm to be implemented under some criteria before mentioned. Elliptic curve cryptosystem seems to be the best suited for low power applications (such as sensor nodes). The EC-ElGamal offers the same level of security with a smaller bit size by reducing processing overhead as compared to RSA or other homomorphic algorithms.

Smaller key sizes result in less power, bandwidth and computational requirements. This makes EC-ElGamal a good choice for low power environments. EC-ElGamal has got applications as a public key sharing scheme and as digital signature authentication scheme.

Due to these factors, ECC is better suited for low bandwidth, computational power and memory situations especially in mobile and wireless environment (Malik, 2011). So undoubtedly, we can make an ascertainment that EC-ElGamal is the strongest and the fastest (efficient) among the present techniques (Malik, 2011).

Table 1: Comparison of homomorphic cryptosystems

Homomorphic cryptosystems	Additive homomorphic operation	Exponential based encryption	Asymmetric encryption
RSA (Rivest *et al*., 1978)	No	Yes	Yes
Pailler (1999)	Yes	Yes	Yes
CMT (Castelluccia *et al*., 2005)	Yes	No	No
EC-ElGamal (Koblitz, 1987)	Yes	No	Yes
Naccache and Stern (1998)	Yes	Yes	Yes
Domingo-Ferrer (2002)	Yes	Yes	No
Goldwasser and Micali (1984)	No	Yes	Yes
Okamoto and Uchiyama (1998)	Yes	Yes	Yes
Benaloh (1994)	Yes	Yes	Yes
DGHV (Dijk *et al*., 2010)	Yes	No	No

Based on the criteria mentioned above, EC-ElGamal seems to be efficient compared to other cryptosystems but its implementation is not obvious by considering it as homomorphic cryptosystem. Take for example an elliptic curve E over prime integers defined on a finite field F_p is represented by a following equation:

$$E_p(a,b): y^2 = x^3 + ax + b \ mod \ p \tag{1}$$

Since the coefficients a and b are integers chosen from the field F_p and the cubic $x3 + ax + b$ must not have the repeated roots in F_p which is equivalent to the condition $\Delta = 4a3 + 27b2 \neq 0$ ($mod\ p$). In addition to the points of the curve, we must define a point $\boldsymbol{O}$ that we affectionately name "point at infinity". If $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ are points on $Ep(a, b)$ with $P_1, P_2 \neq \boldsymbol{O}$, let define $P_3 = (x_3, y_3) = P_1 + P_2$ by:

1. if $x_1 \neq x_2$, then $x_3 = \lambda^2 - x_1 - x_2$ and $y_3 = \lambda(x_1 - x_3) - y_1$, where $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$.
2. if $x_1 = x_2$, but $y_1 \neq y_2$, then $P_1 + P_2 = \boldsymbol{O}$.
3. if $P_1 = P_2$ and $y_1 \neq 0$ then $x_3 = \lambda^2 - 2x_1$ and $y_3 = \lambda(x_1 - x_3) - y_1$, where $\lambda = \frac{3x_1^2 - a}{2y_1}$.
4. if $P_1 = P_2$ and $y_1 = 0$ then $P_1 + P_2 = \boldsymbol{O}$.

Elliptic curve cryptography can be used to encrypt the plaintext message m into ciphertext. The plaintext message m is encoded at a point Pm from the finite set of points in the elliptic group, $E_p(a, b)$. The first step consists in choosing a generating point $G \in E_p(a, b)$ such that the smallest value of n for which $nG = \boldsymbol{O}$ is a very large prime number. The number n (called order of G) must verify the condition $1 \leq n < N$, where N is the number of all points of $E_p(a, b)$ (Rabah, 2005).

Assume that Bob and Alice intends to communicate. Each user selects a private key and uses it to compute their public-key. For example, Alice (A) selects a private-key $n_A<n$ and computes the public-key $P_A = n_AG$. To encrypt the message P_m for Bob (B), Alice chooses a random integer k and computes the ciphertext pair of points P_C using Bob's public-key $P_B = n_BG$: $P_C = [(kG), (P_m + k P_B)]$.

After receiving the ciphertext pair of points, P_C, Bob multiplies the first point, (kG) with his private-key, n_B and then subtracts the result to the second point in the ciphertext pair of points, $(P_M + k P_B)$: $(P_m + k P_B) - [n_B(kG)] = (P_m + k n_BG) - [n_B(kG)] = P_m$ which is the plaintext point, corresponding to the plaintext message m. Only Bob, knowing the private-key n_B, can remove $n_B(kG)$ from the second point of the ciphertext pair of point, i.e., $(P_m + k P_B)$ and hence retrieve the plaintext information P_m.

The problem with elliptic curves as an additive homomorphic cryptosystem is that the addition on an elliptic curve must involve only the points on that curve and the result must be a point on that curve, so if we have to encrypt the messages (data from WSN such as temperature for example) in the form of integers we must first map them to the corresponding points on the elliptic curve $E_p(a, b)$. There are several ways of mapping (Potey *et al*., 2018; Ugus *et al*., 2009) to convert the message m to the point P_m of the curve. Using a method where P_m is a multiple of the generator point G i.e. $P_m = mG$ requires the reverse function to extract the original message m from the given encoded message on the point mG. The mapping function obeys the property of homomorphism because:

$$\begin{aligned} P_{m1} + P_{m2} + P_{m3} + \ldots &= map(m_1 + m_2 + m_3 + \ldots) \\ &= (m_1 + m_2 + m_3 + \ldots)G \\ &= m_1G + m_2G + m_3G + \ldots \end{aligned}$$

where, m_1, m_2, m_3 ... are integer messages $\in F_p$. To perform homomorphic encryption the reverse mapping function is needed, such function is required to resolve the discrete logarithm problem over an elliptic curve. For this reason, the sensor nodes (cluster header of the WSN) cannot execute the reverse mapping function because of lack of computation power (Ugus *et al*., 2009). Since it seems impossible for us to implement elliptic curves as homomorphic ciphers in WSNs, we must think of the other algorithms that best meet the criteria. The DGHV and the CMT cryptosystem are better candidates approaching the criteria but these two algorithms also have weaknesses against the proposed architecture.

First of all, DGVH is applicable to the binary number which implies that each time it is necessary to send the messages (the data in the case of WSN) that it is also necessary first to convert them in binary then to apply the encryption what includes a costly step in energy consumption and time. The second disadvantage is the imposing size of the encryption key, the DGVH generates a large cryptogram that requires more bandwidth, transmit power and sufficient energy to perform encryption. For these reasons, DGVH is not the proper cryptosystem for WSNs. Without going into the details of the CMT cryptosystem, it is not adequate for a network using a thousand wireless sensors because the base station shares a unique secret key with each node in the network which makes the aggregation process of ciphertexts more complicated to implement.

In Domingo-Ferrer's cryptosystem, the size of parameter d affects the size of ciphertext (Newman, 2018). Domingo-Ferrer's encryption algorithm is a symmetric-key based cryptosystem uses two secret parameters r_p and r_q for encryption and computes r_p^{-1} and r_q^{-1} for corresponding decryption. Drastically reducing the size of the parameter *d* could attenuate the effect of exponentiation for not having very wide cryptogram. Although the cryptosystem is symmetrical, we could use an asymmetric algorithm (EC-ElGamal for example) to exchange the secret key globally throughout the WSN:

- This encryption algorithm begins with the choice of two prime numbers *p* and *q* and from them we could easily calculate $n = p * q$. These parameters (*p*, *q* and *n*) are only available on sensor nodes and end users. The parameter *d* must be greater or equal to 2.
- Then choose randomly r_p and r_q belonging to the multiplicative subgroup Z_p* and Z_q* respectively.
- Before the encryption process, it is necessary to split randomly the message *m* into different integers m_1, m_2, ..., m_d such that $\sum_{i=1}^{d} m_i$ = *m* mod *n* and $m_i \in Z_n$. The ciphertext is obtained by computing: $E_k(m) = ([m_1 * r_p^1 \; mod\, p,\; m_1 * r_q^1 \; mod\, q],\; [m_2 * r_p^2 \; mod\, p,\; m_2 * r_q^2 \; mod\, q],\; ...,\; [m_d * r_p^d \; mod\, p,\; m_d * r_q^d \; mod\, q])$
- To decrypt, compute the scalar product of the i^{th} [*mod p, mod q*] pair by [r_p^{-i} *mod p*, r_q^{-i} *mod q*] to retrieve the [m_i *mod p*, m_i *mod q*]. Add up to get [*m mod p*, *m mod q*]. Use the Chinese remainder theorem (Ireland and Rosen, 1990) to obtain the unique *m mod n*

The Fig. 3 shows an unrealistically small example of a Domingo-Ferrer cryptosystem over an integer. This cryptosystem supports homomorphic processing when the ciphertext is obtained using the same key. The size of the parameter *d* directly affects the size of the cryptogram and each plaintext is divided into *d* sub-plaintexts so that each of them is encrypted using the secret parameters *p*, *q*, r_p and r_q.

Domingo-Ferrer Implementation

The implementation is performed on a 16-bit microprocessor-based sensor node (i.e. the highest digit can handle is 2^{16} -1= 65535) called Waspmote and manufactured by Libellium. For this implementation we take the values of the ambient temperature (in Celsius degrees) assuming they are positive numbers in the range [0, 55]. So the choice of encryption parameters will be based on this interval so that *n* is greater than 55. Every 15 minutes, each sensor node wakes up and collects the temperature as a float number and converts (because this cryptosystem is applicable to the integer numbers but in order to produce accurate results, instead of converting the floats values of temperature to the integers, we could multiply each value by 100 for example but this would lead to excessive calculations) it to an integer to be encrypted using the previously programmed Domingo-Ferrer algorithm. The programming of the Waspmote nodes uses C++ language and code is loaded on the sensor node via the USB cable.

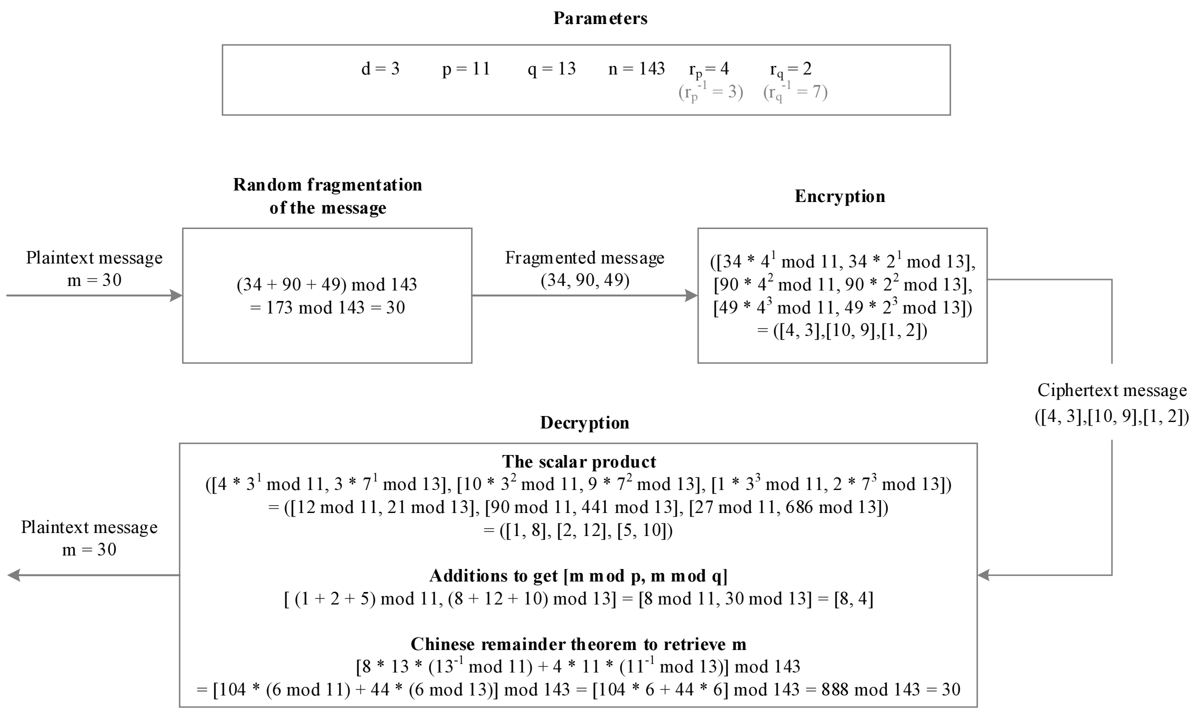
Parameters
d = 3 p = 11 q = 13 n = 143 r_p = 4 r_q = 2
(r_p^-1 = 3) (r_q^-1 = 7)
Random fragmentation of the message
Encryption
Plaintext message m = 30
(34 + 90 + 49) mod 143 = 173 mod 143 = 30
Fragmented message (34, 90, 49)
([34 * 4^1 mod 11, 34 * 2^1 mod 13], [90 * 4^2 mod 11, 90 * 2^2 mod 13], [49 * 4^3 mod 11, 49 * 2^3 mod 13]) = ([4, 3],[10, 9],[1, 2])
Ciphertext message ([4, 3],[10, 9],[1, 2])
Decryption
The scalar product
([4 * 3^1 mod 11, 3 * 7^1 mod 13], [10 * 3^2 mod 11, 9 * 7^2 mod 13], [1 * 3^3 mod 11, 2 * 7^3 mod 13])
= ([12 mod 11, 21 mod 13], [90 mod 11, 441 mod 13], [27 mod 11, 686 mod 13])
= ([1, 8], [2, 12], [5, 10])
Plaintext message m = 30
Additions to get [m mod p, m mod q]
[(1 + 2 + 5) mod 11, (8 + 12 + 10) mod 13] = [8 mod 11, 30 mod 13] = [8, 4]
Chinese remainder theorem to retrieve m
[8 * 13 * (13^-1 mod 11) + 4 * 11 * (11^-1 mod 13)] mod 143
= [104 * (6 mod 11) + 44 * (6 mod 13)] mod 143 = [104 * 6 + 44 * 6] mod 143 = 888 mod 143 = 30

Fig. 3: Encryption and decryption using Domingo-Ferrer's cryptosystem

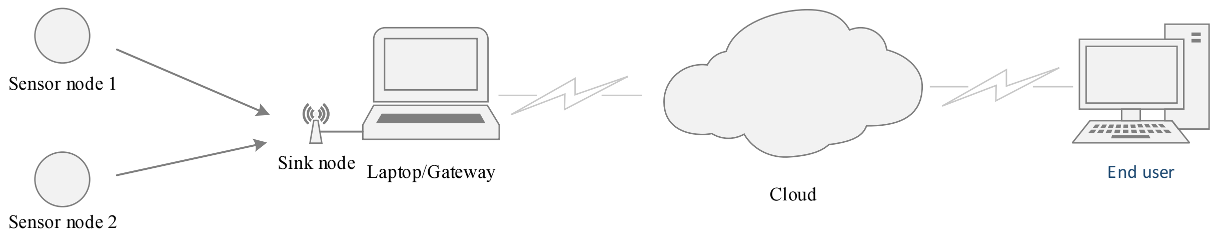

Fig. 4: Implementation configuration

After encryption, the sensor node creates a ZigBee frame and sends the temperature as ciphertext to the MAC address of the base station. For this implementation we do not include clustering due to lack of sufficient sensor nodes and the data are sensed every 10 sec in order to accelerate the experiments. Through the sink node (connected to a laptop acting as a gateway toward the cloud), the received ciphertexts are parsed and then routed to the cloud for storage. The complete configuration is shown in Fig. 4.

End users can access data stored in the cloud at any time, data operations must be performed in the cloud (Amazon EC2) and the user only gets results. Imagine we want to calculate the average annual temperature, as sensor nodes send data every 15 min, which is four values in one hour and 96 values each day. In a year, each sensor node collects 35040 temperature values. Suppose that the WSN is composed by 100 sensor nodes, we would have 3504000 values and it would be difficult to manipulate them with end-user computers having not enough power to compute. In the next Table 2 are the sample of sensed data and we will prove the homomorphism on them (the parameters used for this sampling are: $d = 2$, $p = 17$, $q = 13$, $n = 221$, $r_p = 6$ and $r_q = 9$).

To demonstrate the homomorphism, just decrypt the sum as ciphertext ([11, 6],[7, 7]) to verify whether we get the value 198 or not. So, let's start finding r_p^{-1} and r_q^{-1} in Z_p and Z_q respectively:

$$r_p * r_p^{-1} \ mod\ p = 1 \ and\ r_q * r_q^{-1} \ mod\ q = 1$$
$$6 * r_p^{-1} \ mod\ 17 = 1 \ and\ 9 * r_q^{-1} \ mod\ 13 = 1$$
$$\rightarrow r_p^{-1} = 3 \ and\ r_q^{-1} = 3$$

Let's compute the scalar product of the i^{th} [$mod\ p$, $mod\ q$] pair by [$r_p^{-i}\ mod\ p$, $r_q^{-i}\ mod\ q$] to retrieve [$m_i\ mod\ p$, $m_i\ mod\ q$]:

$$([11 * r_p^{-1}\ mod\ p, 6 * r_q^{-1}\ mod\ q],[7 * r_p^{-2}\ mod\ p, 7 * r_q^{-2}\ mod\ q]) = ([11 * 3\ mod\ 17, 6 * 3\ mod\ 13],[7 * 3^2\ mod\ 17, 7 * 3^2\ mod\ 13]) = ([16\ mod\ 17, 5\ mod\ 13],[12\ mod\ 17, 11\ mod\ 13])$$

Let's add [$m_i\ mod\ p$, $m_i\ mod\ q$] to obtain [$m\ mod\ p$, $m\ mod\ q$] = [11 mod 17,3 mod 13].

Table 2: Sample of encrypted data received after Domingo-Ferrer's Implementation

	Enrypted temperature	Decrypted temperature
	([12,12],[9,1])	32
	([9,6],[12,6])	33
	([1,9],[11,8])	34
	([9,5],[14,5])	34
	([2,12],[3,4])	33
	([12,1],[9,9])	32
Total	([11, 6],[7, 7])	198

Finally, we have to apply the Chinese Remainder Theorem to obtain the unique $m\ mod\ n$.

Assume that $z_1 = n/p = q$ and $z_2 = n/q = p$ and let $y_1 = z_1^{-1}\ mod\ p = 4\ mod\ 17$ and $y_2 = z_2^{-2}\ mod\ q = 10\ mod\ 13$

Thus, $m = (m_1y_1z_1 + m_2y_2z_2)\ mod\ n = (11 * 4 * 13 + 3 * 10 * 17)\ mod\ 221 = 198$. Hence, we proved the homomorphism on temperature values using the Domingo-Ferrer algorithm.

To understand how data is protected with this encryption process, we must first know that the encryption keys are known only by the sensor nodes and the decryption by the end user. If an attacker gains access to the encrypted data from the Table 2 stored in the cloud, he/she will not be able to do anything with it because the decryption keys are only available to the end user. Even when the end user wants to perform some arithmetic operations on the data of Table 2 such as the temperature average for example, the cloud calculates their sum by being encrypted and he/she receives only the result ([11, 6],[7, 7]) and the number of lines participated into the addition. The end-user decrypts ([11, 6],[7, 7]) locally on his/her computer using his decryption keys to get 198 and divides it by the number of summed values to find 33. In no case does the user send the decryption keys over the network (cloud) and this is why the attacker cannot intercept the decryption keys.

Domingo-Ferrer Performance Analysis

With a 16-bit processor we have to carefully choose the parameters to encrypt the data from the WSN. First of all, the choice of p and q must be such that the maximum value to be encrypted must be strictly less than n because by splitting the message m into m_1, m_2, …, m_d, all these short messages must be less than n also. During the implementation we used $n = 221$ with the temperature values included in the interval [0, 55], this assures us that this condition is sufficiently verified. Second condition is related to the randomly chosen parameters r_p and r_q, these two belong respectively in Z_p and Z_q that is why they can take any value but realistically small so that the r_p^d and r_q^d (also r_p^{-d} and r_q^{-d}) are not very big to saturate the maximum bits capacity of the processor. The third criterion is the choice of the parameter d, the more it is bigger the more we have several splits of message, so the r_p^d and r_q^d (also r_p^{-d} and r_q^{-d}) becomes relatively large which can lead to the bad results if during the encryption the calculations include the intermediate results exceeding the 16 bits (the maximum bits for the sensor node's microprocessor).

To evaluate the effects of Domingo-Ferrer encryption on the life state of the sensor node's battery, we carried out four experiments: the first is about to send 10,000 messages (temperature and battery level) in plaintext (Fig. 5a), the second experiment (Fig. 5b) is about to send the same number of messages but being encrypted (whose encryption parameters are: $p = 17$, $q = 13$, $n = 221$, $r_p = 3$ and $r_q = 2$) with the parameter d equal to 2, the third experiment (Fig. 5c) repeats experiment 2 but changing $d = 3$ and the fourth (Fig. 5d) is identical to the last three but with 4 fragments of plaintext ($d = 4$). The experiment configurations and parameters are summarized in the following Table 3.

Table 3: Experiment configuration and range of varying parameter values

Implementation	Sent messages	p	q	n	r_p	r_q	d
(a) No cryptography	10,000	-	-	-	-	-	-
(b) Encryption with 2 fragments	10,000	17	13	221	3	2	2
(c) Encryption with 3 fragments	10,000	17	13	221	3	2	3
(d) Encryption with 4 fragments	10,000	17	13	221	3	2	4

Table 4: Summary of the experiment results

Implementation	Sent messages	Battery before experiment	Battery after experiment	Dropped %
(a) No cryptography	10,000	39%	38%	1%
(b) Encryption with 2 fragments	10,000	68%	62%	6%
(c) Encryption with 3 fragments	10,000	82%	75%	7%
(d) Encryption with 4 fragments	10,000	90%	82%	8%

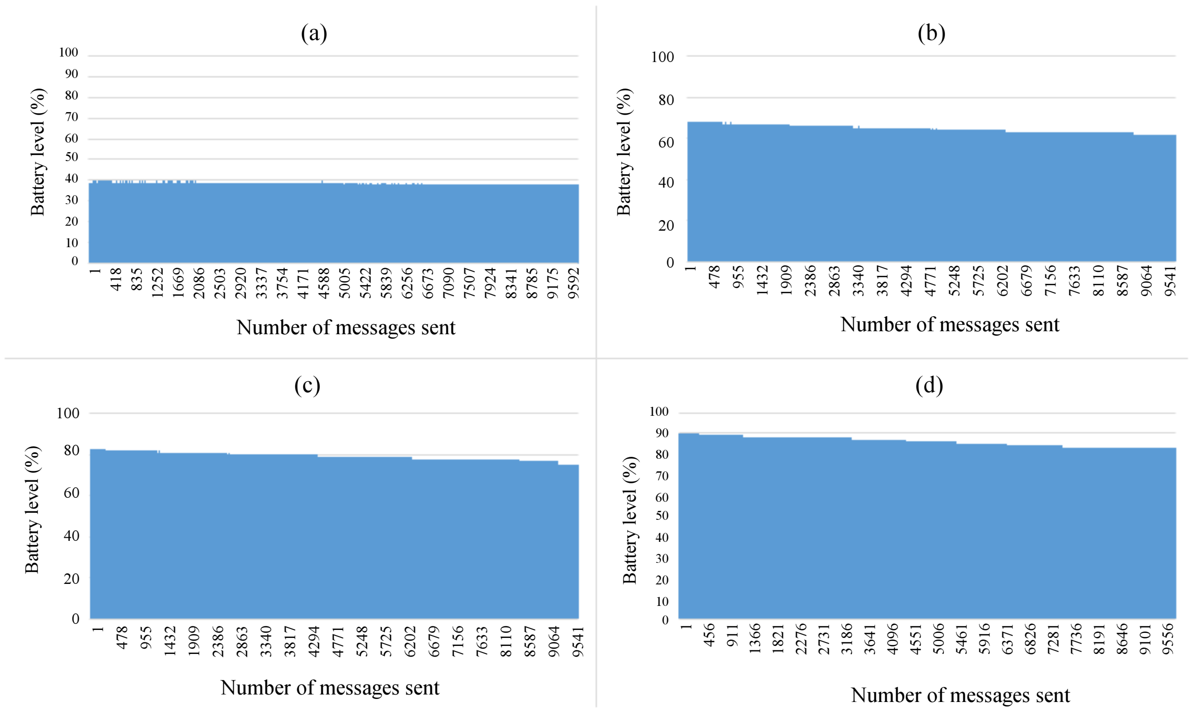

Fig. 5: Impact of domingo-ferrer homomorphic encryption on the state of the battery life

Note in Fig. 5a when the 10,000 messages are sent as plaintext the battery level of 39% drops to 38%, so the sensor node loses one percent which is relatively convincing for autonomous sensor nodes. When applying the encryption to the data collected with the parameter $d = 2$ we notice that at the end of the sending of 10,000 messages the level of the battery has dropped 6% (from 68% to 62% by referring to the Fig. 5b). By modifying the parameter $d = 3$, we notice a slight difference with the result obtained with $d = 2$ because in Fig. 5c we had the battery level which varied from 82% to 75%, i.e., 7% in this case and 1% more using $d = 2$. In Fig. 5d, the battery has dropped from 90% to 82% with $d = 4$, so undoubtedly we can comment that when increasing the number of fragments of the plaintext is slightly decreasing the whole energy of the wireless sensor network (Find the summary of the obtained results in the Table 4).

Comparing with the results obtained by some of the other related works (Ertaul and Yang, 2008), we can validate our methodology because we come with the same ascertainment that even if several message splits offer better security they have a detrimental impact on the whole network's energy, thus the number of fragments of the message should be kept at minimum.

Conclusion

Homomorphic encryption is useful in some applications of wireless sensor networks, its use has a positive impact firstly by promoting the confidentiality of the aggregated data and then allowing it to be stored safely into the cloud while performing arithmetic operations on the encrypted data. The use of a symmetric encryption algorithm such as the Domingo-Ferrer cryptographic system is implementable in the WSNs but it is first necessary to know the specifications of the sensor nodes in order to choose the best encryption parameters in order to avoid depletion of the network energy or exceeding the value limits that processors of the sensor nodes can support. Mutual encryption for WSN and cloud computing seems to be an essential solution for two types of networks, which one is WSN and the other is the Internet. Despite the advantages of these homomorphic encryption algorithms, they also have disadvantages to wireless sensor networks, the encryption process requires power calculations that consume a lot of energy, which makes the network vulnerable to longevity. The use of these algorithms lies between the importance of the project to be protected and the profitability of the network.

In this paper we have proposed a solution to secure the aggregated data by using Domingo-Ferrer's homomorphic encryption scheme with the objective of understanding its impact on the life of the WSN. From the obtained results we find that although more message splits offer a higher level of security, generally in terms of performance the energy of the wireless sensor network is penalized. Therefore, message splitting should be kept to minimum. To maintain the balance between the expected services and the security level, it would be better to choose the encryption parameters on the basis of the desired performances and the energy state of the sensor nodes. Thereafter, we will work on the arithmetic operations that Domingo-Ferrer homomorphic encryption is able to ensure.

Acknowledgement

I would like to thank my supervisor who guided me for this work to finish and Professor **Aziz GHAZI** for his technical support for the wireless sensor network platform.

Author's Contributions

Christophe Ishimwe Ngabo: He proposed this idea of using common security in the WSN integrated with cloud infrastructure. He was able to implement and evaluate the performance of the proposed system.

Omar El Beqqali: Supervised this work and helped to improve the layout. He found the collaborators to work together on the technical support of the sensor nodes.

Ethics

On behalf of my co-author, we inform you that this article has no ethical issues, we confirm that it is an original work which hasn't been published elsewhere. Each author has personally and actively reads this work before submission and that in no case will there be an ethical issues.

References

Ahmed, K. and M. Gregory, 2011. Integrating wireless sensor networks with cloud computing. Proceedings of the 7th International Conference on Mobile Ad-hoc and Sensor Networks, Dec. 16-18, IEEE Xplore Press, Beijing, China, pp: 364-366. DOI: 10.1109/MSN.2011.86

Buyya, R., J. Broberg and A. Goscinski, 2011. Cloud Computing: Principles and Paradigms. 1st Edn., Wiley, ISBN-10: 0470887990, pp: 664.

Castelluccia, C., E. Mykletun and G. Tsudik, 2005. Efficient aggregation of encrypted data in wireless sensor networks. Proceedings of the 2nd Annual International Conference on Mobile and Ubiquitous Systems -Networking and Services, Jul. 17-21, IEEE Xplore Press, San Diego, CA, USA, pp: 109-117. DOI: 10.1109/MOBIQUITOUS.2005.25

Dijk, M.V., C. Gentry, S. Halevi and V. Vaikuntanathan, 2010. Fully homomorphic encryption over the integers. Proceedings of the 29th Annual International Conference on Theory and Applications of Cryptographic Techniques, May 30-Jun. 03, Springer, French Riviera, France, pp: 24-43. DOI: 10.1007/978-3-642-13190-5_2

Domingo-Ferrer, J., 2002. A provably secure additive and multiplicative privacy homomorphism. Proceedings of the 5th International Conference on Information Security, Sept. 30-Oct. 02, Springer, Berlin, pp: 471-483. DOI: 10.1007/3-540-45811-5_37

Elhoseny, M., H. Elminir, A. Riad and X. Yuan, 2016. A secure data routing schema for WSN using Elliptic Curve Cryptography and homomorphic encryption. J. King Saud Univ. Comput. Inform. Sci., 28: 262-275. DOI: 10.1016/j.jksuci.2015.11.001

Ertaul, L. and J.H. Yang, 2008. Implementation of domingo ferrer's a new privacy homomorphism (DF a New PH) in Securing Wireless Sensor Networks (WSN). Security Manage.

Gentry, C., 2009. A fully homomorphic encryption scheme. PhD Thesis, Stanford.

Goldwasser, S. and S. Micali, 1984. Probabilistic encryption. J. Comput. Syst. Sci., 28: 270-299. DOI: 10.1016/0022-0000(84)90070-9

Holohan, E. and M. Schukat, 2010. Authentication using virtual certificate authorities: A new security paradigm for wireless sensor networks. Proceedings of the 9th IEEE International Symposium on Network Computing and Applications, Jul. 15-17, IEEE Xplore Press, Cambridge, MA, USA, pp: 92-99. DOI: 10.1109/NCA.2010.19

Ireland, K. and M. Rosen, 1990. A Classical Introduction to Modern Number Theory. 1st Edn., Springer, ISBN-10: 354097329X, pp: 389.

Johnson, M., M. Healy, P. van De Ven, M.J. Hayes and J. Nelson *et al*., 2009. A comparative review of wireless sensor network mote technologies. Proceedings of the IEEE SENSORS, Oct. 25-28, IEEE Xplore Press, Christchurch, New Zealand, pp: 1439-1442. DOI: 10.1109/ICSENS.2009.5398442

Benaloh, J., 1994. Dense probabilistic encryption. Proceedings of the Workshop on Selected Areas of Cryptography, (SAC' 94), pp: 120-128.

Koblitz, N., 1987. Elliptic curve cryptosystems. Math. Comput., 48: 203-203. DOI: 10.1090/S0025-5718-1987-0866109-5

Malik, M.Y., 2011. Efficient implementation of elliptic curve cryptography using low-power digital signal processor. Proceedings of the 12th International Conference on Advanced Communication Technology, Feb. 7-10, IEEE Xplore Press, Phoenix Park, South Korea, pp: 1464-1468.

Naccache, D. and J. Stern, 1998. A new public key cryptosystem based on higher residues. Proceedings of the 5th ACM Conference on Computer and Communications Security, Nov. 02-05, San Francisco, California, USA, pp: 59-66. DOI: 10.1145/288090.288106

Newman, C.R.E., 2018. Computer and Network Security Essentials. 1st Edn., Springer International Publishing, Cham.

Ngabo, C.I. and O.E.L. Beqqali, 2016. Real-time lighting poles monitoring by using wireless sensor networks applied to the smart cities. Proceedings of the International Conference on Big Data and Advanced Wireless Technologies, Nov. 10-11, ACM, Blagoevgrad, Bulgaria. DOI: 10.1145/3010089.3010097

Okamoto, T. and S. Uchiyama, 1998. A new public-key cryptosystem as secure as factoring. Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques, May 31-Jun. 4, Espoo, Finland, pp: 308-318. DOI: 10.1007/BFb0054135

Paillier, P., 1999. Public-key cryptosystems based on composite degree residuosity classes. Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques, May 02-06, Springer, Prague, Czech Republic, pp: 223-238. DOI: 10.1007/3-540-48910-X_16

Potey, M.M., C.A. Dhote and D.H. Sharma, 2018. Low-Size Cipher Text Homomorphic Encryption Scheme for Cloud Data. In: Networking Communication and Data Knowledge Engineering, Perez, G.M., K.K. Mishra, S. Tiwari and M.C. Trivedi (Eds.), Springer Singapore, ISBN-10: 981104600X, pp: 93-102.

Rabah, K., 2005. Theory and implementation of elliptic curve cryptography. J. Applied Sci., 5: 604-633. DOI: 10.3923/jas.2005.604.633

Rivest, R.L., A. Shamir and L. Adleman, 1978. A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM, 21: 120-126. DOI: 10.1145/359340.359342

Roy, S., M. Conti, S. Setia and S. Jajodia, 2012. Secure data aggregation in wireless sensor networks. IEEE Trans. Inform. Forens. Security, 7: 1040-1052. DOI: 10.1109/TIFS.2012.2189568

Sohraby, K., D. Minoli and T. Znati, 2007. Wireless Sensor Networks: Technology, Protocols and Applications. 1st Edn., John Wiley and Sons, ISBN-10: 0470112751, pp: 328.

Tebaa, M., S. El Hajji, A. El Ghazi, S. El Hajji and A. El Ghazi, 2012. Homomorphic encryption applied to the cloud computing security. Proceedings of the World Congress on Engineering, Jul. 4-6, Newswood Limited, London, UK, pp: 536-539.

Ugus, O., D. Westhoff, R. Laue, A. Shoufan and S.A. Huss, 2009. Optimized implementation of elliptic curve based additive homomorphic encryption for wireless sensor networks. Arxiv Preprint ArXiv09033900.

Wang, L., L. Wang, Y. Pan, Z. Zhang and Y. Yang, 2011. Discrete logarithm based additively homomorphic encryption and secure data aggregation. Inform. Sci., 181: 3308-3322. DOI: 10.1016/j.ins.2011.04.002

Xie, K., X. Ning, X. Wang, S. He and Z. Ning *et al*., 2017. An efficient privacy-preserving compressive data gathering scheme in WSNs. Inform. Sci., 390: 82-94. DOI: 10.1016/j.ins.2016.12.050

Zhang, P., J. Wang, K. Guo, F. Wu and G. Min, 2018a. Multi-functional secure data aggregation schemes for WSNs. Ad Hoc Net., 69: 86-99. DOI: 10.1016/j.adhoc.2017.11.004

Zhang, P., S. Wang, K. Guo and J. Wang, 2018b. A secure data collection scheme based on compressive sensing in wireless sensor networks. Ad Hoc Net., 70: 73-84. DOI: 10.1016/j.adhoc.2017.11.011

Zheng, J. and A. Jamalipour, 2009. Wireless Sensor Networks: A Networking Perspective. 1st Edn., Wiley-IEEE Press, ISBN-10: 0470167637, pp: 520.