Securing a wireless sensor network for human tracking: a review of solutions

Alejandro Carrasco[1,*,†], Francisco Alcaraz[1], Julio Barbancho[1], Diego F. Larios[1] and Jose Luis Sevillano[2]

[1]*Departamento de Tecnología Electrónica, Universidad de Sevilla, Seville, Spain*
[2]*Departamento de Arquitectura de Computadores, Universidad de Sevilla, Seville, Spain*

SUMMARY

Currently, wireless sensor networks (WSNs) are formed by devices with limited resources and limited power energy availability. Thanks to their cost effectiveness, flexibility, and ease of deployment, wireless sensor networks have been applied to many scenarios such as industrial, civil, and military applications. For many applications, security is a primary issue, but this produces an extra energy cost. Thus, in real applications, a trade-off is required between the security level and energy consumption. This paper evaluates different security schemes applied to human tracking applications, based on a real-case scenario.

KEY WORDS: wireless sensor networks; security; tracking, localization

1. INTRODUCTION

Wireless sensor networks (WSNs) consist of hundreds or thousands of small and autonomous devices that cooperate wirelessly among themselves to monitor or control an area. WSN devices are based on a small microcontroller, a power supply system, and a low power radio transceiver (Figure 1). They are generally small and low cost, and therefore, they present several limitations such as

- Limited energy availability. In some applications, especially those for mobile and portable devices, a maximum weight and size are imposed that does not permit the use of a large rechargeable battery system.
- Limited computing power, for obtaining high lifetimes.
- Limited bandwidth and coverage, to keep power consumption low.

Despite these issues, the use of WSNs keeps growing. Numerous applications, such as monitoring environmental scenarios, health applications, or survivor help in post-disaster scenarios, are areas where WSNs are used successfully [1]. These networks create intelligent services to make living environments more comfortable and safer with the networked interconnection of everyday objects, in what is known as Internet of Things [2].

But for some applications, WSN manages confidential information. This requires the use of mechanisms to send the information securely over the network routes. Therefore, security is an important issue with WSNs [3].

*Correspondence to: Alejandro Carrasco, Departamento de Tecnología Electrónica, Universidad de Sevilla, Seville, Spain.

†E-mail: acarrasco@us.es

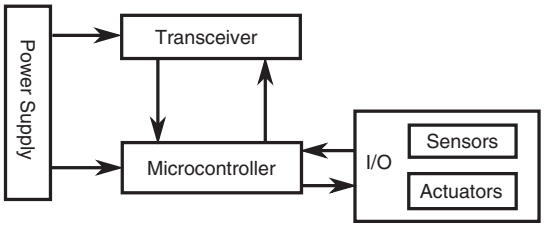


Figure 1. Architecture of a wireless sensor network node.

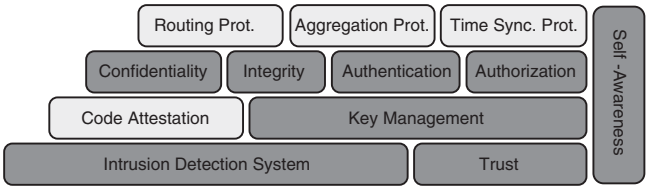


Figure 2. Security integral service.

Network security has been studied widely in literature, especially for military applications. However, due to hardware constraints, the implementation of security protocols over WSNs is a nontrivial problem [4]. As a consequence, the study and development of secure protocols for WSNs continue to pose a challenge for the scientific community. In-depth, studies have recently been conducted into [5–7] large infrastructure management and key administration.

Wireless sensor network security should be considered as an integral [8] and configurable [9] service based on a group of basic features selected according to the security requirements and hardware characteristics of the devices [10]. The most important features to be considered for preventing attacks on WSNs are depicted in Figure 2.

The commonest potential threats to WSNs include the following:

- Data modification: An attacker node tries to delete or replace part or all of eavesdropped information, causing a loss of the network's integrity.
- Impersonation of a node: An attacker node tries to supplant a real node from the network.
- Replaying: The attacker can eavesdrop on a piece of valid information and resend it to the network at another moment.
- Eavesdropping: An external device acts as a sniffer and tries to store confidential information.

Preventing these security breaches requires the use of complex cryptographic algorithms. But these algorithms add an extra energy cost to the device, regardless of how they are implemented. In hardware implementation, security requires additional devices that increase a device's power consumption. In software implementation, cryptographic algorithms consume CPU cycles, affecting particularly microcontrollers with low computational resources, such as those commonly used in WSN devices. These algorithms also add overheads to communications, due to the addition of security layers [11], that increase the power energy consumption in the radio transceiver.

Generally, as security levels increase, so does power energy consumption; this has been demonstrated by several studies [12]. Therefore, in a real application, there needs to be a trade-off between security levels and power consumption [13]. Maintaining low power consumption requires a careful study of the features for each application.

This paper evaluates the main security requirements in people tracking applications. On the basis of these requirements, we conducted a study and evaluation of different security mechanisms by searching for a trade-off between security requisites and power consumption. We performed some real bench mark testing on these security mechanisms, to evaluate their suitability in the proposed application.

The rest of the paper is organized as follows: Section 2 exposes the application of the WSN for tracking people; Section 3 highlights some of the potential threats that are exposed in WSN because

of its *modus operandi*. This brings us to the need for implementation of security measures, which are described in Section 4. At the same time, these measures have to comply with certain security properties, which are described in paragraph 3.1 of Section 3. Finally, Section 5 provides some conclusions and notes on future projects.

2. WIRELESS SENSOR NETWORK APPLIED TO TRACKING PEOPLE

A tracking system has many applications [14], such as detecting enemy vehicles or wildfires. This problem has been discussed widely in the literature, especially for outdoor applications using external sensors, such as GPS. But in some applications, such as indoor localization or long-term tracking systems, GPS or similar devices do not work properly or consume too much energy.

One example of these applications are people tracking systems on hiking trails, where there are no mobile connections, required for A-GPS systems to work. These localization systems have the following objectives:

- Provide users with information about their localization and the points of interest around them.
- Offer an automatic safety system to users. The system can detect emergencies (i.e., if a person has a fall in the park) and send an alarm to the base station.
- Supply the park manager with information about public affluence in different areas. This facilitates the control of access to restricted areas and the evaluation of the load in terms of visitors to the park.
- This portable system has to be both low weight and low cost, allowing the deployment of hundreds of devices.
- The battery life has to be relatively high (a minimum of 24 h).

In these situations, a WSN works well, offering a low cost and low power consumption solution. The proposed system consists of the following subsystems:

- **Anchor nodes:** located in fixed positions in the park. These devices are used to route the information to the base station for the localization algorithm.
- **Tags:** are the small devices users take with them.
- **Base station:** This is a special anchor node that acts by routing the WSN information from the network to a PC. This PC supplies the information acquired by the network to the other devices.

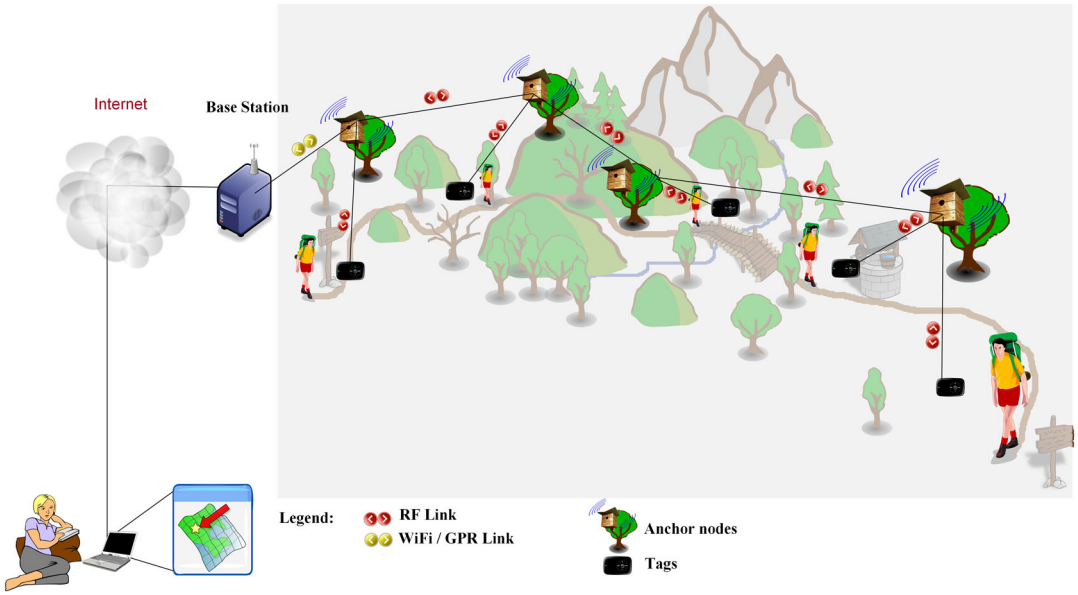


Figure 3. General scheme of a people tracking system.

Figure 3 shows the layout of the proposed system.

People tracking systems represent specific cases within the more general problem of localization. Localization is an important issue in WSNs because to correctly interpret a sensor's measurements, you have to know the position of each WSN device [15]. The literature includes many localization systems for WSNs [16], and they feature two categories of localization algorithms:

- **Range-based:** These techniques estimate, point-to-point, the distance between all the nodes using sensors such as ultrasound [17]. With this information, and by using techniques such as triangulation, the absolute position of the non-anchor nodes can be estimated. Generally, these techniques require additional hardware. The most common ones are received signal strength indication (RSSI) [18, 19], time of arrival [20], time difference of arrival [21], and angle of arrival [22].
- **Range-free:** In these techniques, the position of non-anchor nodes is obtained through implicit information provided by anchor nodes, usually based on exchanged messages, commonly called beacons. This information normally consists of different aspects, such as radio coverage membership or the number of hops between devices. The most common ones are centroid [23] and DV-Hop [24].

In general, the range-based techniques offer better accuracy, but additional hardware is often needed. As a result, the weight, cost, and power consumption of node devices increase, making these techniques unviable. RSSI range-based techniques are an exception because most of the current transceivers provide this measure by default. However, RSSI techniques are very sensitive to noise and interference.

It is important to point out that in WSNs power consumption is very high in transmission and also in reception modes. Because of this, to reduce the power consumption in tags, the number of messages exchanged needs to be reduced. All the node activity enabling low power modes also need to be stopped and the radio transceiver switched off. Therefore, a suitable activity manager is needed.

A localization algorithm can also be classified as centralized and distributed. Distributed algorithms execute the main part of the code in the tag. In contrast, in a centralized algorithm, all the information is sent to the base stations, which execute that main code. In general, regarding energy consumption, distributed algorithms are more efficient than centralized algorithms and are therefore generally better options for tracking applications [25], especially for people tracking [26].

Typical WSNs are sensitive to different kinds of attacks, as we shall see in Section 3. The proposed application requires a robust and secure localization technique. Security mechanisms are only considered by a small number of localization techniques [27, 28]. They normally use symmetric-key cipher for the encryption and authentication of beacons. They focus on the robustness of localization considering different types of communication attacks.

Other papers focus on the authentication of anchor nodes, and only consider the localization algorithm nodes without suspicious behavior. For example, [29] proposes a statistical approach to detect these malicious nodes, [30] trying to filter malicious nodes based on their location within the network. Generally, these algorithms do not consider security requirements for the transmitted information. This approach is therefore not suitable for people tracking applications.

Other authors propose security systems to prevent specific attacks, such as [31] preventing a whorm-hole attack on a DV-Hop, [32] avoiding collaborative collusion attacks, or [33] preventing selective forward attacks. None of them take the integrity of tag messages into account.

In general, secure localization techniques do not take power consumption into account [34]. An increase in security normally causes an increase in power consumption because of the extra overhead.

Given this situation, this paper proposes the use of a low power consumption localization technique (at the application level) for tracking people. This people tracking system is based on LIS ç[34], a localization range-free technique focused on reducing power consumption. Different security mechanisms to be applied to the localization algorithm are evaluated, measuring the over-cost energy that they produce in mobile devices. These measurements allow us to evaluate the best trade-off between security and power consumption for the proposed application.

3. SECURITY REQUIREMENTS, THREAT MODELS, AND ATTACKS

Typical WSN nodes are based on IEEE 802.15.4 [35] or *ZigBee* [36] radio transceivers. This specification offers security facilities in the medium access control (*MAC*) sublayer. These facilities can be harnessed by upper layers to achieve the desired level of security.

This section presents the major security services that are wanted in WSNs [37], in addition to the requirements that must be fulfilled by the security protocols [38]. The most important security services in WSNs are as follows:

- Confidentiality: The owner of the information needs assurance that the information will be kept secret from unauthorized entities. This must be achieved through encryption.
- Security semantics: An encryption scheme must be implemented to prevent an adversary from gleaning partial information from the encrypted data. This prevents, for example, differential cryptanalysis attacks. Due to this, two encrypted texts will give identical security levels despite the texts being encrypted differently. To do this, you typically use an initialization vector IV or $numberusedonce$ (nonce).
- Availability: The system must ensure its network services, even in the case of denial of a service attack.
- Authorization: The system must ensure that only authorized sensors can transmit or receive on WSNs.
- Authentication: Ensures that the communication among nodes is legitimate. That is to say that an adversary cannot impersonate a legitimate network node.
- Integrity: Messages sent among nodes should not be modified during retransmissions. This is achieved by appending a message authentication code (*MAC*) to the payload. The IEEE standard 802.15.4 refers to this code as *message integrity code (MIC)* to differentiate it from the link layer of the Open Systems Interconnect (OSI) stack.

Regarding its requirements, WSN security protocols must fulfill the following conditions:

- Scalability: Considering that in a WSN, the number of nodes can vary from a few nodes to thousands dynamically, the key management of the security protocols must deal with this problem dynamically.
- Freshness of data: This has three implications: firstly, the adversary that is listening to the communication channel should not be able to replay old messages. This is known as *replay protection*; the second implication is weak freshness. This requires a partial order of the messages, but does not consider delays in information; thirdly, strong freshness provides a total order of the *message couples request–response* and allows for delays in the information. Synchronization of the network requires strong freshness.
- Robustness against attacks: If an attack has been carried out, the security protocols must minimize the impact over the network. They should also detect faults and act to adapt the network topology to prevent them.
- Authentication broadcasts: Commonly, a base station sends data and commands to the whole network (broadcast messages). An adversary might modify these commands and make the nodes perform incorrect operations. Therefore, the security protocols must provide authentication in the broadcast messages.
- Auto-organization: Typically, WSNs are ad-hoc networks without a fixed infrastructure. Nodes must have self-organization and self-healing capacity to create multihop communications routes. Due to this, the security protocols must have an efficient policy of key management.

3.1. Evaluation of security requisites in wireless sensor networks applied to people tracking

Security requisites vary as a function of the application. In this case, we focus on a distributed people tracking system based on LIS localization Algorithm [34]. The main steps of this localization algorithm can be summarized as follows:

S1: Anchor nodes wait for non-anchor node beacons.
S2: Non-anchor broadcasts a beacon.
S3: Some anchor nodes receive the beacon. The localization is obtained based on radio message transmission between these anchor nodes that surround the non-anchor node, executing LIS algorithm. Each partial localization estimation from non-anchor nodes is sent to a Cluster-Head.
S4: Cluster-Head obtains final non-anchor position estimation. This information is sent to the non-anchor node.
S5: After receiving its localization, the non-anchor node waits a certain period before repeating the cycle.

As it can be seen, in this algorithm, the localization estimation is obtained between the anchor nodes that surround a non-anchor node. It reduces considerably the amount of information exchange through the network and therefore the amount of information that a passive attacker node can collect.

Moreover, because in the proposed application only non-anchor nodes require knowing their own position, an attacker node can only collect messages when a non-anchor node pass near to its coverage area. Most of the exchange information contains only partial results. Only the message exchanged between Cluster-Head and non-anchor node contains the position estimation and the node identification. This makes cryptanalysis attacks much more difficult.

For these reasons, the messages can be classified in different categories, with different security requisites: beacon from non-anchor node to anchor nodes, messages between anchor nodes, and messages from anchor nodes to non-anchor nodes. These messages require the following security services:

- Confidentiality: Non-anchor node does not send any sensitive information, just a beacon. Thus, it does not require a confidentiality service. The messages from anchor nodes manage private information, requiring confidentiality.
- Security semantics: To avoid cryptanalysis attacks, all the messages of the networks require this service.
- Availability: Beacons from non-anchor node and the message with the localization non-anchor nodes do not send ACK. For this reason, these messages are not sensitive to denial-of-service *DoS* attack. Messages between anchor nodes require availability services.
- Authorization: Apart from beacons from anchor nodes, all the other messages require this service.
- Authentication: If an attacking node tries to override a non-anchor node, it is not going to receive sensitive information from the non-anchor node. In any case, the network can respond with its own localization, not with the localization of other devices. For this reason, beacons do not require authentication, but this is required for the other devices.
- Integrity: Only messages between anchor nodes can have multiple hops. For this reason, only these messages require this security service.

Table I summarizes these security requisites. As can be seen, messages between anchor nodes require high security services. On the other hand, beacons from non-anchor nodes require low levels of security, making it possible to improve battery lifetime in these devices.

Table I. Security requisites.

Service	Tag → Anchor	Anchor → Anchor	Anchor → Tag
Confidentiality	No	Yes	Yes
Security semantics	Yes	Yes	Yes
Availability	No	Yes	No
Authorization	No	Yes	Yes
Authentication	No	Yes	Yes
Integrity	No	Yes	No

Communications between anchor nodes must fulfill the requisites of scalability, freshness of data, authentication, and auto-organization, because the localization algorithm is executed mainly over these devices.

Communication to or from non-anchor nodes does not require these services, because its communication does not expect ACK, and beacons are broadcast messages, not requiring information about network topology.

Using this information, the security issues can be adapted to this application, looking for a trade-off between system reliability and power consumption.

3.2. Attacks on wireless sensor network

Several authors have proposed different classifications of threats and attacks on WSNs [38–40]. The most extended classifications are the following:

- 'Mote attacks' versus 'laptop-class attacks': In the first case, an adversary uses nodes with similar capabilities to the network devices. In contrast, 'laptop-class attacks' use devices with more resources than network nodes. Using hardware with more resources permits complex attacks and complex cryptanalysis [41]. For example, an opponent using a mote may only cause interference in its surroundings; however, with 'laptop-class' hardware, it could cause interference over wider areas and over the entire network. This is commonly because 'laptop-class' devices have less energy consumption restrictions than motes. WSNs applied to human tracking are sensitive to these kinds of attacks, especially 'laptop-class attacks'.
- Attacks from external versus internal adversaries: In the first case, the adversary is not part of the network, and it does not have access to the network resources. Internal attacks at least imply that a legitimate node in the network is running malicious code and often has access to the cryptographic keys. WSNs applied to human tracking are sensitive to external adversaries. The nodes can be blocked to avoid extraction of information of devices and therefore avoid internal adversaries' attacks.
- Passive and active attacks: The first case involves a node that is monitoring or listening to the communication channel. Active attacks involve modification and injection of WSN network traffic. It is difficult for the network under attack to detect passive attacks. A typical solution to minimize these kinds of attacks is to use privacy services. If the attacker node modifies or injects false traffic, additional security services need to be used, such as integrity, authentication [42], and protection against replicas. The proposed application may be sensitive to these two kinds of attacks.

The most common type of attack over wireless networks is known as *DoS* [43]. *DoS* is an interruption of any service provided by a network or server. In WSNs [38, 39], however, we consider a *DoS* to be the loss, or degradation of some of the functionality of the WSN. We can classify these *DoS* according to the layers of the OSI stack, as we explain as follows.

3.2.1. Physical attack. His type of attack is also known as a catch from the node in the sense that the adversary physically has direct access. As the cost of the nodes has to be as low as possible, applying techniques and measures to prevent direct manipulation is costly and therefore not practical. These attacks have a significant impact on the routing information and the access control mechanisms in WSNs. For example, an adversary that obtains the keys stored in the sensor, will have full access to information. A physical attack can be executed without disrupting the activity of the sensor using the *JTAG* connection and gaining control of the microcontroller. Possible security methods include disabling the *JTAG* interface or password-protecting the boot loader. This prevents internal attacks over the physical layer.

3.2.2. Attacks on the Open Systems Interconnection layers: Physical layer. Jamming (i.e., attacking the network via interferences) is very common. This is an attack that an adversary can only cause by knowing the frequency at which the WSN is transmitting. A signal is sent randomly at the same frequency as the nodes, making this signal interfere with the original signal. Thus, the receivers

within range of the attacker cannot receive messages. Frequency hopping [44] can be used as a method to fight against this kind of attack. This is the method used in Bluetooth networks, but it is not the optimal solution in WSNs because it requires extra processing, and the number of bands is limited. A more effective technique [45], which makes a jamming attack almost impossible, is to use *ultra wide band*, based on the transmission during very short periods of the order of nanoseconds on the entire bandwidth. In fact, the 802.15.4 standard has included a *ultra wide band* version since 2007 [46].

3.2.3. Attacks on the Open Systems Interconnection layers: Link layer collision. The purpose of this type of attack is to exhaust the energy of the nodes. The adversary hears transmissions of a WSN and when it detects a message, it sends a signal over a short period. This makes the receiver detect an error in the *cyclic redundancy check (CRC)* and ask the sender to send the message again. If this is prolonged in time, it will cause a degradation of the batteries. In addition, this method is more effective from the adversary's point of view than technical *jamming* because it requires less energy to cause the *DOS* as the collision in a single byte is sufficient to cause error in the *CRC*. Also this attack is more difficult to detect than *jamming*. To avoid this attack, the same techniques used with jamming could be used.

3.2.4. Attacks in the Open Systems Interconnection layers: Network layer. According to [38], all network-level attacks can be classified into the following types:

- Selective forwarding (selective routing): In this type of attack, a malicious node can refuse to steer some packages, and even delete all packages, called a *blackhole* attack. The success of this attack depends on two factors: closeness of this malicious node to the base station and percentage of packets aimed at or deleted, which will allow the adversary to reserve energy.
- Sinkhole attack: In this case, the adversary captures or attracts traffic to a committed node. The closer to the base station it is, the more effective the attack. Also, committing a node could take place through the base station. Due to their nature, WSNs are very susceptible to this type of attack.
- Sybil attack or 'multiple personalities': A node can represent multiple identities to the other nodes in the WSN. This causes the memory of the routing tables of the neighboring nodes to be filled with garbage information.
- Flooding attack: A malicious node can cause a flood of junk messages over the entire network, for example forwarding *broadcast* messages. This can cause congestion in the WSN, resulting in a *DOS*.

4. EVALUATION

In this section, we present the results of a security evaluation of the WSN we are testing, which is based on IEEE 802.15.4 nodes using *TinyOS*. *TinyOS* is an open source operating system based on components, specially designed for WSN devices with few resources. It is written in *NesC* language [47], a dialect of *C* language optimized to take into account the restrictions of the hardware sensors. Sensors have been used in the motes of the platform *telos* [48]. Figure 4 shows the node that has been used for testing. This is the TelosB platform, and it has the following characteristics [48]: Microcontroller of Texas Instruments MSP430F16x 16-bit 8 MHz, with 10 KB of RAM and 48 KB of Flash, with integrated light sensors, temperature and humidity, USB, ADC converters, integrated antenna, radio via Chipcon CC2420 SmartRF [49] with the ability to perform encryption and decryption tasks using the 128-bit Advanced Encryption Standard (AES) algorithm that supports IEEE 802.15.4 [35], bandwidth of 250 kbps at 2.4 GHz.

To perform the evaluation, a localization application is implemented that uses different messages, with different payloads:

- *Beacons from non-anchor node*: These messages do not contain any payload.
- *Messages between anchor nodes*: 2 bytes of Payload: 1 byte with an ID of the non-anchor node and 1 byte with the received RSSI.

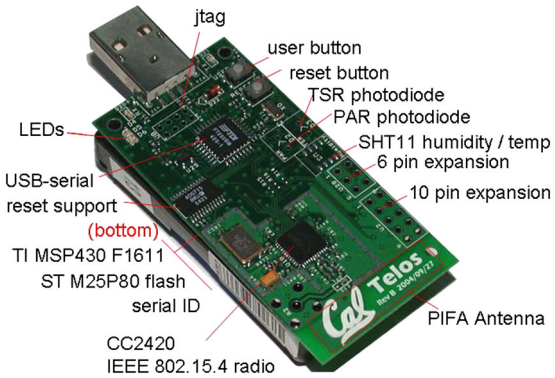


Figure 4. Mote TelosB.

- *Messages from anchor node to non-anchor node*: 2 bytes of Payload: 1 byte with X coordinate and 1 byte with Y coordinate.

In order to quantize the cost of implementing the security protocol, searching for a trade-off between security and energy consumption, we are going to analyze the data frame, using two sniffer softwares: *Perytons* [50] and *Texas SmartRF* [51].

4.1. Evaluation of IEEE 802.15.4 security primitives

The IEEE 802.15.4 standard provides security at the link layer, supported by hardware in most standard radio transceivers, such as the *CC2420* from *Texas Instruments* used in TelosB motes.

The algorithm used is the *AES* or *Rijndael*, which was announced in *Federal Information Processing Standards, FIPS 197,* by the *National Institute of Standards and Technology* in the year 2001. This is a symmetric encryption scheme of 128-bit blocks, with a number of iterations (10, 12, 14) that depends on the size of key 128, 192, or 256 bits.

The main advantage of using these security primitives is that, as defined in the standard, they are implemented via the radio transceiver. For this reason, the overload processing security (used for encrypt and decrypt messages) may be considered negligible (as an example, encrypting or decrypting 69 bytes using CCM mode-8 security primitives for a CC2420 radio transceiver, only requires around 99 μs of extra time with the radio in on state). In this sense, security primitives only increase the power consumption because of the increase in the size of headers.

According to the IEEE 802.15.4 standard, there are three modes of operation for encryption:

- *Cipher block chaining-message authentication code (CBC-MAC)*: This is used for integrity and authentication. At the end of the operation with *AES*, we obtain a summary or *hash* of the 128 bit message from both the header and the payload, which can be appended to the payload in 32, 64, or 128 bits, so we obtain the three variants of *CBC-MAC*.

  An initialization vector uses a zero, as defined in the *Request for Comments, (RFC 3602)*. Different *CBC-MAC* implementations can be found in the literature [52], considering different security evaluations.
- *Counter (CTR)*: Provides confidentiality to encrypt the payload. A CTR is used to frame part of the *nonce* semantics to provide security and protection against replicas, RFC 3686.
- *Counter with CBC-MAC (CCM-MAC)*: This represents the union of the previous versions; therefore, it provides the services of security confidentiality, integrity, and authentication. RFC 3610 has to process the algorithm twice, resulting in a higher cost. The first processing generates the MAC, and the second processing encrypts the payload and the computed MAC in the first stage. The initialization vector or nonce is the same for both stages. Depending on the number of bits attached to the payload for the MAC generated, we obtain three variants of 32, 64, or 128 bits.

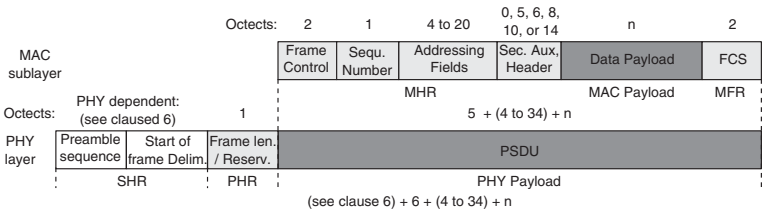


Figure 5. Frame format data 802.15.4.

```
/**
 * CC2420 Security Header
 */
typedef nx_struct security_header_t {
  nx_uint8_t secLevel:3;
  nx_uint8_t keyMode:2;
  nx_uint8_t reserved:3;
  nx_uint32_t frameCounter;
  nx_uint8_t keyID[1]; // One byte for now
} security_header_t;
```

Figure 6. Security header CC2420.

A small number of studies into the IEEE 802.15.4 standard security primitives have identified some of the most important lapses of security in these primitives [52]. Firstly, the use of encryption in application payloads only is strongly discouraged, as it presents vulnerabilities and incurs a false sense of security. We might provoke a *DoS* attack by assigning to the *frame control, key identifier values* `Oxffffffff, OxFF` the receiver will not accept more packages because we have reached the maximum. Secondly, we should not rely on the recognition packages (ACK), which will be sent without authentication and integrity due to the associated dangers. Thirdly, a series of vulnerabilities due to the use of the ACLs for key management has been identified (IEEE 802.15.4 year 2003) [52].

However, the 2006 revision of the standard, according to the CC2420 datasheet, delegates the policy of management and maintenance of the key to the upper layers of the protocol. With respect to the replay service protection, the standard of 2006 indicates that it can be performed in the value field of the frame control, which must be unique under the same key in modes *AES-CTR and AES-CCM*. However if we have a network of n shared key nodes, each node would send $2^{32}/n$ packages; therefore, it is understood that the *replay protection* service is supported properly when the communication is between pairs of nodes, otherwise this service should be delegated to higher layers.

To achieve entirely safe communications in terms of confidentiality and integrity using IEEE 802.15.4 security primitives, there needs to be a way of periodically renewing the keys in the upper layer in the protocol, because symmetric encryption is vulnerable to cryptanalysis, if a sniffer acquires enough frames.

IEEE 802.15.4 data frame format is depicted in Figure 5. As can be seen, there is a field called *auxiliary security header*. This field will only be present if the security services have been enabled, which is indicated by one bit of the *frame control* field. This security header is subdivided into three fields [35]:

- *Security control*: Indicates the type of security service enabled (none, CBC-MAC, CTR, and CCM).
- *Frame control*: 4 bytes that are incremented each time a message is encrypted to provide *replay protection*.
- *Key identifier*: Provides the information required to select the key in the receiving node.

In our application, this header has a size of 6 bytes, as shown in Figure 6.

4.1.1. Evaluation without security primitives. Figure 7 shows the capture made by *Perytons* of a radio transmission without security primitives:

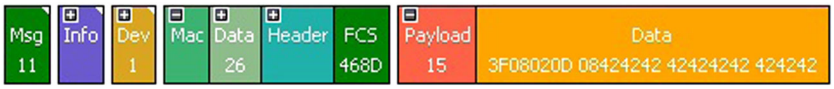


Figure 7. Capture with Perytons.

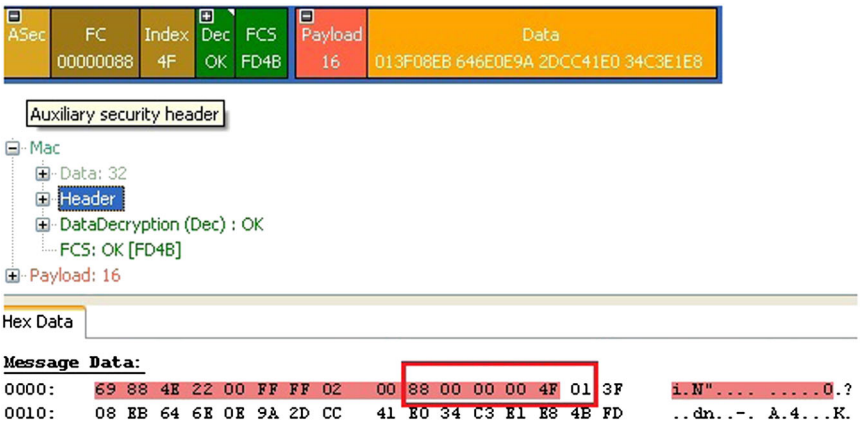


Figure 8. Capture with Perytons, CTR.

The IEEE 802.15.4 payload size is 15 bytes = 1 byte (compatibility 6LoWPAN) + 1 byte (AM type) + 1 byte (node id) + 2 bytes (message) + 10 bytes (padding).

The first two bytes (6LoWPAN and AM type) do not appear in the IEEE 802.15.4 standard. They are added to the payload from *TinyOS* to maintain the compatibility with IPv6 over Low power Wireless Personal Area Networks. For this reason, these bits can be considered as an application header.

Summarizing, using *TinyOS*, for an application without security, we have a header of 11 bytes (considering 6LoWPAN and AM type bytes) plus an application payload of 13 bytes plus 2 bytes (CRC), which gives a total of 26 bytes.

Standard emphTinyOS frames do not provide any security services. Therefore, it is not applicable to the people tracking application.

4.1.2. Evaluation with control security primitives. Figure 8 shows a capture of a frame between devices with CTR security. One can see that the security bit is active, and the total size of the frame is now 32 bytes. The first byte, with value 0×88, indicates the security level (level 4) and the key mode (1). Additionally, the system attaches to the message the index key field, with a value of 0×01 in this case.

As can be seen, there is an increase of the auxiliary security header, compared to a message without security primitives. With this security primitive, the header has been increased by 6 bytes.

This security mechanism introduces the lower overhead of the studied alternatives and satisfies the security requisites of the messages from non-anchor nodes to anchor nodes but not the requisites of the messages between anchor nodes or messages from anchor nodes to non-anchor nodes.

4.1.3. Evaluation with cipher block chaining-message authentication code security primitives. In this case, we are going to use the CBC mode-MAC-4 bytes. Figure 9 shows a captured frame. It shows that the payload is increased by 4 bytes, due to the addition of the hash of 4 bytes. The application payload is not encrypted and maintains 6 bytes of the security header.

In this case, the first header byte value is 0×28, representing a security level 4 (that indicates CBC-MAC security) and a key mode 1.

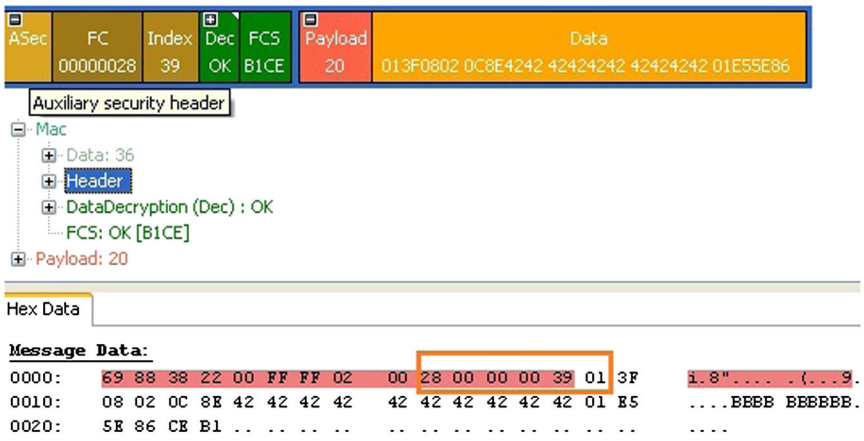


Figure 9. Capture with Perytons, CBC-MAC-4.

In conclusion, with the CBC-MAC-4 byte security, we can say that the frame size increases by 10 bytes because of the 6 frames of the security headers and the 4 bytes of the hash. It is an increment of about 67% in overhead in relation to the CTR security primitive.

On the other hand, this security mechanism has more overhead than CRT and does not satisfy the security requisites of our application.

4.1.4. Evaluation with counter with cipher block chaining-message authentication code security primitives. This presents the same overhead as CBC-MAC security primitives: that is, 10 bytes per message, 4 bytes in the payload with the HASH, and 6 bytes with the security header.

Therefore, this security primitive is preferable to the CBC-MAC, as it presents the same power energy consumption (it has the same overhead), but with a better security level.

This security mechanism provides all the requisites of the messages from non-anchor nodes to anchor nodes and the messages from anchor nodes to non-anchor nodes: confidentiality, security semantics, authorization, and authentication. But it does not meet the security requisites of the messages between anchor nodes: it does not preset the services of integrity or availability.

4.2. Evaluation of other non-normalized wireless sensor network security primitives

Outside the IEEE 802.15.4 standard, several schemes have been proposed, looking for an increase in security levels compared to the standard. But in general, these schemes require a software implementation to be used in the current WSN Devices.

All these methods are based on the use of encryption protocol. Generally, symmetric encryption algorithms are used because they require less microcontroller resources. One of the most widely used security algorithms in WSNs is AES. We performed a test to obtain runtime execution of the AES software algorithm for the TelosB platform under *TinyOS* 2.x operating system.

In this test, 16 bytes were encrypted and decrypted. Figure 10 presents the results, showing that, using a software implementation, the time consumed by the microcontroller in security tasks is not negligible. It is more than 1000 times higher than the time consumed by the encryption using standard radio transceiver primitives.

4.2.1. TinySec. TinySec [53] is a security software architecture at the link layer. It is used in many applications, such as in securing electronic gas or water metering communications [54]. It was developed in 2004 by Chris Karlof *et al.* [39], to replace the *secure network encryption protocol utility* security protocol with a *secure protocols for sensor networks*.

TinySec has two modes of operation: *TinySec-AE* (encryption and authentication) and *TinySec-Auth* (authentication only). TinySec can be used with different symmetric encryption algorithms, such as *Triple-DES, AES, RC5, and Skipjack*. An evaluation of these algorithms can be found in

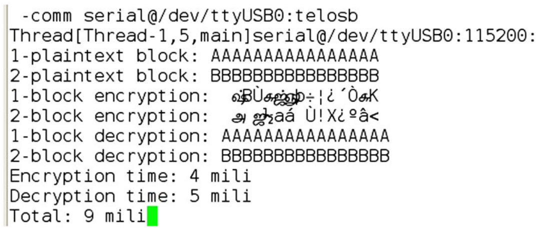


Figure 10. Advanced Encryption Standard runtime software.

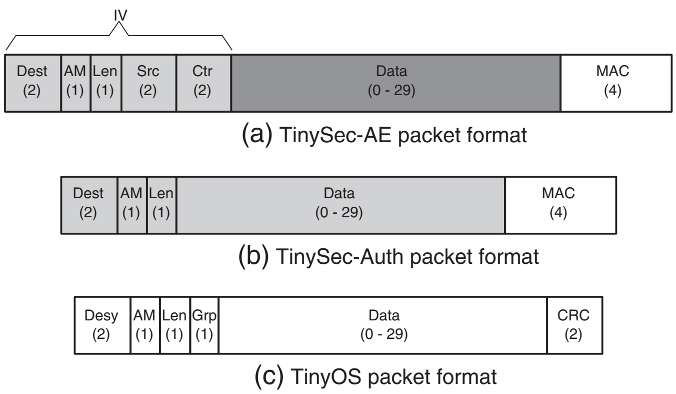


Figure 11. Frame TinySec.

Table II. TinySec consumption.

Service	Consumption (mAH)	Increase (%)
TinyOS Stack	0.000160	—
TinySec-Auth	0.000165	3
TinySec-AE	0.000176	5

[55]. According to our analysis, the first is slow for the software implementation required by WSN, the second requires a block length, and the third is patented and cannot be used freely. *Skipjack* does not present these limitations. We therefore consider *Skipjack* to be the best option for use in WSNs. In our case, we have used it with 64-bit blocks and an 80-bit key.

Figure 11 shows the data frames of *TinySec*, for every operation mode.

In *TinySec*, the CRC of the *TinyOS* format is replaced by a MAC, generated with 4 bytes CBC-MAC. It authenticates the header and payload. In *TinySec*, the initialization vector length (indicated as 'IV' in Figure 11) has to reach a compromise, trying not to introduce too much overhead, while offering enough security semantics. To do this, the header part of 802.15.4 is reused, and 2 bytes are added as a CTR. The number of messages that you can send to a node before repeating the IV vector is 2^{16} messages.

Tiny Sec-Auth has an overhead of 1 and *TinySec-AE* an overhead of 5, in comparison with a *TinyOS* standard frame. Table II shows the additional energy cost used in processing the information needed to send a package of 24 bytes of payload.

For this security protocol, the maximum length of the application payload is 29 bytes [53]. However, this current implementation can be modified easily to allow bigger messages to be sent.

Focusing on our test, the most important characteristics of *TinySec* are

- It offers authentication and integrity.
- It offers confidentiality.
- It is easy to use.
- It introduces a low overhead, because both the delay and throughput are not adversely affected.

- As schema, it uses a key distribution network shared file.
- It does not provide the service of *replay protection*. The authors [53] argue that this management is best carried out in the upper layers, with one reason being that the link-level lacks knowledge of the network topology.
- Because it is a software implementation, it requires additional microcontroller processing.

In conclusion, this software implementation offers high security levels. It can be applied to WSN, but it increases power consumption, due to the software encrypt and decrypt tasks. Despite this, *TinySec* is one of the software implementations with the lowest power consumption, but this is still several times higher than the consumption of hardware radio primitives.

4.2.2. MiniSec. *MiniSec* [56] is a secure software network-level protocol. It is designed to work over *TinyOS* and is freely available. It unifies the benefits of *TinySec* (low power consumption) and *ZigBee* (high rate of security).

To do this, it follows three premises: first, the authenticated encryption mode is used in one single step; second, only a small number of bits of the IV vector (Figure 11) are sent; and third, two modes of operation are featured: communications in unicast and broadcast modes.

This is achieved through the reduction of power consumption of the radio transceiver by using time synchronization. Time synchronization reduces the energy consumption of the radio transceiver, although energy consumption still increases as time synchronization requires an extra computation in the nodes. The security services that *MiniSec* provides are

- Authentication and integrity, which can be carried out by appending the MAC code to the package.
- Confidentiality.
- Achievement of the *MiniSec* previous service in a single stage using the encryption mode *offset codebook (OCB)* by applying the symmetric encryption algorithm *Skipjack* block with a 64-bit and 80-bit key.
- Protection against replicas of messages.
- Security semantics with the IV vector through an internal 64-bit CTR, which only sends 3 bits *(LB optimization)*.

As can be seen, this technique uses *Skipjack* as encryption algorithm, with an 80-bit key length. It may be unsafe for a short period, but it can easily be replaced with one more secure technique, such as AES.

According to [56], the packet length should never be greater than 29 bytes. However, the IEEE standard in its two versions (2003–2006) allows a maximum MAC packet size of 127 bytes (header + payload + CRC). The authors have considered the default application payload of *TinyOS* to implement this protocol, but for applications with exchange messages larger than 28 bytes, this could increase the power consumption wasted by headers.

Most WSN applications (such as the proposed people tracking application) do not require messages larger than 29 bytes. However, to avoid the over cost of extra headers in the case of large messages, *MiniSec* has been designed to keep the headers as small as possible. As an example, they avoid the use of IV vector, because it consumes about 1/3 of the required extra headers of *TinySec*. *MiniSec* requires only 3 bytes of overhead more than *TinyOS* standard frames.

One disadvantage of this security technique is that it is designed to be used over *Tinyos 1.x*. The implementation over Tinyos 2.x is still in progress.

The following section briefly describes the modes of operation of *MiniSec*: *MiniSec-U* and *MiniSec-B*.

- **MiniSec-U**

  Consists of the unicast communication between a sender node and a receiver node. The authors [56] assume symmetric keys established between each pair of nodes. It uses a synchronized and monotone CTR increasing for the *nonce OCB* and applies *Last bit (LB) optimization*. This technique sends the last x-bit CTR in each message. These bits are stored by each receiver locally *per-sender state*.

In addition, this technique allows for implicit resynchronization when packets are lost, provided that this loss is no greater than 2^x packages.

Even so, the receiver can make n additional attempts to synchronize the CTR used for time synchronization. If the receiver receives a packet with a lower CTR than the figure maintained in this state, it will reject the package. This reduces overhead in exchange for RAM consumption. Figure 12 shows the frame format for *MiniSec-U* based on the format of *TinyOS* for radio CC2420. It adds 4 bytes for the *MIC/tag*.

This technique may offer a good level of security and not increase power consumption excessively, despite being a software implementation. But it is limited to unicast messages. For this reason, in our application, it can only be valid for messages between anchor and non-anchor nodes, but not for the communications between anchor nodes.

- **MiniSec-B**

  In this case, the communication is many-to-many, and the network shares the same key. It is clear that with this behavior you cannot use the same scheme for the *replay protection*, the receiving nodes could run out of memory to store the CTRs, and resynchronization time could be increased. *MiniSec-B* uses two techniques *replay protection*, which are

  - A windows method that uses the number of times as nonce to the OCB. An era is defined as a finite interval of time or period; therefore, the time is segmented into times *(E1, E2, ... , In)* . To implement this technique, it should also be considered that the node clocks are not synchronized by defining a maximum Δ_T error or a maximum of Δ_N latency. Otherwise, we would have too many false positives. According to [56] any received packet can only belong to an interval of two times. This technique has vulnerability in the $3\delta_t + 2\delta_n$ range, δ_t is the synchronization error and δ_n is the network latency. However outside of this range of replay, attacks are detected.
  - Bloom filter: This method is used to resolve the vulnerability seen earlier as applied to this technique. It again uses a *CA* CTR to represent the internal state of the sender *A*. This CTR is initialized to zero at the beginning of each time. For this reason, it can be appended to the message without increasing the overhead excessively. The nonce used is *(sender ID, CA ,Ei)* . The resulting encrypted package will be $(C, tag) = OCB_{K_A} C_A node_i d|||e_i, M, H)$. The receiving node maintains a buffer in order to be able to identify packets received. This structure is accessed using a hash function employing the ID node and CTR as parameters. If this function returns to the position of empty buffer, the packet is not a replica, and the position occupied is marked. Figure 13 shows an example of the encryption scheme and the data that is sent in the package. As there are only two possible valid times, the receiver decrypts using these times, which is why the epoch does not travel in the package, not having to increase the overhead.

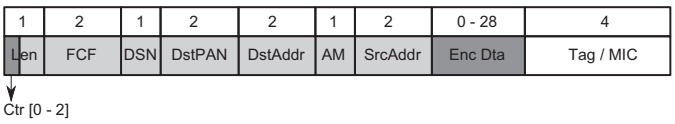


Figure 12. Frame MiniSec-U.

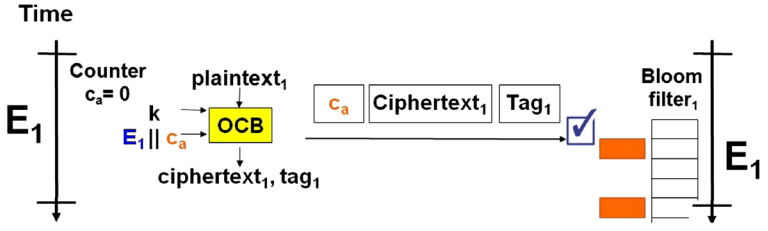


Figure 13. Schema bloom filter [56].

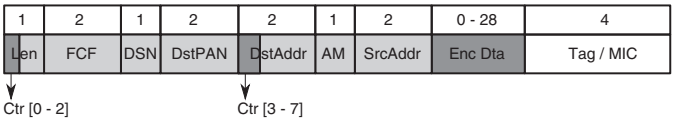


Figure 14. Frame MiniSec-B.

With this technique, the percentage of false negatives is zero, and there are very few false positives. By changing the size of the buffer and the duration of the period, false positives can be further reduced. The frame format for *MiniSec-B* is shown in Figure 14.

In conclusion, this technique offers a similar security level to *MiniSec-U*, but allows communications between multiple devices. This technique is designed particularly for devices with low resources, not requiring heavy processing, and keeping the communications overhead low, but again, it presents higher consumption than using hardware primitives.

4.2.3. TinyECC. *TinyECC* [57] is a security library configurable for WSNs using elliptic curve cryptography (ECC). It was not until 2004 that the study of [58] demonstrated that it was possible to use asymmetric cryptography through elliptic curves.

Elliptic curve cryptography is based on asymmetric cryptography, such as RSA. But the standard implementation of the RSA algorithm in WSNs is not possible, as it requires high computation resources. However, the security provided by 160-bit ECC key is comparable to using an RSA with 1024 bits.

The current version of *TinyECC* is the 2.0, released in 2010. It is freely available and can been installed and tested over TelosB nodes using *TinyOS 2.x*.

The main contributions of *TinyECC* are as follows:

- It provides a public encryption scheme based on *Elliptic Curve Diffie–Hellman* variant of the distribution pattern key *Diffie–Hellman*, *Elliptic Curve Digital Signature Algorithm* variant of the digital signature algorithm DSA, and *Elliptic Curve Integrated Encryption Software* public encryption scheme.
- Portability: *TinyECC* can be installed in *TelosB, micaz, Tmote Sky, and Imote2* devices.
- Efficiency: *TinyECC* uses a series of optimizations to reduce power consumption as much as possible. It allows for software implementation to be adapted to application requisites. These optimizations include: optimizations for calculations with large prime numbers (method for the reduction of *Barret*, multiplication square, and hybrid) and optimizations for the operations of ECC (trick *Shamir*, sliding window for scalar multiplications, and projective coordinates), including the assembler code for some critical operations.

The consumption of ROM and RAM varies depending on the optimization used. It makes it possible to adapt security to application requisites.

Figure 15 shows our execution time results, obtained as the average *Elliptic Curve Digital Signature Algorithm* time with 10 iterations of the algorithm over a TelosB node. According to our results, the execution times are slightly higher than those indicated by [57].

TinyECC is the most secure of the analyzed techniques over WSN, but it also has the highest power consumption. This is because, despite the optimization, it requires a long processing time. Therefore, it is not a good option with mobile devices with high energy restrictions.

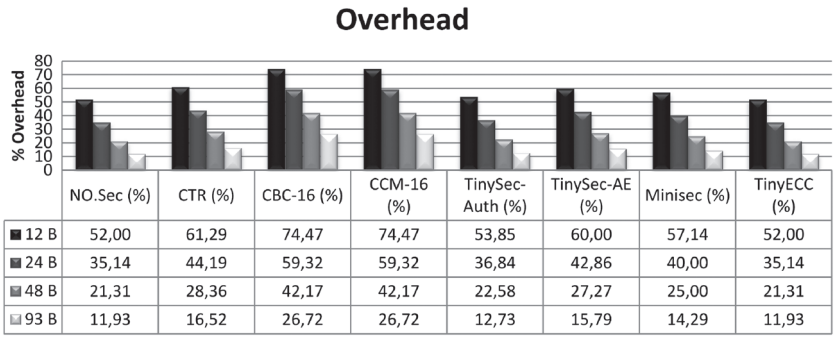
4.3. Comparatives between security primitives

We compared the security primitives in terms of the cost of overheads, throughputs, and energy consumption. This section summarizes the results.

4.3.1. Analysis of the overheads. Figure 16 shows the overheads for each one of the security schemes considered for this application. In this figure, overheads represent the portion of the total

```
-------------------- round 1 --------------------
Private key:
d: 1bb495515a273d9c86435ee440315d4f4c29716f
[ time of ECC.init() is 2.692 sec ]
Public key:
x: c09a19a652c1bb72d0e5810fc2c254606ae2cca7
y: 2fcffa30b9b1696f1549679744175b9dfbb04280
[ time of public key generation is 2.889 sec ]
[ time of ECDSA.init() is 5.187 sec ]
content and signature
msg: 33700a061e2350ca86133964d2bb69c497386fc88f01143276f
fdecc29b206a
signature
r: fd2b99580c18532a1b3aff89d11bf6c07ff82bf1
s: e60a92964988055cd0e52148155e104d614daa91
[ time of signature generation is 3.044 sec ]
[ time of signature verification is 3.889 sec ] (pass)
Average timing result
ECC.init(): 2.6894999
ECDSA.init(): 5.1879997
public key gen: 2.862
sign: 3.034
verify: 3.87
```

Figure 15. TinyECC execution time.



	NO.Sec (%)	CTR (%)	CBC-16 (%)	CCM-16 (%)	TinySec-Auth (%)	TinySec-AE (%)	Minisec (%)	TinyECC (%)
12 B	52,00	61,29	74,47	74,47	53,85	60,00	57,14	52,00
24 B	35,14	44,19	59,32	59,32	36,84	42,86	40,00	35,14
48 B	21,31	28,36	42,17	42,17	22,58	27,27	25,00	21,31
93 B	11,93	16,52	26,72	26,72	12,73	15,79	14,29	11,93

Figure 16. Overhead operation modes for various payload sizes.

message sent (considering header, application payload and CRC) that is used for the headers. To obtain this, we considered application messages with 12, 24, 48, and 93 bytes.

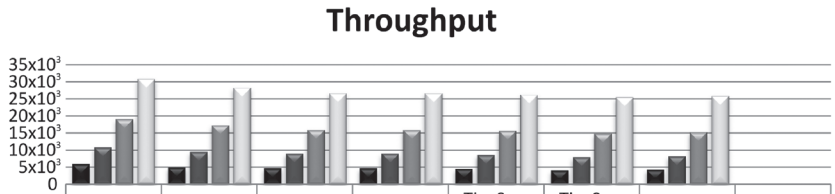
In general, as the size of the application payload increases, the impact of overheads decreases. But many WSNs use small application payloads. For this reason, in common WSN applications, such as people tracking, the cost of overheads is not depictable.

Focusing only on IEEE 802.15.4 primitives, the CTR security mechanism does not have high security, but it presents the lowest overhead cost. CBC-MAC and CCM-MAC add the same cost in headers, but CCM-MAC presents higher security.

Software security implementation has in some cases a lower cost in headers, especially in the case of *MiniSec*.

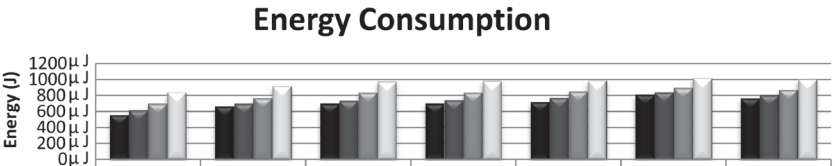
4.3.2. Analysis of the throughput. We could define *maximum throughput* as the number of bits per time unit that can be transmitted from a top layer. To calculate throughput, a series of assumptions have been taken into account: there are no collisions, there is only one sender and one receiver, the beacon-less version of IEEE 802.15.4 is used, there is no ACK, the packet sent only contains a node ID, and it is processed as soon as the event is received.

Our study is different from [59] and [60] that take into account a number of measures of time such as the *inter frame space* time or *back off period* that complicate its calculation. We have opted for a less accurate study, but one which is sufficient for comparing the different security mechanisms studied here.



	No.Sec	CTR	CBC-16	CCM-16	TinySec-Auth	TinySec-AE	Minisec	TinyECC
12 B	6000,0	5052,7	4800,0	4800,0	4651,2	4137,9	4379,5	8,7
24 B	10909,1	9600,0	9142,8	9142,8	8699,5	8010,0	8340,5	17,5
48 B	19200,0	17454,7	16000,1	16000,1	15789,3	14982,5	15372,3	34,9
93 B	31000,2	28397,0	26762,4	26762,4	26364,4	25566,8	25958,9	67,7

Figure 17. Throughput.



	Eng. No Sec	Eng. CTR	Eng. CBC-16	Eng. CCM-16	Eng. TinySec-Auth	Eng. TinySec-AE	Eng. Minisec
12 B	5,60E-04	6,65E-04	7,00E-04	7,00E-04	7,22E-04	8,12E-04	7,67E-04
24 B	6,16E-04	7,00E-04	7,35E-04	7,42E-04	7,73E-04	8,39E-04	8,06E-04
48 B	7,00E-04	7,70E-04	8,40E-04	8,40E-04	8,51E-04	8,97E-04	8,74E-04
93 B	8,40E-04	9,17E-04	9,73E-04	9,80E-04	9,88E-04	1,02E-03	1,00E-03

Figure 18. Consumption.

In *TinyOS*, *AMSend.senddone* starts a new dispatch. Every 10 s, the number of packets sent and the average time for each dispatch are displayed on the screen through UART communication with the station. As the structure of each packet is known (PHY-MPDU), one can calculate the ratio of the number of bits per second. The experiments have been made with a payload size of 12, 24, 48, and 93 bytes.

Figure 17 shows that the *throughput* increases when the packet size is greater and decreases because of security.

As one can see, hardware encryption has higher throughput than software encryption. The throughput of *TinyECC*, due to the time required for encrypting the information, is negligible compared to the other security primitives.

4.3.3. Analysis of energy consumption. Using the *TelosB* datasheet [48], one can find out the consumption of all node devices. The time required to process a message and send a packet are also given. Using all this information, the power consumption can be estimated.

Other authors [61, 62] have performed more accurate studies. They obtained the energy consumption measuring directly with an oscilloscope or a power analyzer over a real circuit.

Our estimation is less accurate, but it is sufficient for a comparative analysis of the different security techniques. Moreover, the time taken to encrypt a message, using hardware implementation, is roughly 1 µs, and it is very difficult to measure accurately the energy consumed during such a short period.

The results of evaluating the power consumption are summed-up in Figure 18. In this figure, the power consumption is not considered because the high time required to encrypt the information produces energy consumption, which is several times higher than it is for the rest. As can be seen, power consumption increases with the size of the messages. This is because radio transceivers have high power consumption, compared to the other node devices.

According to the security primitives, software implementation consumes more power than hardware. In a comparison of the different software implementations, asymmetric security was shown to have several times the power consumption of symmetric security implementations.

5. CONCLUSIONS

Wireless communications as used in WSNs are exposed to potential attacks. To overcome this, the use of security mechanisms is required. But these security mechanisms increase the power consumption and reduce the lifetime of WSN devices.

For this reason, the security needs for each particular scenario have to be studied, searching for a trade-off between security and power consumption. This paper has focused on evaluating this trade-off for people tracking applications.

Without security in people tracking systems, a potential intruder can retrieve confidential information. Therefore, there is a need for security management. The minimum requirements for these security mechanisms include: confidentiality, data integrity, authentication of components, and freshness of data. In addition, one cannot forget that energy consumption in WSN devices has to be kept as low as possible. We have found that there is a cost for the use of security mechanisms: an increase in the overhead of the package and an increase in message transmission, which in turn leads to additional actions and therefore, higher energy consumption.

In order to fulfill these premises, we have conducted a review and comparative analysis of different security mechanisms.

This analysis demonstrates that software security schemes are securer, but they involve more computational costs and energy consumption than a hardware solution.

In general, all the considered software implementations satisfy the security requisites of all the messages shared between devices in our application, but the increase in energy consumption makes them unsuitable for use in applications with non-anchor nodes with high power energy constraints.

In turn, our tests demonstrate that asymmetric cryptography is also more expensive than symmetric cryptography.

Therefore, for the proposed application, we conclude that different security techniques will be required, depending on the information managed by the system:

- *Messages between anchor nodes and non-anchor nodes*: Non-anchor nodes have high energy restrictions and their messages require lower security levels. For this reason, *CTR with CCM-MAC*, implemented by hardware symmetric cryptography (such as an AES mechanism, provided by a CC2420 radio transceiver used in TelosB nodes) represents the best trade-off between security and power consumption for these messages.
- *Messages between anchor nodes*: This requires higher security levels and has less energy restrictions. Thus, software implementations can be used. *MiniSec-B* or *TinySec* satisfy all the security requirements, without increasing power consumption excessively.

However, this security mechanism requires the use of credentials, that is, symmetric keys for the AES algorithm. This requires a secure sublayer for the distribution and management of these keys. Regarding our evaluation, a public key infrastructure *PKI* could be developed, thanks to the development of elliptic curves. This allows for the use of costly asymmetric cryptography energy in *WSNs*.

Regarding future studies, the authors are currently deploying a real prototype of the proposed system to test it in a real scenario.

ACKNOWLEDGEMENTS

This research has been supported by the 'Consejería de Innovación, Ciencia y Empresa, Junta de Andalucía', Spain, through the excellence project ARTICA (reference number: P07-TIC-02476) and ESAPIENS (TIC-5705) and by the 'Cátedra de Telefónica, Inteligencia en la Red', Seville, Spain.

REFERENCES

1. Akyildiz I, Su W, Sankarasubramaniam Y, Cayirci E. Wireless sensor networks: a survey. *Computer Networks* 2002; **38**(4):393–422. DOI: 10.1016/S1389-1286(01)00302-4.
2. Xia F, Yang LT, Wang L, Vinel A. Internet of things. *International Journal of Communication Systems* 2012; **25**(9):1101–1102.
3. Liu Y, Chen Z, Xia F, Lv X, Bu F. An integrated scheme based on service classification in pervasive mobile services. *International Journal of Communication Systems* 2012; **25**(9):1178–1188.
4. Le A, Loo J, Lasebae A, Aiash M, Luo Y. 6lowpan: a study on qos security threats and countermeasures using intrusion detection system approach. *International Journal of Communication Systems* 2012; **25**(9):1189–1212.
5. Na R, Ren Y, Hori Y, Sakurai K. A generic evaluation method for key management schemes in wireless sensor network. *Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication (ICUIMC'11),* ACM: New York, NY, USA, 2011; 55:1–55:9, DOI: 10.1145/1968613.1968680.
6. Sankaran S, Husain MI, Sridhar R. Idkeyman: an identity-based key management scheme for wireless ad hoc body area network. *4th Annual Symposium on Information Assurance, 12th Annual 2009 NYS Cyber Security Conference*, Albany, NY, 2009; 23–28.
7. Roman R, Alcaraz C, Lopez J, Sklavos N. Key management systems for sensor networks in the context of the internet of things. *Computers and Electrical Engineering* 2011; **37**(2):147–159. DOI: 10.1016/j.compeleceng.2011.01.009.
8. Roman R, Alcaraz C, Lopez J. A survey of cryptographic primitives and implementations for hardware-constrained sensor network nodes. *Mobile Networks and Applications* 2007; **12**(4):231–244. DOI: 10.1007/s11036-007-0024-2.
9. Trilok I, Patil MU. Design and implementation of flexible framework for secure wireless sensor network applications. *International Journal of Computer Science and Information Security* 2010; **8**(3):188–194.
10. Chen S, Dunkels A, Österlind F, Voigt T, Johansson M. Time synchronization for predictable and secure data collection in wireless sensor networks. *The Sixth Annual Mediterranean Ad Hoc Networking Workshop*, Corfu, Greece, 2007; 165–172.
11. Oliveira LML, de Sousa AF, Rodrigues JJPC. Routing and mobility approaches in ipv6 over lowpan mesh networks. *International Journal of Communication Systems* 2011; **24**(11):1445–1466.
12. Daidone R, Dini G, Tiloca M. On experimentally evaluating the impact of security on ieee 802.15.4 networks. *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, Barcelona, Spain, 2011; 1–6, DOI: 10.1109/DCOSS.2011.5982223.
13. Liu C-X, Liu Y, Zhang Z-J, Cheng Z-Y. High energy-efficient and privacy-preserving secure data aggregation for wireless sensor networks. *International Journal of Communication Systems* 2012; **26**(3):380–394. DOI: 10.1002/dac.2412.
14. Wang Y, Shi P, Li K, Chen Z. An energy efficient medium access control protocol for target tracking based on dynamic convey tree collaboration in wireless sensor networks. *International Journal of Communication Systems* 2012; **25**(9):1139–1159.
15. Shi Q, Comaniciu C, Wang D, Tureli U. Cross-layer mac design for location-aware wireless sensor networks. *International Journal of Communication Systems* 2011; **24**(7):872–888.
16. Mao G, Fidan B, Anderson BDO. Wireless sensor network localization techniques. *Computer Network* 2007; **51**(10):2529–2553. DOI: 10.1016/j.comnet.2006.11.018.
17. Piontek H, Seyffer M, Kaiser J. Improving the accuracy of ultrasound-based localisation systems. *Personal and Ubiquitous Computing* 2007; **11**(6):439–449. DOI: 10.1007/s00779-006-0096-1.
18. Awad A, Frunzke T, Dressler F. Adaptive distance estimation and localization in wsn using rssi measures. *10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007)*, Lubeck, Germany, 2007; 471–478, DOI: 10.1109/DSD.2007.4341511.
19. Qiu T, Zhou Y, Xia F, Jin N, Feng L. A localization strategy based on n-times trilateral centroid with weight. *International Journal of Communication Systems* 2012; **25**(9):1160–1177.
20. Shao-Hua W, Nai-Tong Z. A two-step toa estimation method for uwb based wireless sensor networks. *Journal of Software* 2007; **18**(05):1164–1172. CNKI Journal.
21. Alam MS, Alsharif S, Haq N. Efficient cdma wireless position location system using tdoa method. *International Journal of Communication Systems* 2011; **24**(9):1230–1242.
22. Rong P, Sichitiu M. Angle of arrival localization for wireless sensor networks. *Sensor and Ad Hoc Communications and Networks (SECON '06). 3rd Annual IEEE Communications Society on*, Reston, VA, 2006; 374–382, DOI: 10.1109/SAHCN.2006.288442.
23. Bulusu N, Heidemann J, Estrin D. Gps-less low-cost outdoor localization for very small devices. *Personal Communications, IEEE* 2000; **7**(5):28–34. DOI: 10.1109/98.878533.
24. Gao GQ, Lei L. An improved node localization algorithm based on dv-hop in wsn. *2010 2nd International Conference on Advanced Computer Control (ICACC)*, vol. 4, Shenyang, 2010; 321–324, DOI: 10.1109/ICACC.2010.5486944.
25. Abumansoor O, Boukerche A. A secure cooperative approach for nonline-of-sight location verification in vanet. *IEEE Transactions on Vehicular Technology* 2012; **61**(1):275–285. DOI: 10.1109/TVT.2011.2174465.
26. Lorincz K, Welsh M. MoteTrack: a robust, decentralized approach to RF-based location tracking. In *Location and Context-Awareness (LoCA),* vol. 3479, Strang T, Popien CL (eds), Lecture Notes in Computer Science. Springer: Berlin, Heidelberg, 2005; 63–82.

27. Boukerche A, Oliveira H, Nakamura E, Loureiro A. Secure localization algorithms for wireless sensor networks. *Communications Magazine, IEEE* 2008; **46**(4):96–101. DOI: 10.1109/MCOM.2008.4481347.
28. Zeng Y, Cao J, Hong J, Zhang S, Xie L. Secure localization and location verification in wireless sensor networks: a survey. *The Journal of Supercomputing* 2013; **64**(3):685–701.
29. Liu D, Ning P, Du W. Detecting malicious beacon nodes for secure location discovery in wireless sensor networks. *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS 2005)*, Columbus, Ohio, USA, 2005; 609–619, DOI: 10.1109/ICDCS.2005.21.
30. Li Z, Trappe W, Zhang Y, Nath B. Robust statistical methods for securing wireless localization in sensor networks. *Fourth International Symposium on Information Processing in Sensor Networks (IPSN 2005)*, Los Angeles, California, USA, 2005; 91–98, DOI: 10.1109/IPSN.2005.1440903.
31. García-Otero M, Zahariadis T, Álvarez F, Leligou HC, Población-Hernández A, Karkazis P, Casajús-Quirós FJ. Secure geographic routing in ad hoc and wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking - Special Issue on Signal Processing-Assisted Protocols and Algorithms for Cooperating Objects and Wireless Sensor Networks* 2010; **10**:1–12. DOI: 10.1155/2010/975607.
32. Jiang J, Han G, Shu L, Chao HC, Nishio S. A novel secure localization scheme against collaborative collusion in wireless sensor networks. *Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International*, Istanbul, 2011; 308–313, DOI: 10.1109/IWCMC.2011.5982551.
33. Pandarinath P. Secure localization with defense against selective forwarding attacks in wireless sensor networks. *2011 3rd International Conference on Electronics Computer Technology (ICECT)*, vol. 5, Kanyakumari, 2011; 112–117, DOI: 10.1109/ICECTECH.2011.5941968.
34. Larios D, Barbancho J, Molina F, Leon C. Lis: localization based on an intelligent distributed fuzzy system applied to a wsn. *Ad Hoc Networks* 2012; **10**(3):604–622. DOI: 10.1016/j.adhoc.2011.11.003.
35. Ieee 802.15.4, 2003.
36. Baronti P, Pillai P, Chook VW, Chessa S, Gotta A, Hu YF. Wireless sensor networks: a survey on the state of the art and the 802.15.4 and zigbee standards. *Computer Communications* 2007; **30**(7):1655–1695. DOI: 10.1016/j.comcom.2006.12.020.
37. Rehana J. Security of wireless sensor network. In *Seminar on Internetworking, TKK Technical Reports in Computer Science and Engineering, B TKK-CSE-B5*, Vol. 1, Ylä-Jääski A, Suoranta S (eds). Helsinki University of Technology Press: Espoo, 2009; 80–85.
38. Sharma R, Chaba Y, Singh Y. Analysis of security protocols in wireless sensor network. *International Journal of Advanced Networking & Applications* 2010; **2**(03):707–713.
39. Karlof C, Wagner D. Secure routing in wireless sensor networks: attacks and countermeasures. *2003 IEEE International Workshop on Sensor Network Protocols and Applications. Proceedings of the First IEEE*, Anchorage, AK, USA, 2003; 113–127, DOI: 10.1109/SNPA.2003.1203362.
40. Healy M, Newe T, Lewis E. Security for wireless sensor networks: a review. *Sensors Applications Symposium, 2009. SAS 2009. IEEE*, New Orleans, LA, 2009; 80–85, DOI: 10.1109/SAS.2009.4801782.
41. Yoon EJ, Yoo KY. Cryptanalysis of a simple three party password-based key exchange protocol. *International Journal of Communication Systems* 2011; **24**(4):532–542.
42. Tang Hb, Liu Xs. Cryptanalysis of a dynamic id-based remote user authentication with key agreement scheme. *International Journal of Communication Systems* 2012; **25**(12):1639–1644.
43. Malekzadeh M, Ghani AAA, Subramaniam S. A new security model to prevent denial-of-service attacks and violation of availability in wireless networks. *International Journal of Communication Systems* 2012; **25**(7):903–925.
44. Hasan MM, Prakash R, Jue JP. Parallel and gaussian frequency hopping for dynamic coexistence in the unlicensed band. *International Journal of Communication Systems* 2011; **24**(9):1163–1182.
45. Datema S. A case study of wireless sensor network attacks. *MSC Thesis*, Delft Univertity of Technology, 2005.
46. IEEE 802.15.4a-2007, 2007.
47. Levis P, Gay D. *TinyOs Programming*. Cambridge University Press: Cambridge, United Kingdom, 2009.
48. Polastre J, Szewczyk R, Culler D. Telos: enabling ultra-low power wireless research. *Fourth International Symposium on Information Processing in Sensor Networks, 2005. IPSN 2005*, Los Angeles, California, USA, 2005; 364–369, DOI: 10.1109/IPSN.2005.1440950.
49. Instruments T. DataSheet Chipcon's SmartRF CC2420. (Available from: http://www.ti.com/lit/ds/symlink/cc2420.pdf).
50. Perytons. 802.15 protocol analyzer. (Available from: www.perytons.com).
51. Texas. Smartrf protocol packet sniffer. (Available from: www.ti.com/tool/packet-sniffer).
52. Sastry N, Wagner D. Security considerations for ieee 802.15.4 networks. *Proceedings of the 3rd ACM Workshop on Wireless Security,* ACM: New York, NY, USA, 2004; 32–42, DOI: 10.1145/1023646.1023654.
53. Karlof C, Sastry N, Wagner D. TinySec: a link layer security architecture for wireless sensor networks. In *SenSys '04: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*. ACM Press: New York, NY, USA, 2004; 162–175, DOI: 10.1145/1031495.1031515.
54. Spanagel D, Prevostini M. *System-Level Design of a Tinysec-Based Secure Low-Power Protocol for a Remote Meter Reader*. ALaRI pubblications: University of Lugano, 2006.
55. Law YW, Doumen J, Hartel P. Survey and benchmark of block ciphers for wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)* 2006; **2**(1):65–93. DOI: 10.1145/1138127.1138130.

56. Luk M, Mezzour G, Perrig A, Gligor V. Minisec: a secure sensor network communication architecture. *6th International Symposium on Information Processing in Sensor Networks (IPSN 2007)*, Cambridge, MA, 2007; 479–488, DOI: 10.1109/IPSN.2007.4379708.
57. Liu A, Ning P. Tinyecc: a configurable library for elliptic curve cryptography in wireless sensor networks. *International Conference on Information Processing in Sensor Networks (IPSN '08)*, St. Louis, MO, 2008; 245–256, DOI: 10.1109/IPSN.2008.47.
58. Gura N, Patel A, Wander A, Eberle H, Shantz SC. Comparing elliptic curve cryptography and rsa on 8-bit cpus. In *CHES, Lecture Notes in Computer Science,* vol. 3156, Joye M, Quisquater JJ (eds). Springer: Springer Berlin Heidelberg, 2004; 119–132.
59. Latré B, Mil PD, Moerman I, Dhoedt B, Demeester P, Dierdonck NV. Throughput and delay analysis of unslotted IEEE 802.15.4. *Journal of Networks* 2006; **1**:20–28. DOI: 10.4304/jnw.1.1.20-28.
60. Sun T, Chen LJ, Han CC, Yang G, Gerla M. Measuring effective capacity of ieee 802.15.4 beaconless mode. *Wireless Communications and Networking Conference (WCNC 2006). IEEE*, vol. 1, Las Vegas, NV, 2006; 493–498, DOI: 10.1109/WCNC.2006.1683513.
61. Milenkovic A, Milenkovic M, Jovanov E, Hite D, Raskovic D. An environment for runtime power monitoring of wireless sensor network platforms. *Proceedings of the Thirty-Seventh Southeastern Symposium on System Theory (SSST'05)*, Tuskegee, Alabama United States, 2005; 406–410, DOI: 10.1109/SSST.2005.1460946.
62. Lee J, Kapitanova K, Son SH. The price of security in wireless sensor networks. *Computer Networks: The International Journal of Computer and Telecommunications Networking* 2010; **54**(17):2967–2978. DOI: 10.1016/j.comnet.2010.05.011.