Computer Communications 47 (2014) 65–76

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

On evaluating the performance impact of the IEEE 802.15.4 security sub-layer

Roberta Daidone *, Gianluca Dini, Giuseppe Anastasi

Department of Information Engineering, University of Pisa, Pisa, Italy

ARTICLE INFO

Article history:
Received 22 May 2013
Received in revised form 13 December 2013
Accepted 24 March 2014
Available online 19 April 2014

Keywords:
IEEE 802.15.4
Security
Performance evaluation
Wireless sensor network

ABSTRACT

Nowadays, wireless sensor networks (WSNs) are used in a wide range of application scenarios ranging from structural monitoring to health-care, from surveillance to industrial automation. Most of these applications require forms of secure communication. On the other hand, security has a cost in terms of reduced performance. In this paper we refer to the IEEE 802.15.4 standard and investigate the impact of the 802.15.4 security sub-layer on the WSN performance. Specifically, we analyze the impact that security mechanisms and options, as provided by the standard, have on the overall WSN performance, in terms of latency, goodput, and energy consumption. To this end, we develop an analytical model and a security enabled simulator. We also use a real testbed, based on a complete open-source implementation of the standard, to validate simulation and analytical results, as well as to better understand the limits of the current WSN technology.

1. Introduction

IEEE 802.15.4 is a standard for low-rate wireless personal area networks with a focus on enabling low power devices, personal area networks, and wireless sensor networks (WSNs). The standard is characterized by maintaining a high level of simplicity, allowing for low cost and low power implementations [1]. IEEE 802.15.4 is adopted in a wide range of application scenarios, ranging from structural monitoring to health care, from military surveillance to industrial automation. Most of these applications require forms of secure communication. For this reason, IEEE 802.15.4 specification includes a number of security provisions and options that constitute the *security sublayer* [1]. The security sublayer provides link-level security services by guaranteeing confidentiality and/or authenticity and replay detection on a per-frame basis. Specifically, it provides two *security parameters*, the *security level*, which specifies one (out of eight) possible security service, and the *key identifier mode*, which specifies one (out of four) possible way to store and lookup cryptographic keys.

Security and performance of IEEE 802.15.4 have been thoroughly analyzed. For instance, a performance analysis of IEEE 802.15.4 without considering security has been performed in quite a few papers including [2–4]. In addition, a security analysis of IEEE 802.15.4 security sublayer, its services, vulnerabilities, and related countermeasures, has been presented in [5–7]. However, a thorough analysis of the impact that the security sublayer has on the overall IEEE 802.15.4 performance is missing. Some related works have been presented but they focus on specific aspects. For example, [5,7–12] deal with the cost for the sensor node of using off-the-shelf ciphers, encryption modes, and authentication algorithms in terms of energy, storage and computing overhead. Other works focus instead on the cost of key establishment, an important although collateral aspect [3,13–15]. However, what it is really missing is an analysis providing quantitative indications regarding the impact that the security sublayer has on the overall standard performance. We believe that this analysis is crucial. Security and performance compete for the same system resources, namely memory, CPU, bandwidth and energy, which are scarce in low power, low cost sensor devices. Therefore, quantitative indications regarding resource consumption are fundamental to design and implement adequate performance-security trade-offs in IEEE 802.15.4-based applications.

In this paper we present a performance analysis of the IEEE 802.15.4 security sublayer. In particular we evaluate the impact that security levels and key identification modes have on network performance indices such as latency, goodput, and energy consumption. The objective of our analysis is twofold. On the one hand, we aim at evaluating how security impacts on network performance, i.e., how security services (e.g. confidentiality and/or authenticity and replay detection) and security options (e.g., the

* Corresponding author. Tel.: +39 0502217549; fax: + 39 0502217600.
E-mail addresses: roberta.daidone@iet.unipi.it (R. Daidone), gianluca.dini@iet.unipi.it (G. Dini), giuseppe.anastasi@iet.unipi.it (G. Anastasi).

http://dx.doi.org/10.1016/j.comcom.2014.03.025
0140-3664/

length of the message authentication code) influence performance. On the other hand, we aim at devising a cost model that allows designers and implementers to carry out, for example at pre-deployment, simulation and/or performance analysis that include security too.

IEEE 802.15.4 security sublayer provides its services to above network and application layers. Although IEEE 802.15.4 security sublayer is the natural choice for ZigBee [16], nevertheless this is not the only option. Actually, different network and/or application protocols, can be deployed on top of the IEEE 802.15.4 MAC layer [17]. For this reason, we have chosen to evaluate the performance of the IEEE 802.15.4 security sublayer in isolation, irrespective of the actual network or application protocols that will be layered on top of it, so as to give our work a wider and more general scope.

We claim that our work has the following merits. First, we show that (i) securing traffic has performance costs due to the increased length of a secured frame and the additional computations required for security processing; and, (ii) these costs depend on the chosen security parameters. Second, we show that the highest cost has to be paid when we switch from unsecured to secured traffic. However, when traffic is secured via hardware-based cryptography, the chosen security service has little, or even negligible, impact on performance. Conversely, when traffic is secured via software-based cryptography, the performance penalty strongly depends on the chosen security level. Third, we propose a simple yet effective analytical model that we also use to extend an Ns2-based simulator of the IEEE 802.15.4 MAC protocol. The model and the extended simulator have been experimentally validated by means of real measurements on an open-source implementation of the IEEE 802.15.4 for TinyOS on Tmote Sky motes [18,19]. Finally, the availability of an implementation of the standard has allowed us to evaluate the memory overhead related to the security sublayer. It turns out that, while the code implementing the sublayer has limited memory occupancy, the internal data structures may constitute a constraint to the system scalability.

To the best of our knowledge, this is the first work that analyzes performance of the IEEE 802.15.4 security sublayer through analysis, simulation and experimental measurements, and achieves the aforementioned results. The closest work to ours is [5]. However, in this work Chen et al. present a performance analysis that is only based on simulations and lacks of any experimental validation. In addition, they neglect the impact of the key identifier mode, and refer to a partial implementation of the security sublayer that fails to capture the memory costs and the consequent constraints on the system scalability.

The paper is organized as follows. Section 2 discusses related works. Section 3 provides an overview of the standard focusing, in particular, on the CSMA/CA access protocol. The IEEE 802.15.4 security sublayer services are presented in Section 3.1. Section 4 presents our performance evaluation. More precisely, Section 4.1 presents the analytical model of the costs of security in terms of latency, goodput and energy consumption. In Section 4.2 we extend a Ns2-based simulator by means of the analytical model. In Section 4.3 we experimentally evaluate memory consumption of the IEEE 802.15.4 security sublayer. Finally, in Section 5 we draw some conclusions.

2. Related work

Security of IEEE 802.15.4 has been largely investigated. Many works have focused on the analysis of the security services offered by the IEEE 802.15.4 security sublayer, its vulnerabilities, the possible attacks and related countermeasures. Among them, relevant examples are [6,7,20]. In addition to this, another branch of research has focused on the impact of security on performance. For instance, several works have investigated the cost of using off-the-shelf ciphers, encryption modes, and authentication algorithms on wireless sensor nodes in terms of energy consumption, storage and computing overhead. Relevant examples are [8–11,21,22]. However, none of these works focuses on the performance implications of the standard security sublayer.

Xiao et al. and Zhu et al. explored first the impact of security on IEEE 802.15.4 performance [7,12]. However, these works greatly differ from ours for several reasons. They both investigate the cost of a software implementation of the ciphers, encryption modes, and authentication algorithms. Such an investigation only focuses on performance implications on a single node. In contrast, we refer to more efficient sensor node architectures where cryptographic transformations are applied at the hardware level by the communication device. Furthermore, we focus on the overall wireless sensor network performance rather than on a single node. Last, but not the least, we refer to the current version of the standard (released in 2006 [1]) whereas Zhu et al. and Xiao et al. refer to the 2003 version [23]. The two versions greatly differ in the security sublayer.

The closest work to ours is certainly [5]. Like us, Chen et al. refer to the 2006 version of the standard and evaluate the impact of the security sublayer on the overall network performance. They mainly focus on the influence of the packet size and inter-arrival time, whereas we mainly focus on the impact of the security level and the key identification mode. In addition, there are other strong differences. First of all, like [7,12], Chen et al. consider an incomplete implementation of the security sublayer. Actually, their implementation is limited to the cryptographic transformations but completely neglects the data structures required by the security sublayer and, consequently, their impact on memory consumption. Therefore, they fail to capture an important factor limiting the overall scalability. As we consider a complete implementation, we are able to capture such a scalability issue (Section 4.3). Furthermore, they only consider a software implementation of AES-128 [24], the block cipher at the basis of the cryptographic transformations. More in details, they only refer to 20-byte payload frames and consider a 26 ms per-block encryption/decryption delay, a particularly large value derived in a previous work [25]. Instead, we consider several payload sizes (namely 2, 18, and 80 bytes), and use both hardware-based and software-based cryptography. Specifically, we consider an hardware-based cryptography supported by the CC2420 communication device [26], and a software-based cryptography based on an implementation of AES-128 taken from the TinyOS security algorithms repository [27]. From our experiments it turns out that hardware-based cryptography accounts for an approximately constant overhead of 1.4 ms. Furthermore, software-based cryptography introduces an initial computing delay of 0.74 ms for key scheduling and an additional computing delay of 1.93 ms for each encrypted/decrypted block (Section 4.1.3). It follows that performance indicators reported by Chen et al. in [5] result about one order of magnitude larger than ours in the case of software-based cryptography and two orders of magnitude larger in the case of hardware-based cryptography (see Section 4). Finally, Chen et al.'s analysis is only based on simulation without any experimental validation of the results. The only measurements account for the cost of software cryptography but they come from a previous paper [25]. In contrast, we present an analytical model, an extended simulator, and a set of experiments on real sensor nodes validating both the model and the simulation results.

A preliminary performance evaluation of the IEEE 802.15.4 security sublayer was presented by the authors in [28]. The present work largely extends the previous workshop paper [28] from several standpoints. First, this paper completes and integrates the experimental evaluation in [28] by also considering latency and per-packet energy consumption. Furthermore, this paper considers

both hardware-based and software-based encryption whereas [28] only considers hardware-based encryption. Finally, this paper presents and validates both an analytical and a simulation cost model whereas [28] only focuses on an experimental evaluation.

Using security mechanisms requires establishing the cryptographic keys to be used by the encryption algorithms. However, the IEEE 802.15.4 security sublayer does not specify any key establishment scheme and, for this reason, we will not discuss this issue any further in the rest of the chapter. Notwithstanding, it is important to notice here that, due to the limited resources and the large scale of WSNs, the key management scheme for desktop- and server-computing are generally not suitable. Therefore, key management and its performance in WSNs has become a very active research topic [29,30]. Many key management schemes have been proposed and evaluated, that are ready to use in IEEE 802.15.4 [13,29,31,32]. Relevant examples are [14,33–35].

Finally, we would like to spend a comment on [36]. TinySec is not compliant with IEEE 802.15.4. Actually, it can be considered an alternative solution to link-level security. However, from a performance point of view, Karloff et al. achieve similar conclusions as ours. Namely, much of the overhead can be fully explained by the increased packet length and additional computations that security imposes.

3. IEEE 802.15.4: an overview

In this section we provide an overview of the IEEE 802.15.4 standard, with a focus on the CSMA/CA network multiple access protocol. The reader may refer to the standard [1] for further details.

IEEE 802.15.4 is a standard for low-rate, low-power *Personal Area Networks* (*PANs*). The standard defines two different types of device, namely *Reduced-Function Devices* (*RFDs*) and *Full-Function Devices* (*FFDs*). RFDs are intended to perform simple operations and typically feature minimal resources in terms of memory, storage and processing capabilities. In contrast, FFDs may have more resources and can fulfill network management tasks. A device may play one of the following roles: *ordinary device*, *coordinator*, or *PAN coordinator*. An RFD can only be an ordinary device, whereas an FFD can play any role. A network may have one or more coordinators but only one PAN coordinator that is selected among the coordinators. A coordinator is responsible to manage a subset of ordinary nodes by relaying messages among them. In order to communicate, ordinary nodes must associate with a coordinator. IEEE 802.15.4 supports two network topologies, namely *star*, and *peer to peer*. The former one is single-hop, whereas the latter is multi-hop. Also, the standard defines two channel access modes, namely, *beacon-enabled* and *nonbeacon-enabled*. In the beacon-enabled mode, the PAN coordinator periodically broadcasts beacon frames to synchronize channel access. In the nonbeacon-enabled mode, coordinators do not emit beacon frames and devices transmit frames without waiting for beacons. In this paper we focus on the beacon-enabled mode.

With reference to Fig. 1, in the beacon-enabled mode, two consecutive beacons bound a *superframe*. A superframe is divided into

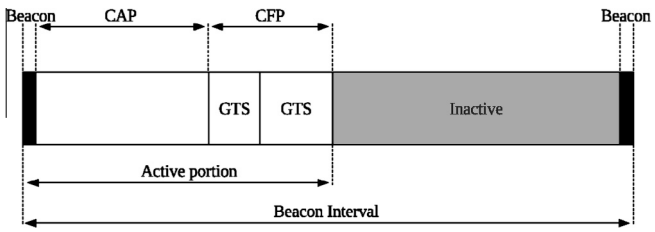


Fig. 1. Structure of a superframe.

superframe slots whose duration is 320 μs. All operations are slot-aligned. A superframe has an *active portion* and an optional *inactive portion*. The PAN coordinator can switch to low-power mode during the inactive portion.

The active portion of a superframe may be divided in two periods, the *Contention Access Period* (*CAP*) and, optionally, the *Contention Free Period* (*CFP*). The Contention Access Period starts immediately after the beacon. The Contention Free Period, if present, goes from the end of the Contention Access Period to the end of the active portion. The Contention Free Period consists in a collection of *Guaranteed Time Slots* (*GTSs*) that are allocated by the PAN coordinator to requesting devices in order to let them access the medium without contention. In this paper we will focus on the Contention Access Period.

In the Contention Access Period sensor nodes use the *Carrier Sense Multiple Access with Collision Avoidance* (*CSMA/CA*) protocol to access the shared communication medium and avoid collisions. The access protocol is organized in *backoff stages*. Initially, a sensor node waits for a random *backoff interval*, which is a time interval multiple of the superframe slot. At the end of this waiting, the sensor node performs two consecutive Clear Channel Assessment (CCA) operations, to ascertain that the channel is free. If the channel is found busy at least once, the sensor node starts another backoff stage with a longer backoff period (if the maximum allowed number of backoff stages is exceeded the frame is dropped). Specifically, the backoff window is doubled at each backoff stage, unless the maximum allowed value has been reached. On the contrary, if the channel results free twice, the sensor node sends the data frame and waits for the related ACK frame. Upon receiving a frame correctly, the recipient replies with an ACK without contention. If the ACK is not received within a predefined time interval, the sender retransmits the data frame (unless the maximum number of retransmissions has been exceeded).

3.1. IEEE 802.15.4 security sublayer

The IEEE 802.15.4 security sublayer optionally provides link-level security services to the higher layers. In general, link-level security secures the wireless link and allows applications to function at least as securely as they would do over a wired network. It follows that link-level security allows a *seamless* integration of wireless networks into existing wired networks and provides the greatest ease of deployment among currently available network cryptographic approaches [37]. Furthermore, specifically in a WSN, link-layer security supports in-network processing, passive participation and local broadcast to save traffic and reduce energy [36,38]. The two other likely alternatives, namely end-to-end security at the application layer and end-to-end security at the transport layer, provide a high level of security, but require a complex setup of cryptographic keys, and neither guarantee seamless integration nor support in-network processing, passive participation and local broadcast. Of course, link-level security and end-to-end security mechanisms can co-exist. Security at multiple places in the protocol stack is not considered harmful and constitutes a means to respond to demand for more security with yet more sophisticated use of cryptography [37,38].

The IEEE 802.15.4 security sublayer guarantees data confidentiality, data authenticity and replay detection on a per-frame basis. ACK frames are not secured. A frame can be secured according to *security levels*. Specifically, three different security levels are defined: the *CTR* security level provides confidentiality; the *CBC-MAC* security level provides authentication and replay detection; and, finally, the *CCM* security level provides authentication and confidentiality. In order to implement the cryptographic transformations required by the security levels, the standard uses the Advanced Encryption Standard (AES) block cipher [24]. AES has a

fixed block size of 128 bits and a variable key size of 128, 192, or 256 bits. IEEE 802.15.4 uses 128-bits keys only.

IEEE 802.15.4 does not define any key establishment schemes, which are entrusted to the higher layers. In practice, the standard assumes that both senders and recipients pre-share common security settings and store the needed security material before secure communications can actually take place. However, IEEE 802.15.4 provides four *Key Identifier Modes* to identify and retrieve a cryptographic key to secure/unsecure a frame.

An unsecured frame is composed of three fields, namely a *MAC Header* (7–23 bytes), and a variable length *Payload* (0–118 bytes) and a *Frame Check Sequence* (*FCS*, 2 bytes). A secured frame contains an additional header called the *Auxiliary Security Header* (*ASH*), and, if the security level includes authentication, the *Message Integrity Code* (*MIC*). The ASH carries the information required for security processing and frame securing and unsecuring. In a secured frame, the ASH is placed next to the standard MAC header (Fig. 2). The ASH is a 5–14 byte data structure composed of three fields: (i) the *Security Control Header* (1 byte) which specifies the security level (3-bits *SecLevel* sub-field) and the *Key Identifier Mode* (2-bits *KeyIdMode* sub-field); (ii) the *Frame Counter* (4 bytes) for the anti-replay service; and, finally, (iii) the *Key Identifier Field* (0–9 bytes) that contains information to identify the key to unsecure a frame. The Auxiliary Security Header (ASH) is transmitted in the clear but it can be authenticated as described in the following.

Security levels are depicted in Fig. 3. The CTR security level secures a frame by encrypting its payload in the counter mode (Fig. 3(a)). As a rule of thumb, the CTR security level requires a block cipher encryption operation for each block to encrypt. The CBC-MAC security level secures a frame by authenticating the frame header, the ASH, and the payload (Fig. 3(b)). The CBC-MAC security level initially computes a 128-bit Message Integrity Code (MIC) by using the AES block cipher in the cipher-block-chaining mode. Then, the MIC is truncated and appended to the frame. The MIC can be truncated at 4, 8 or 16 bytes, so leading to three variations of CBC-MAC of increasing security, namely CBC-MAC-4, CBC-MAC-8, and CBC-MAC-16, respectively. As a rule of thumb, the CBC-MAC security level requires a block cipher encryption operation for each block to authenticate. Finally, the CCM security level secures a frame by using the AES block cipher in the counter with CBC-MAC mode (Fig. 3(c)). The CCM security level initially authenticates the frame header, the ASH, and the payload as in the CBC-MAC security level. Like the CBC-MAC security level, the MIC can be truncated at 4, 8, or 16 bytes so producing three variations of the CCM of increasing security, namely CCM-4, CCM-8, and CCM-16, respectively. Finally, CCM security level encrypts the resulting MIC and the payload in the counter mode. The CCM security level requires one block cipher encryption operation for each block of encrypted or authenticated fields (i.e. frame header, ASH and MIC) and two encryption operations for the payload, that is both authenticated and encrypted.

Table 1 gives an overview of the available security levels. For each security level, the table specifies the security services it provides (i.e. "Confidentiality", "Authentication", and "Replay detection"). If a security level introduces a MIC, column "MIC size" specifies the corresponding length in bytes.

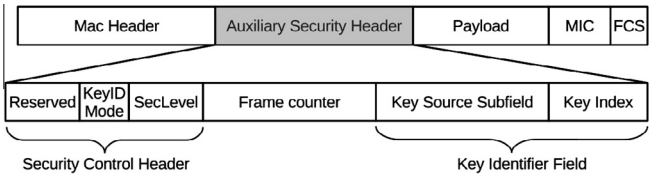


Fig. 2. Auxiliary Security Header.

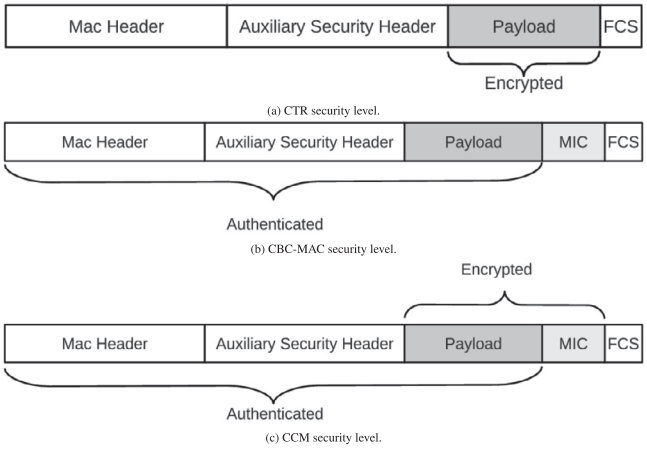


Fig. 3. Security levels.

Table 1
Security levels.

Security mode	Data confidentiality	Data authenticity	Replay detection	MIC size (bytes)
CTR	ON	OFF	ON	–
CBC-MAC-4	OFF	ON	ON	4
CBC-MAC-8	OFF	ON	ON	8
CBC-MAC-16	OFF	ON	ON	16
CCM-4	ON	ON	ON	4
CCM-8	ON	ON	ON	8
CCM-16	ON	ON	ON	16

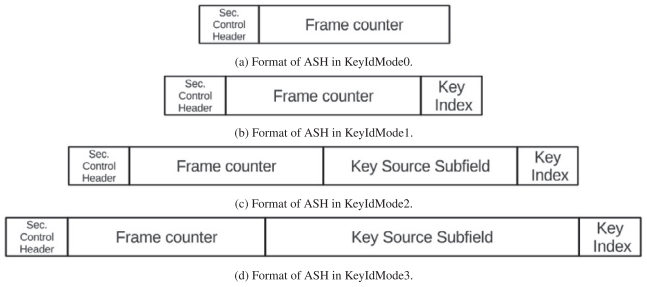


Fig. 4. Format of ASH as a function of the key identifier mode.

Table 2
Size of the ASH as a function of the KeyIdMode.

KeyIdMode	ASH size (bytes)
0	5
1	6
2	10
3	14

Fig. 4 shows the format of the ASH depending on the key identifier mode. In the case of Key Identifier Mode 0 (KeyIdMode0), the ASH does not include any Key Identifier Field and security operations rely on a pre-shared static default key (Fig. 4(a)). In the case of Key Identifier Mode 1 (Fig. 4(b)), the Key Identifier Field contains the Key Index sub-field only (1 byte). In the case of Key Identifier Modes 2 and 3 (KeyIdMode2 and KeyIdMode3), the Key Identifier Field contains both the Key Index and Key Source subfields. The Key Source Subfield is four bytes in the KeyIdMode2 (Fig. 4(c)) and eight bytes in the KeyIdMode3 (Fig. 4(d)).

Table 2 reports the size of the Auxiliary Security Header (ASH) as a function of the key identifier mode.

3.2. Security operations

The standard specifies a number of *security operations*, namely the *security procedures* and *sub-procedures*. A thorough and detailed description of these operations is beyond the scope of this paper (the interested reader may directly refer to the standard [1]). However, in this section, we give a very concise description of the operations in order to convey the intuition of the computations they carry out and the computing overhead they imply. In particular, we highlight that security operations involve not only *cryptographic transformations* but also *management operations*, such as frame parsing and data structures lookups.

The standard considers two main security procedures, the *outgoing frame security procedure*, performed on the sending side upon frame transmission, and the *incoming frame security procedure*, performed on the receiving side upon frame reception. These procedures exploit two main data structures, the *Key Table* and the *Device Table*. The Key Table stores the cryptographic keys used by the node as well as information about the usage of these keys. Typically, the Key Table is accessed using the pair (Key Source, Key Index) as search key to retrieve the cryptographic key identified by such a pair, the list of nodes using such a key, and the types of frames (beacon, data, command) to be protected by means of such a key. The Device Table records the devices with which the node is communicating. Typically, the Device Table is accessed using the device identifier as search key to retrieve the last value of the frame counter received from that device.

The outgoing frame security procedure receives the unsecured frame, the security level, the key identifier mode, the Key Source and the Key Index as input parameters, and secures such a frame as specified by the security level, using the key identified by the pair (Key Source, Key Index) according to the key identifier mode. If the procedure succeeds, the resulting secured frame is returned for transmission. Notice that securing the frame consists in applying to the unsecured frame the cryptographic functions specified by the security level.

The incoming frame security procedure receives the secured frame and, initially, parses it and determines the values of the security level, the key identifier mode, the Key Source and the Key Index as specified in the Auxiliary Security Header. Then, the procedure unsecures the frame, as specified by the security level, using the key identified by the pair (Key Source, Key Index) according to the key identifier mode. If the procedure succeeds, the resulting unsecured frame is returned for reception. Notice that unsecuring a frame also requires checking whether the received frame is a replay or not. The procedure accomplishes this check by accessing the Device Table specifying the sending node identifier as search key, retrieving the corresponding frame counter field value, and ascertaining that this value is smaller than that contained in the secured frame.

3.3. The CONET open implementation of IEEE 802.15.4

We have implemented a complete and fully operational version of the standard security sublayer within an open-source implementation of IEEE 802.15.4 maintained by the TinyOS IEEE 802.15.4 Working Group [39]. The whole standard, including the security sublayer, has been implemented [19] in the nesC language for the TinyOS operating system on the Tmote Sky platform equipped with the CC2420 chipset.[1] The security sublayer implementation can be downloaded from [18]. To the best of our knowledge, this is the first available free implementation of IEEE 802.15.4 including security services. All the experimental evaluations reported in this paper have been carried out on this implementation.

[1] This activity has been carried out within the framework of the European Network of Excellence called CONET (http://www.cooperating-objects.eu/).

4. Evaluation

In the presence of security, the network experiences performance degradation due to two sources of overhead, namely the *communication overhead* and the *processing overhead*. The *communication overhead* is due to the extra bits that are transmitted due to security, namely, the ASH and the MIC field (if present). The processing overhead is due to the extra processing due to security procedures including parsing the ASH, looking up into tables as required by the standard procedures, and applying the cryptographic algorithms to secure/unsecure frames.

In order to quantify the impact of communication and processing overhead, we consider the following performance indices:

- *Latency* (τ), defined as the interval of time between the instant at which the source node starts the frame transmission and the instant at which the same node receives the corresponding ACK.
- *Goodput* (G), defined as the amount of useful information bits correctly received by the PAN coordinator per unit of time.
- *Per-packet energy consumption* (ϵ), defined as the total energy consumed by each sensor node divided by the number of data frames correctly delivered to the PAN coordinator.

In the goodput definition we consider only the payload and not the whole frame in order to underline the impact of the security overhead on transmission of the useful information carried by a MAC frame. The size of the payload field is always the same, irrespectively of the security level used. As a consequence, goodput decreases when security increases. This effect will be quantified in the next sections.

4.1. Analysis

In this section we evaluate analytically the impact of security services on the performance indices defined above. To this end, we consider a very simple network consisting of only two nodes, the PAN coordinator and a sensor node. In this setting, the sensor node always succeeds in accessing the wireless medium at the first attempt. This allows us to better understand the impact of security on performance.

4.1.1. Latency and goodput

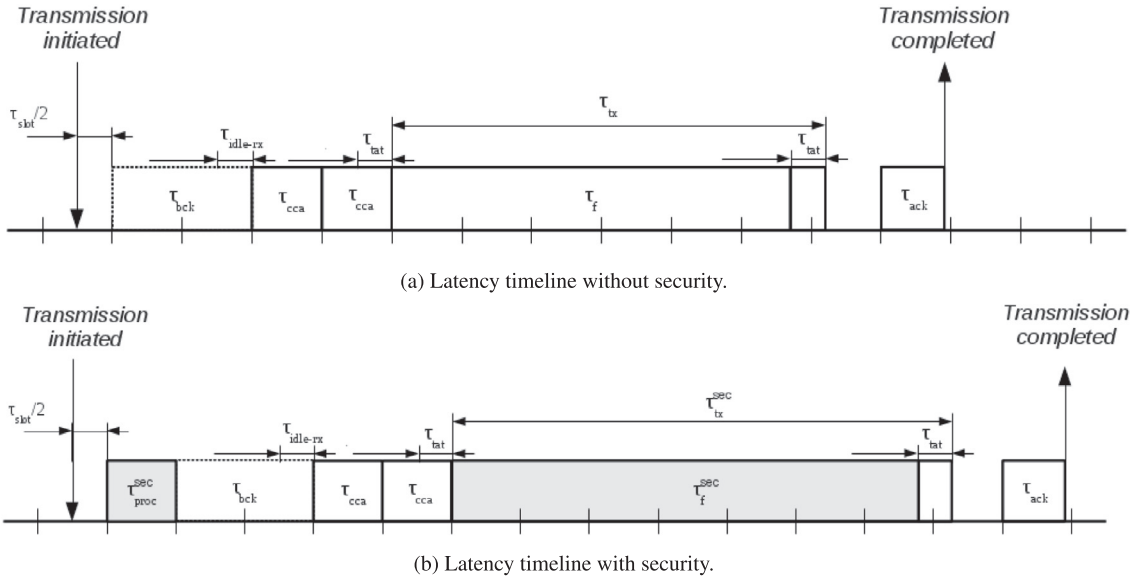
In order to model the impact of security on latency, we first define latency in the absence of security and, then, we add the effects of security. The average latency experienced by a frame consists of a number of components corresponding to the different steps of the CSMA/CA algorithm (see Section 3). As shown in Fig. 5(a), assuming that the sensor node starts in the idle state, latency can be computed as:

$$\tau = \frac{\tau_{\text{slot}}}{2} + \tau_{\text{bck}} + 2 \cdot \tau_{\text{cca}} + \tau_{\text{tx}} + \tau_{\text{ack}} \tag{1}$$

where

$$\tau_{\text{tx}} = \left\lceil \frac{\tau_{\text{f}} + \tau_{\text{tat}}}{\tau_{\text{slot}}} \right\rceil \cdot \tau_{\text{slot}} \tag{2}$$

In Eq. (1), $\frac{\tau_{\text{slot}}}{2}$ accounts for an average delay deriving from the fact that operations are aligned to a backoff slot, whose duration is equal to τ_{slot}; τ_{bck} accounts for the random backoff time, which includes $\tau_{\text{idle-rx}}$, the time necessary to switch the radio from the idle state to the receiving state; $2 \cdot \tau_{\text{cca}}$ accounts for the time necessary to perform two consecutive Clear Channel Assessment operations;



(a) Latency timeline without security.

(b) Latency timeline with security.

Fig. 5. Slotted CSMA/CA timeline (a) without security and (b) with security.

τ_{tx} accounts for the total time required to actually transmit a frame; and, finally, τ_{ack} is the time to receive the corresponding ACK frame. In its turn, τ_{tx} is equal to a whole number of backoff slots that contain the time interval $\tau_f + \tau_{tat}$ (see Eq. (2)), namely the *frame transmission time* τ_f to actually transmit a frame, and the *turnaround time* τ_{tat} to switch the radio from transmission mode to reception (and thus become able to receive the ACK frame). The *turnaround time* τ_{tat} to switch the radio from receive mode to transmission mode is part of the second τ_{cca} time interval.

Security brings in two latency contributions: the *security processing time* τ^{sec}_{proc}, which accounts for the security processing overhead, and the *security communication time* τ^{sec}_{comm}, which accounts for the security communication overhead. The security processing time τ^{sec}_{proc} accounts for the time required by security operations. The security communication time τ^{sec}_{comm} accounts for the time necessary to transmit the additional fields brought about by security, namely the ASH and the MIC field (when present). The communication time τ^{sec}_{comm} has to be added to the frame transmission time τ_f. With reference to Fig. 5(b), it follows that Eq. (1) becomes:

$$\tau^{sec} = \tau^{sec}_{proc} + \frac{\tau_{slot}}{2} + \tau_{bck} + 2 \cdot \tau_{cca} + \tau^{sec}_{tx} + \tau_{ack} \tag{3}$$

where

$$\tau^{sec}_{tx} = \left\lceil \frac{\tau_f + \tau^{sec}_{comm} + \tau_{tat}}{\tau_{slot}} \right\rceil \cdot \tau_{slot} \tag{4}$$

Once we have derived analytical formulas without and with security, we can easily calculate the goodput G experienced in both cases. Assuming that the sensor node has always a frame ready for transmission, the pattern shown in Fig. 5 repeats for each following frame transmission. Hence:

$$G = \frac{P}{\tau} \tag{5}$$

and

$$G = \frac{P}{\tau^{sec}} \tag{6}$$

4.1.2. Per-packet energy consumption

Since we are assuming a network scenario with only two nodes and an ideal communication channel, the PAN coordinator receives all transmitted frames correctly. In addition, the transmission pattern for all frames is the same as the one shown in Fig. 5. Hence, in order to derive the per-packet energy consumption we can refer to a single frame transmission. Specifically, we sum the energy expenditures in every time interval contributing to latency (see Eq. (3)). The energy ϵ consumed in an interval is the product of the power w consumed in τ and time interval τ itself, i.e. $\epsilon = w \cdot \tau$. Power consumption can be derived from the device datasheet.

In order to evaluate the per-packet energy consumption, we observe from Section 3.2 that the processing overhead τ^{sec}_{proc} can be split into two components, namely the *management overhead*, τ^{sec}_{mgmt}, that accounts for frame parsing and tables lookup, and the *encryption overhead*, τ^{sec}_{crypto}, that accounts for applying cryptographic algorithms to frames. The former component is implemented in software on the sensor node microcontroller. The latter component can be implemented both in software on the sensor node microcontroller or in hardware on the radio chipset, provided this device offers hardware support to cryptography. The CC2420 radio chipset available on Tmote Sky sensor nodes provides such a support [26].

Whether cryptography is hardware-based (hw-based) or software-based (sw-based) may have a strong impact on performance for two reasons. Hardware-based encryption is faster than software-based encryption. On the other hand, hardware-based encryption is performed on the communication device that, generally, has larger power consumption than the microcontroller. In the rest of this paper we will evaluate performance in both cases.

Furthermore, whether cryptography is hw-based or sw-based also influences the granularity at which we are able to evaluate parameter τ^{sec}_{crypto}. The AES algorithm consists of a key scheduling algorithm and an encryption (decryption) algorithm. Key scheduling is performed just once, before encryption (decryption) starts, whereas the encryption (decryption) algorithm is performed on each plaintext (ciphertext) block. In the sw-based cryptography case, by properly instrumenting implementation, it is possible to separate the key scheduling overhead ($\tau^{sec}_{key\ sw}$) from the per-block encryption (decryption) algorithm overhead ($\tau^{sec}_{block\ sw}$). In contrast, in the hw-based cryptography case this is not possible. It follows that the encryption processing overhead τ^{sec}_{crypto} will be expressed in terms of a single parameter $\tau^{sec}_{crypto\ hw}$ in the hw-based cryptography. In contrast, the encryption processing overhead $\tau^{sec}_{crypto\ sw}$ in sw-based cryptography will be expressed in terms of two parameters, $\tau^{sec}_{key\ sw}$ and $\tau^{sec}_{block\ sw}$.

4.1.3. Evaluation of parameters

Table 3 shows the parameters values for calculating Eq. 3, assuming that the communication chipset is CC2420 [26] and the microcontroller is MSP430 [40]. The values of absorbed current referring to MSP430 and CC2420 are taken from the respective datasheets [40,26]. The only exception is the value of the absorbed current during $\tau^{sec}_{crypto\ hw}$ that has been taken from [11]. The absorbed current during $\tau_{idle\text{-}rx}$ has been obtained by averaging the current absorbed in the idle state and the current absorbed in the receiving state. The current absorbed during turnaround time τ_{tat} has been estimated analogously (i.e., the mean value between the current absorbed in the receiving state and the current absorbed in the transmitting state). Please note that the approach we used to evaluate these currents is the same used by the Ns2 simulator to evaluate energy consumption [31,41].

The duration of all delay components shown in Table 3 are derived from the standard, except for the values of τ^{sec}_{mgmt}, $\tau^{sec}_{crypto\ hw}$, $\tau^{sec}_{key\ sw}$, and $\tau^{sec}_{block\ sw}$, that have been evaluated experimentally. Specifically, to measure these delays, we used two timers and properly instrumented our implementation of the standard (see Section 3.3). For the sw-based cryptography case, we used the software implementation of AES-128 algorithm that is available in the TinyOS repository [27]. In all cases, we fixed KeyIdMode3 and considered three different payload sizes, i.e., 2, 18, and 80 bytes. We measured the parameters for all possible combinations of security levels and payload sizes. For each measurement, we run an experiment consisting in sending 100 frames and taking the average. Each experiment was repeated 10 times, in order to assure a better accuracy and measure the standard deviation.

It is worthwhile to notice that time τ^{sec}_{mgmt} (260.61 ± 0.53 μs) accounts for the management overhead due to frame parsing and table lookups. This overhead is equal for both sw-based and hw-based cryptography and is independent of the frame size and the security level. Furthermore, in the case of hw-based cryptography, we found that, in practice, $\tau^{sec}_{crypto\ hw}$ (1393 μs), is influenced by neither the security level nor the payload size. In principle, $\tau^{sec}_{crypto\ hw}$ would depend on these parameters, which determine the actual number of blocks to be encrypted and/or authenticated. However, hw-based cryptography is so fast that its overhead is masked by the time necessary for registers setup and device strobing. Finally, in sw-based cryptography, the key scheduling overhead $\tau^{sec}_{key\ sw}$ and the per-block encryption overhead $\tau^{sec}_{block\ sw}$ are not negligible and account to 740 μs and 1630 μs, respectively. It follows that, in contrast to hw-based cryptography, $\tau^{sec}_{crypto\ sw}$ now greatly depends on both the payload size and the security level.

Table 4 shows the frame expansion (in bytes) as a function of the security level and the key identifier mode. Such an expansion is due to the ASH and the MIC, if present. The size of the former depends on the KeyIdMode (see Section 3.1) whereas the size of the latter depends on the security level (see Section 3.1).

Table 4
Frame expansion due to security. Values are in bytes.

	CTR	CBC-MAC-4 or CCM-4	CBC-MAC-8 or CCM-8	CBC-MAC-16 or CCM-16
KeyIdMode0	5	9	13	21
KeyIdMode1	6	10	14	22
KeyIdMode2	10	14	18	26
KeyIdMode3	14	18	22	30

4.1.4. Analytical results

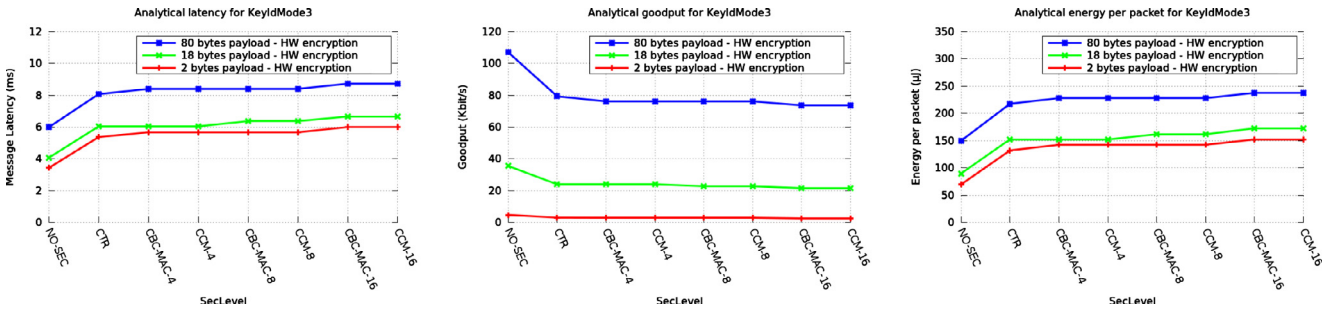
In this section we show the trends of latency, goodput and energy consumption as functions of the security level. In this analysis, we consider the KeyIdMode3 that, for each security level, causes the largest ASH, therefore the largest frame expansion and thus represents the worst case from the communication viewpoint. We evaluate the trends in the case of both hw-based and sw-based cryptography for three different values of the payload, namely 2 bytes, which features a *small payload*; 18 bytes, which features a *realistic payload*; and, finally, 80 bytes, which features the *largest payload* when the MIC and ASH have the largest size.

Fig. 6 shows the trend of latency, goodput and per-packet energy consumption with the security levels for different payload sizes in KeyIdMode3, when using hw-based cryptography. As it turns out, the main performance penalty occurs when we move from unsecured (NO-SEC) to secured traffic. However, a variation of the security level causes little, almost negligible, variations in the security cost. Consider latency for example. Switching from NO-SEC to CTR, causes latency to increase by the 57% in the case of 2-bytes payload, 49% in the case of 18-bytes payload, and 35% in the case of 80-bytes payload. However, switching from CTR to CCM-16 causes just a latency increase of 12%, 11%, and 8%, respectively. As to goodput, switching from NO-SEC to CTR causes a decrement of 36% in the case of 2-bytes payload, 33% in the case of 18-bytes payload, and 26% in that of 80-bytes payload. However, switching from CTR to CCM-16 causes a further decrement of just 11% in the case of 2-bytes payload, 10% in the case of 18-bytes payload, and 7% in that of 80-bytes payload. Finally, as to energy consumption, switching from NO-SEC to CTR causes an increment of 89% in the case of 2-bytes payload, 71% in the case of 18-bytes payload, and 45% in that of 80-bytes payload. However, switching from CTR to CCM-16 causes a further increment of just 15% in the case of 2-bytes payload, 13% in the case of 18-bytes payload, and 5% in that of 80-bytes payload.

It is interesting to observe that, in some cases, a change in the security level that causes a frame size increment does not reflect in a latency increase. For instance, consider the 80-bytes payload curve. Switching from CCM-4 (CBC-MAC-4) to CCM-8 (CBC-MAC-8) does not cause any latency change even though the latter implies transmitting 4 bytes more than the former. This is because the increase in the transmission time due to frame size increment

Table 3
Parameters.

Device	Parameter	Duration (μs)	Current (mA)	Power consumption (mW)	Energy consumption (μJ)
MSP430 $v = 1.8$ V	Security management overhead (τ^{sec}_{mgmt})	260	0.6	1.08	0.28
	SW-based key scheduling overhead ($\tau^{sec}_{key\ sw}$)	740	0.6	1.08	0.80
	SW-based per-block cryptography overhead ($\tau^{sec}_{block\ sw}$)	1630	0.6	1.08	1.76
CC2420 $v = 1.8$ V	Total HW-based encryption overhead ($\tau^{sec}_{crypto\ hw}$)	1393	21.27 [11]	38.14	53.13
	Average backoff period (τ_{bck})	1120	0.427	0.77	0.86
	Slot duration (τ_{slot})	320	0.427	0.77	0.25
	Idle-rx switching ($\tau_{idle\text{-}rx}$)	192	10.067	18.12	3.48
	Turnaround time (τ_{tat})	192	18.55	33.39	6.41
	Clear Channel Assessment (τ_{cca})	320	19.7	35.46	11.35
	Reception of ACK frame (τ_{ack})	352	19.7	35.46	12.48



(a) latency (b) goodput (c) energy

Fig. 6. Latency, goodput and per-packet energy consumption (hw-based cryptography).

is hidden by the backoff alignment, as expressed by Eq. (4). Similar considerations hold for goodput and energy consumption.

Fig. 7 shows the trend of latency, goodput and per-packet energy consumption with the security levels for different payload sizes in KeyIdMode3, when using sw-based cryptography. Similarly to the previous case (i.e. hw-based cryptography), a performance penalty occurs when we move from unsecured (NO-SEC) to secured traffic. However, in contrast to the previous case, variations in the security level (or payload size) cause considerable variations in the security cost. Actually, as discussed in Section 4.1.3, the security level determines the number of block encryption/decryption operations whose delays, in the case of sw-based cryptography, are not negligible.

For example, switching from NO-SEC to CTR, causes latency to increase considerably by the 86% in the case of 2-bytes payload, 112% in the case of 18-bytes payload, and 158% in the case of 80-bytes payload. However, switching from CTR to CBC-MAC causes latency to increase by about 30% in the case of 2-bytes payload, about 23% in the case of 18-byes payload and, finally, about 24% in the case of 80-bytes payload. Switching from CTR to CCM causes latency to increase by about 80% in the case of 2-bytes payload, about 79% in the case of 18-byes payload and, finally, about 87% in the case of 80-bytes payload.

Goodput has a dual behavior. Switching from NO-SEC to CTR causes a goodput decrement of 46% in the case of 2-bytes payload, 52% in the case of 18-bytes payload, and 61% in that of 80-bytes payload. Goodput further decreases upon switching to CBC-MAC and CCM.

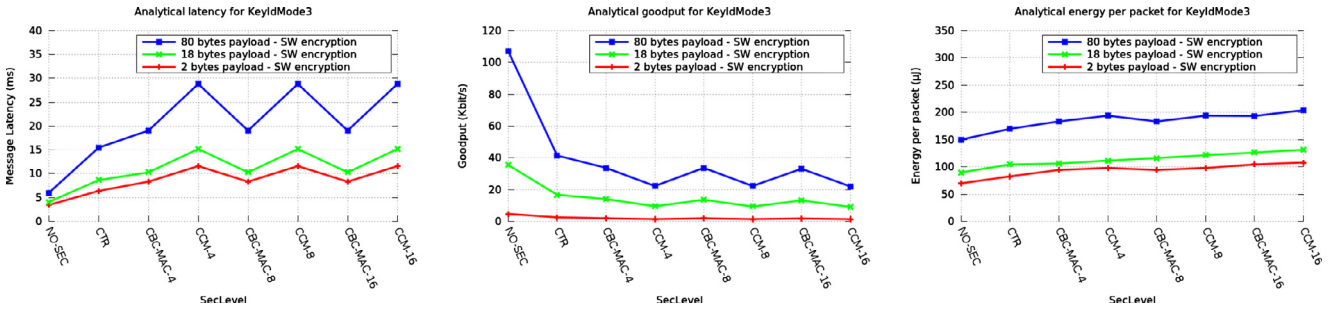
Finally, as to per-packet energy consumption, switching from NO-SEC to CTR causes an increment of 18% in the case of 2-bytes payload, 16% in the case of 18-bytes payload, and 13% in that of 80-bytes payload. Furthermore, switching from CTR to CCM-16 causes a further increment of 15% in the case of 2-bytes payload, 13% in the case of 18-bytes payload, and, finally, 9% in the case of 80-bytes payload.

It turns out that the per-packet energy consumption is the only metric that improves upon moving from hw-based to sw-based cryptography. For instance, if we consider the CCM-16 security level, latency increases by 44% in the case of 2-bytes payload, 137% in the case of 18-bytes payload, and 235% in the case of 80-bytes payload. Consistently, goodput decreases by 50%, 58%, and 70%, respectively. In contrast, per-packet energy increases by 29%, 24%, and 14%, respectively. The reason is that, while performing cryptographic operations, MSP430 absorbs much less power than CC2420. Actually, from Table 3 it turns out that both devices operate at 1.08 V but MSP430 absorbs 0.6 mA, whereas CC2420 absorbs 21.19 mA, i.e. a current, and thus a power that is about 35 times larger than the former. As a consequence, even though sw-based cryptography is slower than hw-based cryptography, the overall energy consumed by the former is smaller than that consumed by the latter.

4.1.5. Experimental validation of the analytical model

The analytical model has been validated through experimental measurements on a real testbed. The experimental testbed consisted of Tmote Sky sensor nodes [40], equipped with an MSP430 microcontroller, 10 KB of RAM, 48 KB of ROM and, finally, a CC2420 radio transceiver. CC2420 is compliant with the IEEE 802.15.4 physical layer and supports a 250 Kbit/s bit rate over an unlicensed 2.4 GHz ISM band [26]. As to system software, sensor nodes run the TinyOS 2.x operating system (available from [42]) and the CONET open-source implementation of IEEE 802.15.4 (see Section 3.3). To validate the analytical results derived in previous section, we considered only two sensor nodes, KeyIdMode3 and a payload size equal to 18 bytes.

Tables 5 and 6 show the analytical and experimental values of latency and goodput, for different security levels, when using hw-based and sw-based cryptography, respectively. The experimental measurements are fully consistent with the analytical results. Furthermore, they completely confirm the trend we have



(a) latency (b) goodput (c) energy

Fig. 7. Latency, goodput and per-packet energy consumption (sw-based cryptography).

Table 5
Experimental vs analytical latency. Values are in ms.

SecLevel	Experimental (HW-based cryptography)	Analytical (HW-based cryptography)	Experimental (SW-based cryptography)	Analytical (SW-based cryptography)
NO-SEC	4.28 (±0.14)	4.06	4.28 (±0.14)	4.06
CTR	6.35 (±0.13)	6.04	9.08 (±0.16)	8.64
CBC-MAC-4	6.57 (±0.18)	6.04	10.83 (±0.16)	10.27
CCM-4	6.51 (±0.17)	6.04	16.51 (±0.16)	15.16
CBC-MAC-8	6.62 (±0.13)	6.36	11.25 (±0.14)	10.59
CCM-8	6.79 (±0.22)	6.36	16.62 (±0.16)	15.48
CBC-MAC-16	6.95 (±0.22)	6.68	11.43 (±0.16)	10.91
CCM-16	6.99 (±0.20)	6.68	16.75 (±0.17)	15.80

Table 6
Experimental vs analytical goodput. Values are in Kbit/s.

SecLevel	Experimental (HW-based cryptography)	Analytical (HW-based cryptography)	Experimental (SW-based cryptography)	Analytical (SW-based cryptography)
NO-SEC	33.62 (±1.1)	35.43	33.62 (±1.1)	35.43
CTR	22.69 (±0.47)	23.85	15.86 (±0.26)	16.66
CBC-MAC-4	21.90 (±0.59)	23.85	13.30 (±0.20)	14.02
CCM-4	22.12 (±0.58)	23.85	8.72 (±0.13)	9.50
CBC-MAC-8	21.77 (±0.43)	22.65	12.80 (±0.15)	13.59
CCM-8	21.20 (±0.69)	22.65	8.67 (±0.14)	9.30
CBC-MAC-16	20.71 (±0.63)	21.57	12.60 (±0.18)	13.19
CCM-16	20.60 (±0.58)	21.57	8.60 (±0.09)	9.11

already observed in Section 4.1.4. As far as hw-based cryptography, a significant variation in performance occurs when we proceed from unsecured to secured frames. However, the security level has little, if not negligible, influence on performance. When using sw-based cryptography, the performance loss is greater than in the case of hw-based cryptography and strongly depends on the number of block encryption operations and thus, ultimately, on the payload size and the security level.

4.2. Simulation analysis

In the previous analysis, we have considered a network composed of two nodes. This allows us to understand the impact of security when there is no contention between sensor nodes. In this section we consider a more complex but more realistic network composed by more nodes.

We consider a star, beacon-enabled PAN composed of a coordinator and a variable number of ordinary sensor nodes that are placed in a circle around the sink node, 10 m far from it. Upon receiving a beacon frame, an ordinary node attempts, until it succeeds, to transmit a frame to the coordinator. The Beacon Interval is 983.04 ms (BO = 6 and SO = 6).

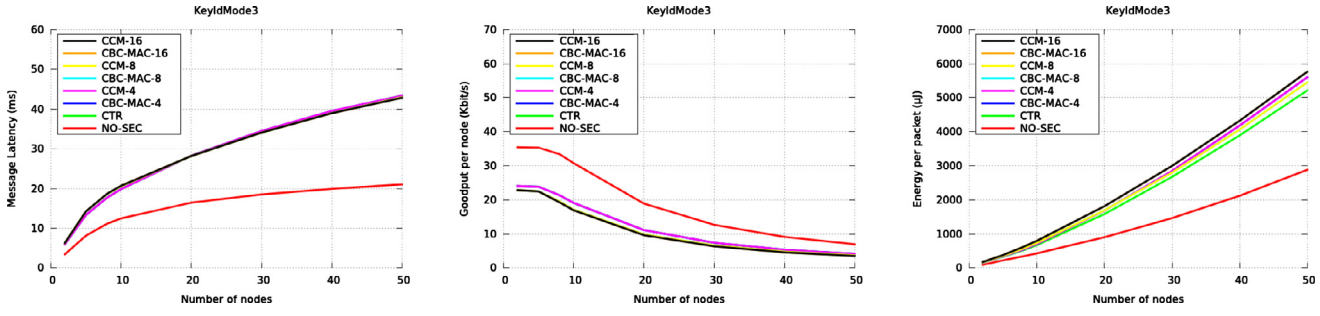
In order to evaluate the impact of security on performance, we simulated such a network by means of the Ns2 simulation tool [43]. The basic IEEE 802.15.4 simulator has been extended to take into account τ^{sec}_{proc} and τ^{sec}_{comm}. The former was modeled as a pure delay. The latter has been implemented by fictitiously enlarging the payload by a quantity specified in Table 4 for each relevant pair (security level, KeyIdMode). In simulations, we only considered KeyIdMode3. We have set the transmission range to 15 m and the carrier sensing range to 30 m as in [44]. In addition, we have considered an 18-bytes payload corresponding to a total unsecured frame size of 33 bytes. We derived simulation results for both hw-based and sw-based cryptography.

For each simulation, we have performed 10 independent replications, each consisting of 1000 Beacon Intervals each. The presented results are averaged over the ten replications with a 95% confidence level.

As to hw-based cryptography, Fig. 8 shows the simulation trend of latency, goodput, and per-packet energy consumption with the number of nodes for each security level. Confidence intervals are so small that they cannot be graphically appreciated.

As above, we validated our simulation results through experimental measurements. Table 7 compares the simulation and experimental results (and the corresponding confidence intervals), for latency and goodput with two and ten nodes. As it turns out, simulation and experimental results agree with each other.

At first glance, we may observe that, in accordance with the previous analysis, for any given number of nodes, switching from unsecured to secured traffic causes a neat performance loss due



(a) latency (b) goodput (c) energy

Fig. 8. Simulation results for latency, goodput, and energy consumption (hw-based cryptography).

Table 7
Simulative vs. experimental latency and goodput with HW-based encryption.

	SecLevel	Latency		Goodput	
		Experimental latency (ms)	Simulative latency (ms)	Experimental goodput (%)	Simulative goodput (%)
2 Nodes	NO-SEC	4.28 (±0.14)	3.42 (±0.01)	33.62 (±1.1)	35.44 (±0)
	CTR	6.35 (±0.13)	5.98 (±0.01)	22.69 (±0.47)	24.07 (±0)
	CBC-MAC-4	6.57 (±0.18)	5.98 (±0.01)	21.90 (±0.59)	24.07 (±0)
	CCM-4	6.51 (±0.17)	5.98 (±0.01)	22.12 (±0.58)	24.07 (±0)
	CBC-MAC-8	6.62 (±0.13)	6.30 (±0.01)	21.77 (±0.43)	22.84 (±0)
	CCM-8	6.79 (±0.22)	6.30 (±0.01)	21.20 (±0.69)	22.84 (±0)
	CBC-MAC-16	6.95 (±0.22)	6.30 (±0.01)	20.71 (±0.63)	22.84 (±0)
	CCM-16	6.99 (±0.20)	6.30 (±0.01)	20.60 (±0.58)	22.84 (±0)
10 Nodes	NO-SEC	13.12 (±0.14)	12.47 (±0.03)	29.34 (±0.96)	30.76 (±0.05)
	CTR	19.14 (±0.58)	19.84 (±0.03)	21.40 (±0.45)	19.03 (±0.02)
	CBC-MAC-4	19.71 (±0.61)	19.85 (±0.03)	21.44 (±0.57)	19.09 (±0.04)
	CCM-4	19.51 (±0.54)	19.85 (±0.03)	20.66 (±0.54)	19.09 (±0.04)
	CBC-MAC-8	19.43 (±0.43)	20.54 (±0.04)	20.55 (±0.41)	17.12 (±0.02)
	CCM-8	20.10 (±0.37)	20.54 (±0.04)	20.40 (±0.66)	17.12 (±0.02)
	CBC-MAC-16	19.41 (±0.50)	20.69 (±0.05)	20.05 (±0.61)	16.92 (±0.04)
	CCM-16	20.33 (±0.53)	20.69 (±0.05)	18.58 (±0.52)	16.92 (±0.04)

to the security processing and communication overhead. However, the specific security level has little, or even no influence on such a loss. Going into more details, let us consider the trend of latency (Fig. 8(a)). For each security level, latency increases with the number of nodes. This depends on the fact that, when the number of nodes increases, it is more likely that a node attempting to transmit has to wait for the free medium. Also, the probability of collisions increases and, hence, some frames have to be retransmitted. However, it turns out that the latency in the case of secured traffic grows with the number of nodes more quickly than the latency in the case of unsecured traffic. Actually, curves tend to diverge. This depends on the additional delays deriving from the security processing and communication overhead that every transmitting node brings in. Due to this additional delay, *ceteris paribus*, in the case of secured traffic, node experiences a latency longer than in the case of unsecured traffic. Goodput has a dual trend (Fig. 8(b)), with respect to latency.

Similar considerations also apply to the energy consumption per delivered packet (Fig. 8(c)). The increasing trend is more remarkable than latency because not only the total energy consumption increases with the number of sensor nodes, but the percentage of delivered frames decreases, as emphasized by the goodput decrease in Fig. 8(b).

As to sw-based cryptography, Fig. 9 shows the trend of latency, goodput and per-packet energy consumption with the number of nodes for each security level. As above, confidence intervals are so small that they cannot be graphically appreciated.

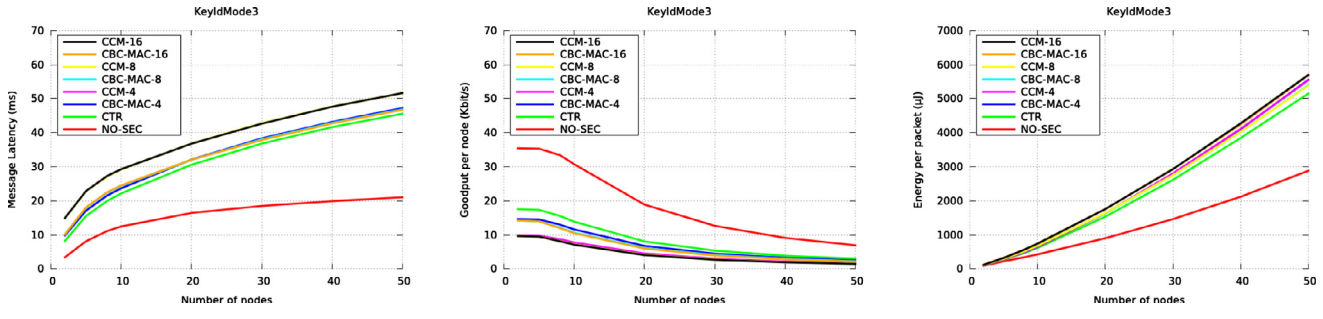
As expected, Fig. 9 shows that switching from unsecured to secured traffic causes a performance loss. Furthermore, figures also show that payload size and security level have influence on such a loss, due to the number of block encryption operations that are required. However, Fig. 9(c) shows that per-packet energy consumption constitutes an exception and its trend is very similar to the hardware-based cryptography (Fig. 8(c)). This is because, with respect to hw-based cryptography, sw-based cryptography increases the encryption processing overhead τ_{crypto}^{sec} but, at the same time, requires a lower power consumption.

As in the previous case, we validated our simulation results through experimental measurements. Again, we observed a general agreement between simulation and experimental results. We omit them for the sake of space.

4.3. Experimental evaluation of memory overhead

In this section we evaluate, through an experimental analysis carried out with the testbed described in Section 4.1.5, the memory overhead introduced by the IEEE 802.15.4 security sublayer.

Fig. 10 shows the ROM footprint breakdown on both the PAN coordinator and a regular sensor node. With hw-based cryptography (Fig. 10(a)), the amount of memory required by the security sublayer executable is the 11.58% of the overall memory available on the PAN coordinator, and the 12.96% on a regular sensor node. In both cases, most of the memory occupancy is due to the IEEE 802.15.4 implementation (i.e. the original communication stack). Note also that the 19.46% (15.44%) of memory on the PAN coordinator (regular node) remains available for other uses (e.g., applications). With sw-based cryptography (Fig. 10(b)), the amount of memory required by the security sublayer executable is the 17.36% of the overall memory available on the PAN coordinator, and the 18.76% on a regular sensor node. In both cases, most of



(a) latency (b) goodput (c) energy

Fig. 9. Simulation results for latency, goodput, and energy consumption (sw-based cryptography).

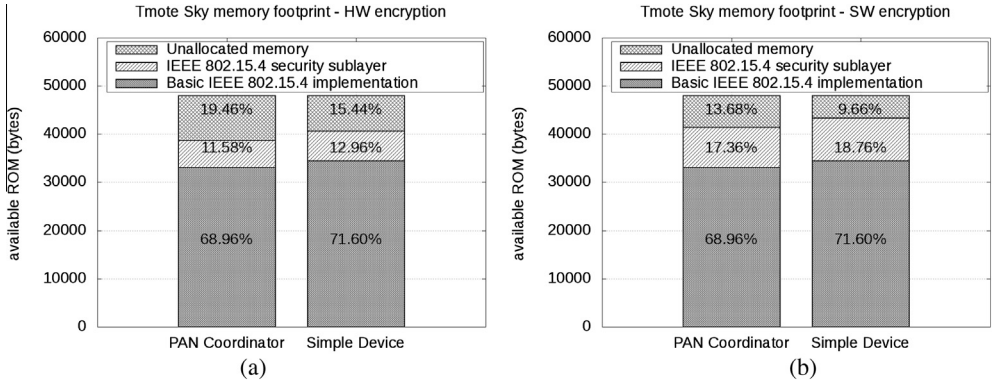


Fig. 10. ROM memory overhead: (a) hw-based cryptography; (b) sw-based cryptography.

the memory occupancy is due to the IEEE 802.15.4 implementation and software-based implementation of the encryption algorithm. Note also that 13.68% (9.66%) of memory on the PAN coordinator (regular node) remains available for other uses (e.g., applications).

However, the space necessary to allocate executable is not the only storage cost that we have to pay in order to use the security sublayer. As discussed in Section 3.2, the security sublayer requires data structures, e.g., the Device Table and the Key Table, that are allocated in RAM and whose size grows with the number of nodes and keys. Fig. 11 shows the trend of RAM occupancy when the number of nodes grows. In our implementation, 9 sender nodes require about 3858 bytes of RAM with hardware encryption. Beyond this threshold, we experimentally observe that motes hang or behave erratically.

In the case of sw-based cryptography, we have to allocate in RAM also the data structures of the AES encryption algorithm, which account for about 1 Kbytes. It follows that the threshold is crossed with a smaller number of nodes, namely four.

With TinyOS/msp430-gcc, there is no limit, but the physical capacity, to the amount of memory that a software component may use. However, it is not recommended to fill up the entire RAM with the component variables, because TinyOS needs space for the stack. There is no straightforward way to calculate the amount of memory TinyOS needs. However, as a rule of thumb, it is better to leave at least 500 byte or 1 KB empty. Otherwise you can get a stack overflow and the mote will hang or do erratic things.

Of course, we cannot exclude that a more efficient implementation than ours may get a greater threshold. However, regardless the actual value of the threshold, the important point to capture here is that in memory scarce devices, the amount of memory necessary for security data structures may constitute a limit to the system scalability.

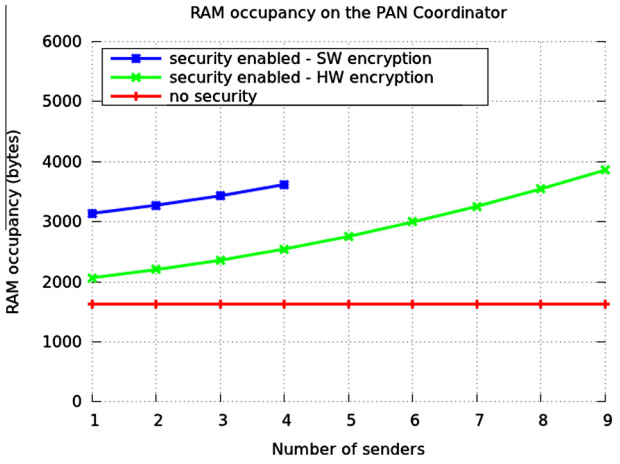


Fig. 11. RAM memory overhead.

5. Conclusion

We have presented a performance evaluation of the IEEE 802.15.4 security sublayer. We have shown that security mechanisms and options, as provided by the standard, cause the increase of frame length (communication overhead) and require additional computations (computing overhead) for security processing. These sources of overhead have an impact on the overall WSN performance in terms of latency, goodput, and memory performance. More precisely, we have obtained the following results. First, we have shown the relationship between the computing and communication overhead and the security parameters, namely security level and key identification mode. In addition, we have shown that the highest cost has to be paid when we switch from unsecured to secured communication. However, when data frames are secured via hardware, the chosen security service has little, or even negligible, impact on performance. In contrast, when traffic is secured via software, both the chosen security service and the payload size have a considerable impact on performance. Differently from previous work [28], we have proposed a simple yet effective analytical model that we have used to extend an Ns2-based simulator of IEEE 802.15.4. The model and the extended simulator have been experimentally validated. Finally, we have evaluated the memory overhead of the security sublayer and, consequently, we have argued that this overhead may pose a fundamental limit to the WSN scalability. We believe that our work can allow designers and implementers to find the best trade-off between security and performance in the application scenario at hand.

We would like to spend a final remark on IEEE Std 802.15.4e, an amendment of IEEE 802.15.4-2011, which adds functionalities to the standard in order to support time constrained applications (e.g. in the industrial domain) and permit compatibility with Chinese WPANs [45]. This amendment potentially impacts our work in two ways. First of all, the amendment introduces optional changes to the MAC model. Second, still optionally, the amendment makes it possible to remove the Frame Counter field as well as increase its size from four to five bytes. However, we point out the following remarks. First, an evaluation of the impact of security on the performance of IEEE 802.15.4e would require an analytical and simulation model of the amended standard. While this is clearly outside the scope of this paper, we claim that the methodology introduced in this paper would remain valid. Second, all the amendments introduced by IEEE Std 802.15.4e are optional. Therefore, our arguments retain their full validity in the default case.

Finally, the IEEE 802.15.4e does not introduce any meaningful change in the security sublayer, but the possible different size of the Frame Counter field. Hence, the security cost model can be extended to encompass these changes as well.

Acknowledgments

This work has been partially supported by PLANET, Platform for the Deployment and Operation of Heterogeneous Networked Cooperating Objects, funded by the European Commission under FP7 with contract number FP7-2009-5-257649 (www.planet-ict.eu) and TENACE, Protecting National Critical Infrastructures From Cyber Threats, funded by the Italian Ministry of Education, University and Research, under the PRIN Framework with contract number 20103P34XC (http://www.dis.uniroma1.it/~tenace/).

We also wish to thank the anonymous reviewers for having helped us to improve our work with their precious comments and advices.

References

[1] IEEE, IEEE Std. 802.15.4-2006, IEEE Standard for Information technology – telecommunications and information exchange between systems – local and metropolitan area networks – specific requirements. Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs), Institute of Electrical and Electronics Engineers Inc., New York, Sep. 2006.
[2] J.S. Lee, Performance evaluation of IEEE 802.15.4 for low-rate wireless personal area networks, IEEE Trans. Consum. Electron. 52 (3) (2006) 742–749.
[3] J. Mišić, S. Shafi, V.B. Mišić, The impact of MAC parameters on the performance of 802.15.4 PAN, Ad Hoc Networks 3 (5) (2005) 509–528.
[4] I. Ramachandran, A.K. Das, S. Roy, Analysis of the contention access period of IEEE 802.15.4 MAC, ACM Trans. Sens. Networks (TOSN) 3 (1) (2007).
[5] F. Chen, Y. Xiaolong, R. German, F. Dressler, Performance impact of and protocol interdependencies of IEEE 802.15.4 security mechanisms, in: Proceedings of the Sixth IEEE International Conference on Mobile Adhoc and Sensor Systems MASS '09, 2009, pp. 1036–1041.
[6] N. Sastry, D. Wagner, Security considerations for IEEE 802.15.4 networks, in: Proceedings of the Third ACM workshop on Wireless security WiSe '04, ACM, New York, NY, USA, 2004, pp. 32–42.
[7] Y. Xiao, H.H. Chen, B. Sun, R. Wang, S. Sethi, MAC security and security overhead analysis in the IEEE 802.15.4 wireless sensor networks, EURASIP J. Wireless Commun. Networking (2006) 81.
[8] C.C. Chang, D.J. Nagel, S. Muftic, Balancing security and energy consumption in wireless sensor networks, in: H. Zhang, S. Olariu, J. Cao, D. Johnson (Eds.), Mobile Ad-Hoc and Sensor Networks, Lecture Notes in Computer Science, vol. 4864, Springer, Berlin, Heidelberg, 2007, pp. 469–480.
[9] G. Guimaraes, E. Souto, D. Sadok, J. Kelner, Evaluation of security mechanisms in wireless sensor networks, in: Proceedings of the Systems Communications, 2005, 2005, pp. 428–433.
[10] Y.W. Law, J. Doumen, P. Hartel, Survey and benchmark of block ciphers for wireless sensor networks, ACM Trans. Sens. Networks (TOSN) 2 (1) (2006) 65–93.
[11] J. Lee, K. Kapitanova, S.H. Son, The price of security in wireless sensor networks, Comput. Networks 54 (17) (2010) 2967–2978.
[12] J. Zhu, G. Leina, Z. Xinfang, Implementation and time performance analysis of security suite in LR-WPAN 802.15.4, in: Proceedings of the Fourth International Conference on Wireless Communications, Networking and Mobile Computing, 2008, WiCOM '08, 2008, pp. 1–5.
[13] C. Alcaraz, J. Lopez, R. Roman, H.H. Chen, Selecting key management schemes for WSN applications, Comput. Secur. 31 (38) (2012) 956–966.
[14] G. Dini, I.M. Savino, S2RP: a secure and scalable rekeying protocol for wireless sensor networks, in: Proceedings of the Third IEEE International Conference on Mobile Adhoc and Sensor Systems MASS '06, 2006, pp. 457–466.
[15] A. Liu, P. Ning, TinyECC: a configurable library for elliptic curve cryptography in wireless sensor networks, in: Proceedings of the International Conference on Information Processing in Sensor Networks, 2008, IPSN '08, 2008, pp. 245–256.
[16] ZigBee Alliance, ZigBee Document 053474r17, ZigBee Specification, ZigBee Alliance, Jan. 2008.
[17] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, P. Levis, Collection tree protocol, in: Proceedings of the Seventh ACM Conference on Embedded Networked Sensor Systems (SenSys'09), 2009.
[18] R. Daidone, G. Dini, M. Tiloca, Open source toolset for IEEE 802.15.4 and ZigBee, 2010, <http://www.open-zb.net/>.
[19] J.H. Hauer, R. Daidone, R. Severino, J. Büsch, M. Tiloca, S. Tennina, Poster abstract: an open-source IEEE 802.15.4 MAC Implementation for TinyOS 2.1, in: Proceedings of the Eighth European Conference on Wireless Sensor Networks EWSN, 2011.
[20] J. Zheng, M.J. Lee, M. Anshel, Toward secure low rate wireless personal area networks, IEEE Trans. Mobile Comput. 5 (10) (2006) 1361–1373.
[21] P. Ganesan, R. Venugopalan, P. Peddabachagari, A. Dean, F. Mueller, M. Sichitiu, Analyzing and modeling encryption overhead for sensor network nodes, in: Proceedings of the Second ACM International Conference on Wireless Sensor Networks and Applications WSNA '03, ACM, New York, NY, USA, 2003, pp. 151–159.
[22] D. Jinwala, D. Patel, K. Dasgupta, Optimizing the block cipher and modes of operations overhead at the link layer security framework in the wireless sensor networks, in: Proceedings of the Fourth International Conference on Information Systems Security, ICISS '08, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 258–272.
[23] IEEE, IEEE Std. 802.15.4-2003, IEEE standard for information technology – telecommunications and information exchange between systems – local and metropolitan area networks – specific requirements. Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs), Institute of Electrical and Electronics Engineers Inc., New York, Oct. 2003.
[24] National Institute of Standards and Technology, Federal information processing standards publication 197, specification for the advanced encryption standard (AES), Nov, 2001.
[25] M. Passing, F. Dressler, Experimental performance evaluation of cryptographic algorithms on sensor nodes, in: Proceedings of the Third IEEE International Conference on Mobile Adhoc and Sensor Systems, MASS '06, 2006, pp. 882–887.
[26] Texas Instruments, Texas instruments cc2420 2.4 GHZ IEEE 802.15.4/ zigbee ready RF transceiver, 2012, <http://focus.ti.com/lit/ds/symlink/cc2420.pdf>.
[27] TinyOS security algorithms repository, <http://sourceforge.net/projects/tinyossecurity/files/>.
[28] R. Daidone, G. Dini, M. Tiloca, On experimentally evaluating the impact of security on IEEE 802.15.4 networks, in: Proceedings of the 2011 International Conference on Distributed Computing in Sensor Systems and Workshops, DCOSS '11, 2011, pp. 1–6.
[29] S.A. Camtepe, B. Yener, Key management in wireless sensor networks, in: J. Lopez, J. Zhou (Eds.), Cryptology and Information Security Series, Wireless Sensor Network Security, The Cryptology & Information Security Series (CISS), IOS Press, 2008, pp. 110–141.
[30] Y. Xiao, V.K. Rayi, B. Sun, X. Du, F. Hu, M. Galloway, A survey of key management schemes in wireless sensor networks, Comput. Commun. 30 (11–12) (2007) 2314–2341.
[31] F. Amini, M. Khan, J. Mišić, H. Pourreza, Performance of IEEE 802.15.4 clusters with power management and key exchange, J. Comput. Sci. Technol. 23 (2008) 377–388.
[32] J. Mišić, Cost of secure sensing in IEEE 802.15.4 networks, IEEE Trans. Wireless Commun. 8 (5) (2009) 2494–2504.
[33] G. Dini, I.M. Savino, LARK: a lightweight authenticated rekeying scheme for clustered wireless sensor networks, ACM Trans. Embedded Comput. Syst. 10 (4) (2011) 1–35.
[34] G. Dini, M. Tiloca, HISS: a highly scalable scheme for group rekeying, Comput. J. 56 (4) (2013) 508–525.
[35] S. Zhu, S. Setia, S. Jajodia, LEAP+: Efficient security mechanisms for large-scale distributed sensor networks, ACM Trans. Sens. Networks 2 (4) (2006) 500–528.
[36] C. Karlof, D. Sastry, D. Wagner, Tinysec: a link layer security architecture for wireless sensor networks, in: Proceedings of the Second International Conference on Embedded Networked Sensor Systems, SenSys '04, ACM, New York, NY, USA, 2004, pp. 162–175.
[37] A. Aziz, W. Diffie, Privacy and authentication for wireless local area networks, IEEE Pers. Commun. 1 (1) (1994) 25–31.
[38] A. Perrig, J. Stankovic, D. Wagner, Security in wireless sensor networks, Commun. ACM – Wireless Sens. Networks 47 (6) (2004) 53–57.
[39] TinyOS Working Group, Tinyos home page, 2012, <http://www.tinyos.net/>.
[40] Moteiv Corporation, Tmote iv low power wireless sensor module, Nov. 2006, <http://www.snm.ethz.ch/snmwiki/pub/uploads/Projects/tmote_sky_datasheet.pdf>.
[41] B. Bougard, F. Catthoor, D.C. Daly, A. Chandrakasan, W. Dehaene, Energy Efficiency of the IEEE 802.15.4 Standard in Dense Wireless Microsensor Networks: Modelling and Improvement Perspectives, in: Proceedings of the Conference on Design, Automation and Test in Europe DATE 2005, IEEE Computer Society, 2005, pp. 196–201.
[42] TinyOS security algorithms repository, <http://sourceforge.net/projects/tinyossecurity/files/>.
[43] Network Simulator Ns2, <http://www.isi.edu/nsnam/ns/>.
[44] G. Anastasi, M. Conti, M. Di Francesco, A Comprehensive analysis of the MAC unreliability problem in IEEE 802.15.4 wireless sensor networks, IEEE Trans. Ind. Inf. 7 (1) (2011) 52–65.
[45] IEEE, IEEE Standard for Local and metropolitan area networks. Part 15.4: Low-rate wireless personal area networks (LR-WPANs) amendment 1: MAC sublayer, Institute of Electrical and Electronics Engineers Inc., New York, Feb. 2012.