On Evaluating the Performance Impact of the IEEE 802.15.4 Security Sub-layer

ROBERTA DAIDONE[1], GIANLUCA DINI, GIUSEPPE ANASTASI
Department of Ingegneria dell'Informazione
University of Pisa
Largo Lazzarino 1, 56100 PISA, Italy
{r.daidone, g.dini, g.anastasi}@iet.unipi.it

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.

KEYWORDS: IEEE802.15.4; security; performance evaluation; wireless sensor network.

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 [20]. 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

[1] **Corresponding author**: Roberta Daidone; phone +390502217453: fax: + 390502217600; email: roberta.daidone@iet.unipi.it

provisions and options that constitute the *security sub-layer* [20]. The security sub-layer 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 a number of papers, including [26][29][34]. In addition, a security analysis of IEEE 802.15.4 security sub-layer — its services, vulnerabilities, and related countermeasures — has been presented in [14][35][40]. However, a thorough analysis of the impact that the security sub-layer 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, [7][14][18][25][27][40][43] 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 on the cost of key establishment, an important although collateral aspect [1][11][28][29]. However, what it is really missing is a thorough analysis providing quantitative indications regarding the impact that the security sub-layer 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 sub-layer. In particular, we evaluate the impact of security levels and key identification modes 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 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 sub-layer provides its services to above network and application layers. Although IEEE 802.15.4 security sub-layer is the natural choice for ZigBee [45], 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 sub-layer 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 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 carried out on an open-source

implementation of the IEEE 802.15.4 for TinyOS on Tmote Sky motes [8][22]. Finally, the availability of an open source implementation of the standard has allowed us to evaluate the memory overhead related to the security sub-layer. We show that while the code implementing the sub-layer 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 the performance of the IEEE 802.15.4 security sub-layer through analysis, simulation and experimental measurements, and achieves the aforementioned results. The closest work to ours is [14]. 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 sub-layer 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 sub-layer 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 sub-layer, its vulnerabilities, the possible attacks and related countermeasures. Among them, relevant

examples are [35][40][42]. 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 [7][16][18][23][25][27]. However, none of these works focuses on the performance implications of the standard security sub-layer.

Xiao *et al*. and Zhu *et al*. explored first the impact of security on IEEE 802.15.4 performance [40][43]. 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 the 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 [20]) whereas both [40] and [43] refer to the 2003 version [19]. The two versions greatly differ in the security sub-layer.

The closest work to ours is certainly [14]. Like us, Chen *et al*. refer to the 2006 version of the standard and evaluate the impact of the security sub-layer 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 [40][43], Chen *et al*. consider an incomplete implementation of the security sub-layer. Actually, their implementation is limited to the cryptographic transformations but completely neglects the data structures required by the security sub-layer and, consequently, their impact on memory consumption. Therefore, they fail to capture an important factor limiting to 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 [15], 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 [32]. 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 [36] and a software-based cryptography based on an implementation of AES-128 from the TinyOS security algorithms repository [39]. 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 [14] 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 Sections 4.1 and 4.2). 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 [32]. In contrast, we present an analytical model, an extended simulator, and a set of experiments on real sensor nodes validating both the analytical and simulation results.

A preliminary performance evaluation of the IEEE 802.15.4 security sub-layer was presented by the authors in [9]. The present work largely extends the previous workshop paper [9] from several standpoints. First, this paper completes and integrates the experimental evaluation in [9] by also considering latency and per-packet energy consumption. Furthermore, this paper considers both hardware-based and software-based encryption

whereas [9] only considers hardware-based encryption. Finally, this paper presents and validates both an analytical and a simulation cost model whereas [9] 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 sub-layer does not specify any key establishment scheme and, for this reason, we will not discuss this issue any further in the rest of the paper. Notwithstanding, it is important to notice here that, due to the limited resources and the large scale of a WSN, the key management schemes for desktop- and server-computing are generally not suitable. Therefore, key management and its performance in WSNs has become a very active research topic [6][41]. Many key management schemes have been proposed and evaluated, that are ready to use in IEEE 802.15.4 [1][2][6][30]. Relevant examples are [11][12][13][44].

Finally, we would like to spend a comment on [24]. 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 standard, with special focus on the CSMA/CA access protocol. The reader may refer to the standard [20] 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 fulfil 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: *beacon-enabled* and *nonbeacon-enabled*. In the beacon-enabled mode, the PAN coordinator periodically broadcasts *beacon* frames to synchronize channel access. In contrast, in the non-beacon 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.

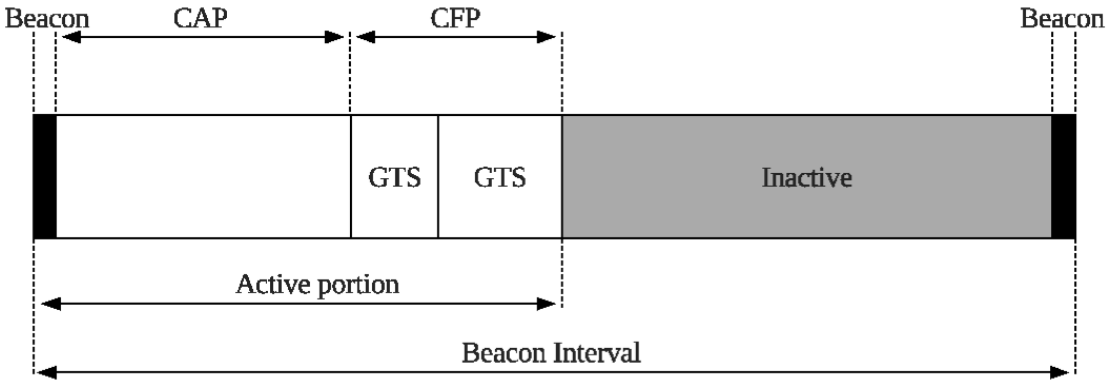


Figure 1. Structure of a superframe

With reference to Figure 1, in the beacon-enabled mode, two consecutive beacons bound a *superframe*. A superframe is divided into *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 the 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 back stage, unless the maximum allowed value has been reached. On the contrary, if the channel is found 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 SUB-LAYER

The IEEE 802.15.4 security sub-layer optionally provides link-layer security services to the higher layers. In general, link-layer 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-layer 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 [4]. Furthermore, specifically in a WSN, link-layer security supports in-network processing, passive participation and local broadcast to save traffic and reduce energy [10][24][33]. The two other alternatives, namely end-to-end security at the application layer and end-to-end security at the transport layer, provide a high layer 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-layer security and end-to-end security mechanisms can co-exist. Security at multiple places in the protocol stack is not necessarily considered harmful and constitutes a means to respond to demand for more security with yet more sophisticated use of cryptography [4][33].

The IEEE 802.15.4 sub-layer 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, finally, the *CCM* security level provides authentication and confidentiality. Furthermore, all security levels provide replay detection. In order to implement the cryptographic transformations required by the security levels, the standard uses the Advanced Encryption Standard (AES) block cipher [15]. 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.

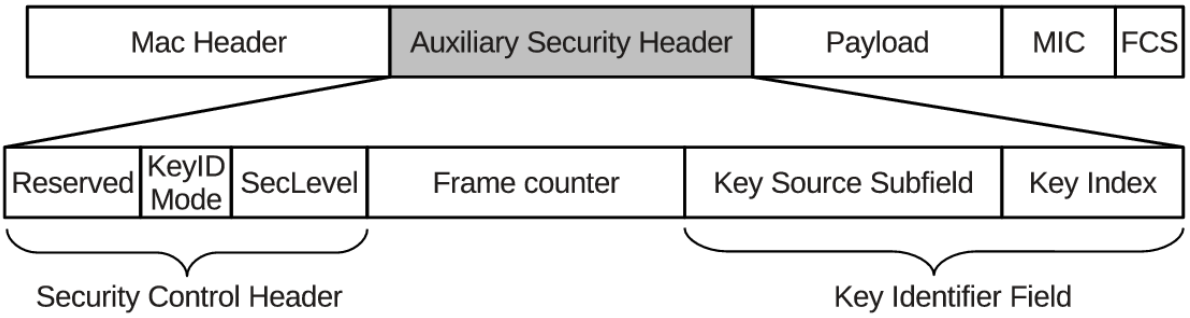
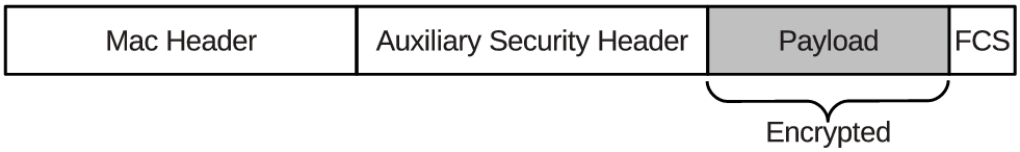
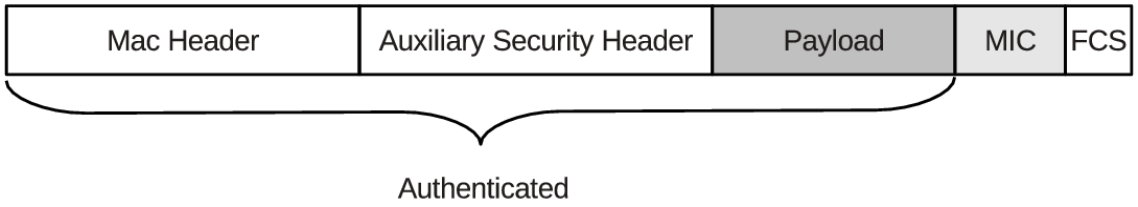


Figure 2. Auxiliary Security Header.

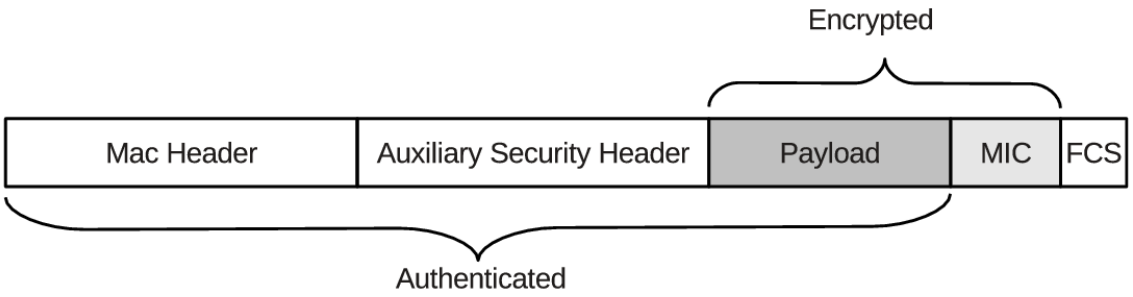
An unsecured frame is composed of four fields, namely a *Mac Header* (7–23 bytes), and a variable length *Payload* (0–118 bytes) and a *Frame Check Sequence* (*FCS*, 2 bytes). A secured frame contains an additional header called the *Auxiliary Security Header* (ASH), which carries the information required for security processing and frame securing/unsecuring and unsecuring. In a secured frame, the ASH is placed next to the standard MAC header (Figure 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.



A: CTR security level



B: CBC-MAC security level



C: CCM security level

Figure 3. Security levels

Security levels are depicted in Figure 3. The CTR security level secures a frame by encrypting its payload in the counter mode (Figure 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 auxiliary security header ASH, and the payload (Figure 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 (Figure 3.B). 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. As a rule of thumb, the CCM security level requires one block cipher encryption operation per 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 I. SECURITY LEVELS

Security level	Confidentiality	Authentication	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

Table I gives an overview of the 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.

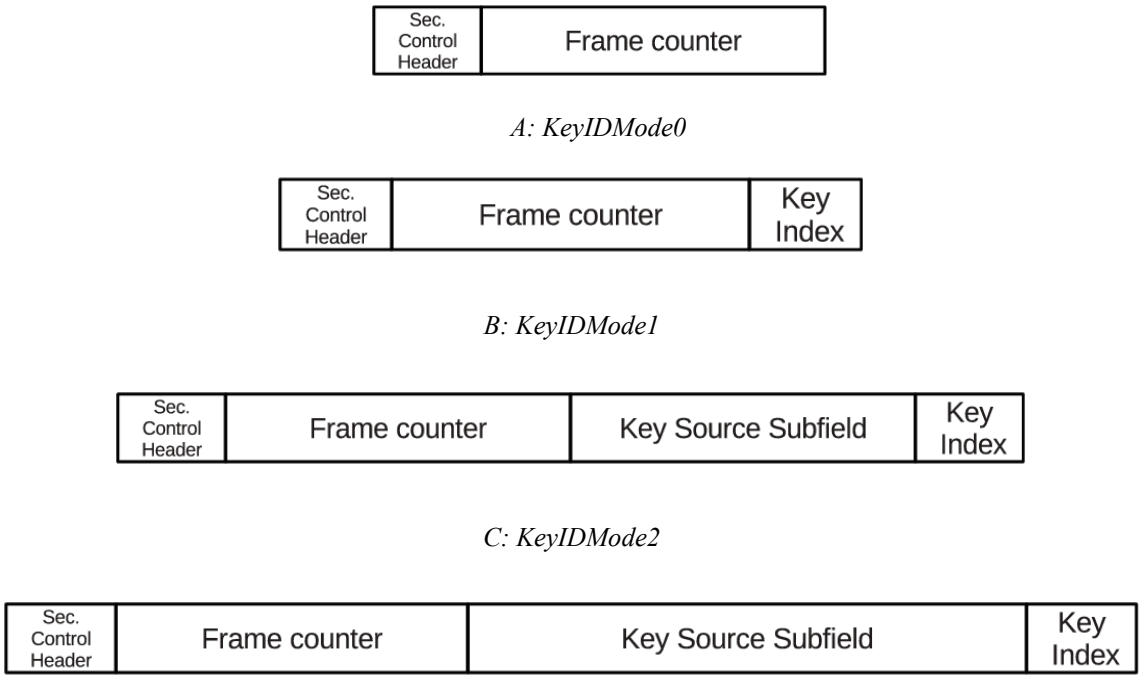


Figure 4. Format of ASH as a function of the key identifier mode.

Figure 4 shows the format of the Auxiliary Security Header (ASH), depending on the key identifier mode. In the case of Key Identifier Mode 0 (KeyIdMode0), ASH does not include any Key Identifier Field and security operations rely on a pre-shared static default key (Figure 2.A). In the case of Key Identifier Mode 1 (KeyIdMode1), 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 and eight bytes in the KeyIdMode3. Table II reports the size of the Auxiliary Security Header (ASH) as a function of the key identifier mode.

TABLE II. AUXILIARY SECURITY HEADER (ASH) SIZE VS. KEY IDENTIFIER MODE (KEYIDMODE).

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

3.1.1 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 [20]), however, in this section, we give a very concise description of the operations so as 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 structure 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 searching 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 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 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.

3.2 THE CONET OPEN IMPLEMENTATION OF IEEE 802.15.4

We have implemented a complete and fully operational version of the standard security sub-layer within an open-source implementation of IEEE 802.15.4 maintained by the TinyOS IEEE 802.15.4 Working Group [38]. The whole standard, including the security sub-layer, has been implemented [22] in the nesC language for the TinyOS operating system on the Tmote Sky platform equipped with CC2420 chipset[2]. The security sub-layer implementation can be downloaded from [8]. To the best of our knowledge, this is the first available free

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

implementation of IEEE 802.15.4 including security services. All the experimental evaluations reported in this paper have been carried out on this implementation.

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 introduced by the 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.

19.09 (± 0.04)antify the impact of communication and processing overhead, we consider the following performance indices:

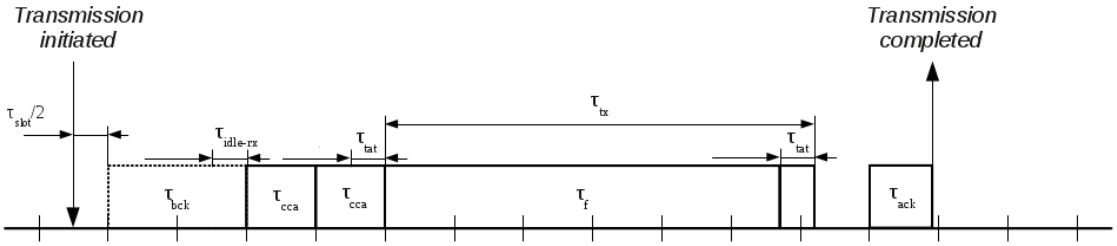
- *Latency* (τ), defined as the interval of time between the instant at which the source node starts the frame transmission and the instant at which the same node receives the corresponding ACK.
- *Goodput* (G), defined as the amount of useful information bits correctly received by the PAN coordinator 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 the transmission of the useful information carried by a MAC frame. The size of the payload field is always the same, irrespectively of

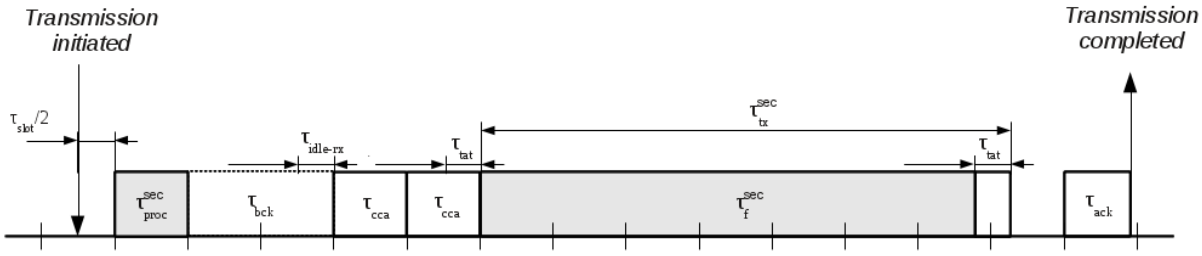
the security level used. As a consequence, goodput decreases when security increases. This effect will be quantified in the next sections.

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.



(a) Latency timeline without security



(b) Latency timeline with security

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

4.1.1 Latency and goodput

In order to model the impact of security on latency, we first derive latency in the absence of security and, then, we consider 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 Figure 5-a , assuming that the sensor node starts in the idle state, latency can be computed as:

$$\tau = \frac{\tau_{slot}}{2} + \tau_{bck} + 2\tau_{cca} + \tau_{tx} + \tau_{ack} \quad [1]$$

where

$$\tau_{tx} = \left\lceil \frac{\tau_t + \tau_{tat}}{\tau_{slot}} \right\rceil \times \tau_{slot} \quad [2]$$

In Equation [1], $\tau_{slot}/2$ 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 $\tau_{idle-rx}$, the time necessary to switch the radio from the idle state to the receiving state; $2\tau_{cca}$ accounts for the time necessary to perform two consecutive Clear Channel Assessment operations; 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 Equation [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* $\tau_{proc}^{\sec}$, which accounts for the security processing overhead, and the *security communication time* $\tau_{com}^{\sec}$, which accounts for the security communication overhead. The security processing time τ_{proc}^{sec} accounts for the time required by security operations. The security communication

time $\tau_{com}^{\sec}$ 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 $\tau_{com}^{\sec}$ has to be added to the frame transmission time τ_f. With reference to Figure 5-b, it follows that Equation [1] becomes:

$$\tau = \tau_{proc}^{\sec} + \frac{\tau_{slot}}{2} + \tau_{bck} + 2\tau_{cca} + \tau_{tx}^{\sec} + \tau_{ack} \quad [3]$$

where

$$\tau_{tx}^{\sec} = \left\lceil \frac{\tau_t + \tau_{tat} + \tau_{com}^{\sec}}{\tau_{slot}} \right\rceil \times \tau_{slot} \quad [4]$$

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 Figure 5-a and 5-b repeats for each following frame transmission. Hence:

$$G = \frac{P}{\tau} \quad [5]$$

and

$$G = \frac{P}{\tau^{\sec}} \quad [6]$$

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 Figure 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 Equation [3]). The energy ε consumed in an interval τ is the product of the power w consumed in τ and the time interval τ itself, i.e., $\varepsilon = W \times \tau$.Power consumption can be derived from the device datasheet.

In order to evaluate the per-packet energy consumption, we observe from Section 3.1.1 that the processing overhead τ_{proc}^{sec} can be split into two components, namely the *management overhead*, τ_{mgmt}^{sec}, that accounts for frame parsing and tables lookup, and the *encryption overhead*, τ_{crypto}^{sec}, 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 [36].

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 the 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_{crypto}^{sec}. 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 III. PARAMETERS.

device	Parameter	duration (μs)	current (mA)	power consumption (mW)	energy consumption (μJ)
MPS430 (v = 1.8V)	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 = 1.8V)	Total hw-based cryptography overhead ($\tau^{sec}_{crypto,hw}$)	1393	21.19 [27]	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

Table III shows the parameters values for calculating Equation [3], assuming that the communication chipset is CC2420 [36] and the microcontroller is MSP430 [31]. The values of absorbed current referring to MSP430 and CC2420 are taken from the respective datasheets. The only exception is the value of the absorbed current during $\tau^{sec}_{crypto,hw}$ that has been taken from [27]. 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 in a similar way (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 [1][5].

The duration of all delay components shown in Table III 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 3.2). For the sw-based cryptography case, we used the software implementation of AES-128 algorithm that is available in the TinyOS repository [39]. 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 then we took 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 tables lookup. 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 payload size nor the security level. In principle, $\tau^{sec}_{crypto,hw}$ should 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 overhead 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.

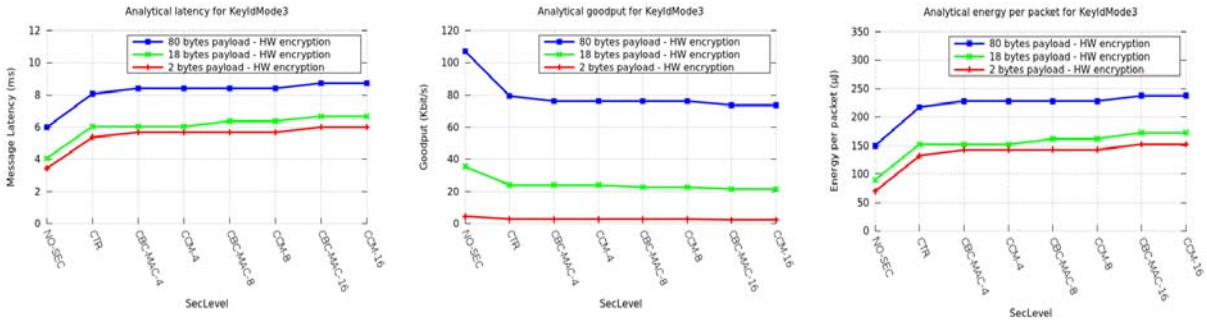
TABLE IV. FRAME EXPANSION DUE TO SECURITY (IN BYTES).

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

Table IV 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).

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.

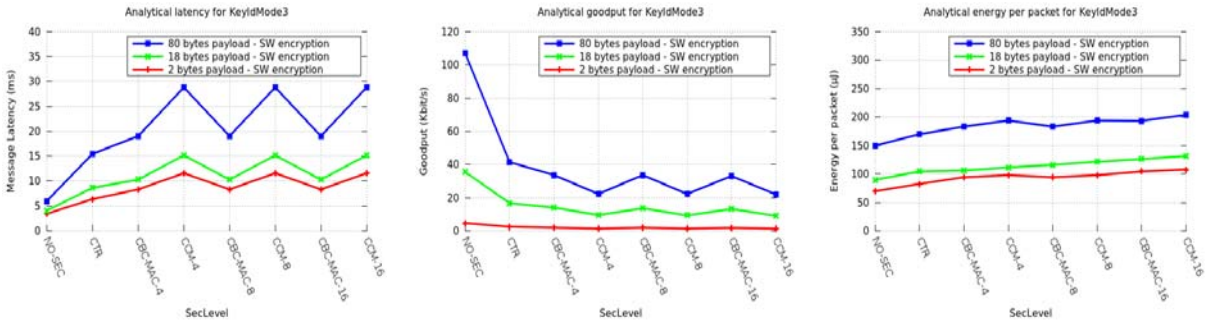


(a) Latency (b) Goodput (c) Energy

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

Figure 6 shows the trend of latency, goodput, and per-packet energy consumption with security levels, for different payload sizes, in KeyIdMode3, when using hardware-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-byte 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 the increased frame size is hidden by the backoff alignment, as expressed by Equation [4]. Similar considerations hold for goodput and energy consumption.



(a) Latency (b) Goodput (c) Energy

Figure 7: Latency, goodput and energy consumption (sw-based cryptography).

Figure 7 shows the trend of latency, goodput, and per-packet energy consumption with the security levels for different payload sizes, in KeyIdMode3, when using software-based cryptography. Similarly to the previous case (i.e. hardware-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. Finally, 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 behaviour. 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 III it turns out that both devices operates at 1.08 V but MSP430 absorbs 0.6 mA whereas CC2420 absorbs 21.19 mA, a current, and thus a power that is about 35 times larger than the former. As a consequence, even though sw-based encryption is slower than hw-based encryption, 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 [31], equipped with a MSP430 microcontroller, 10 Kbytes of RAM, 48 Kbytes 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 [36]. As to system software, sensor nodes run the TinyOS 2.x operating system (available from http://www.tinyos.net/) and the CONET open-source implementation of IEEE 802.15.4 (Section 3.2).

To validate the analytical results derived in previous section, we considered only two sensor nodes, KeyIdMode3 and a payload size equal to 18 bytes.

TABLE V. EXPERIMENTAL VS ANALYTICAL LATENCY. VALUES ARE IN MS.

SecLevel	Hw-based cryptography		Sw-based cryptography	
	Experimental	**Analytical**	**Experimental**	**Analytical**
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 VI. EXPERIMENTAL VS ANALYTICAL GOODPUT . VALUES ARE IN KBIT/S

SecLevel	Hw-based cryptography		Sw-based cryptography	
	Experimental	Analytical	Experimental	Analytical
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

Table V and Table VI show the analytical and experimental measurements for 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 trends 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 software-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, scenario 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 the considered scenario by means of the Ns2 simulation tool [37]. The basic IEEE 802.15.4 model has been extended to take into account delays due to security, i.e., τ_{proc}^{sec} and τ_{comm}^{sec}. The former was modelled as a pure delay. The latter has been implemented by fictitiously enlarging the payload by a quantity specified in Table IV for each relevant pair (security level, KeyIdMode). In simulations, we only considered KeyIdMode3. Furthermore, we set the transmission range to 15 m and the carrier sensing range to 30 m as in [3]. In addition, we 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. The presented results are averaged over the ten replications. For each repetition, we discarded the initial transient period during which nodes associate to the PAN coordinator before starting generating data packets. We also derived confidence intervals considering a 95% confidence level.

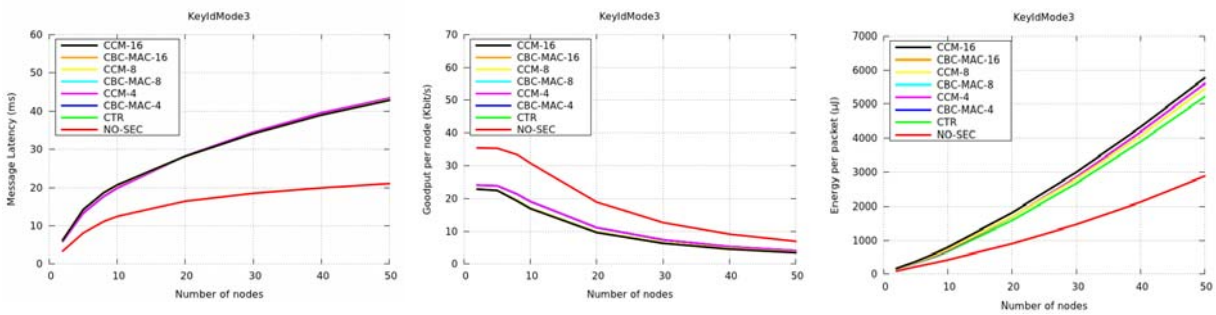


Figure 8. Simulation results for latency, goodput, and energy consumption (hw-based cryptography).

As to hw-based cryptography, Figure 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 VII 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.

TABLE VII. LATENCY AND GOODPUT: EXPERIMENTAL VS SIMULATION RESULTS (HW-BASED CRYPTOGRAPHY)

	SecLevel	**Latency (ms)**		**Goodput (kbit/s)**	
		Experimental	**Simulation**	**Experimental**	**Simulation**
2 nodes	NO-SEC	4.28 (± 0.14)	3.42 (± 0.01)	33.62 (± 1.1)	35.44 (± 0.0)
	CTR	6.35 (± 0.13)	5.98 (± 0.01)	22.69 (± 0.47)	24.07 (± 0.0)
	CBC-MAC-4	6.57 (± 0.18)	5.98 (± 0.01)	21.90 (± 0.59)	24.07 (± 0.0)
	CCM-4	6.51 (± 0.17)	5.98 (± 0.01)	22.12 (± 0.58)	24.07 (± 0.0)
	CBC-MAC-8	6.62 (± 0.13)	6.30 (± 0.01)	21.77 (± 0.43)	22.84 (± 0.0)
	CCM-8	6.79 (± 0.22)	6.30 (± 0.01)	21.20 (± 0.69)	22.84 (± 0.0)
	CBC-MAC-16	6.95 (± 0.22)	6.30 (± 0.01)	20.71 (± 0.63)	22.84 (± 0.0)
	CCM-16	6.99 (± 0.20)	6.30 (± 0.01)	20.60 (± 0.58)	22.84 (± 0.0)
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)

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 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 (Figure 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 that 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, on average, a node experiences a latency longer than that in the case of unsecured traffic. Goodput has a dual trend (Figure 8-b), with respect to latency.

Similar considerations also apply to the energy consumption per delivered packet (Figure 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 Figure 8-b.

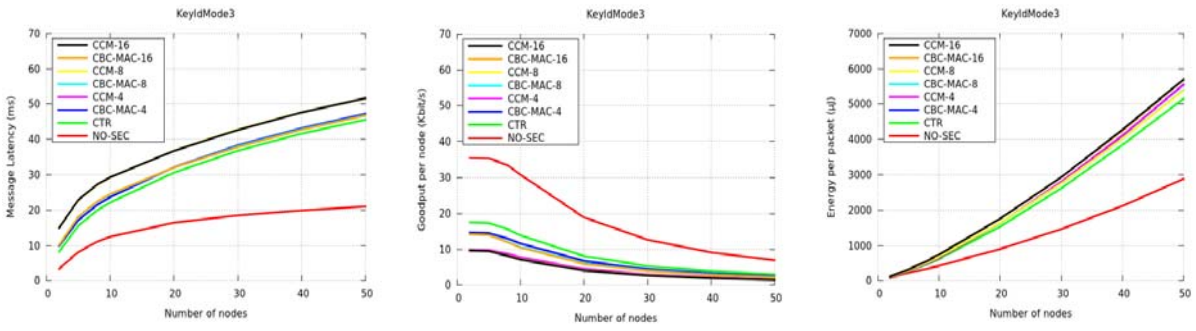


Figure 9: Simulation results for latency, goodput, and per-packet energy consumption (sw-based cryptography)

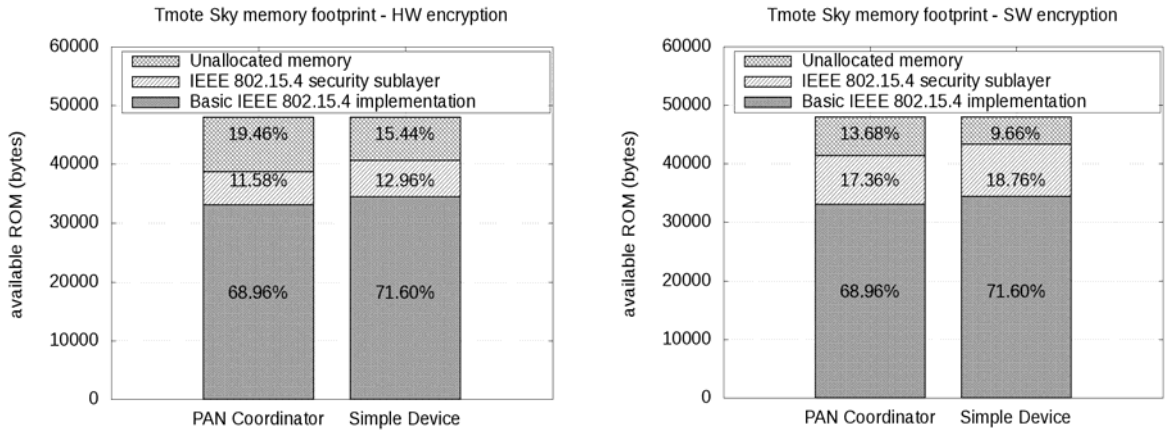
As to sw-based cryptography, Figure 9 shows the trend of latency, goodput, and per-packet energy consumption with the number of nodes for each security level. As above, confidence intervals are so small that they cannot be graphically appreciated.

As expected, Figure 9 shows that switching from unsecured to secured traffic causes a performance loss. Furthermore, the figure also shows that payload size and security level have influence on such a loss, due to the number of block encryption operations that are required. However, Figure 9-c shows that per-packet energy consumption constitutes an exception and as its trend is very similar to that in hw-based cryptography (Figure 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 sub-layer.



(a) (b)

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

Figure 10 shows the ROM footprint breakdown on both the PAN coordinator and a regular sensor node. With hw-based cryptography (Figure 10-a), the amount of memory required by the security sub-layer 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 used is taken by the IEEE 802.15.4 implementation (i.e. the original communication stack). Note also that 19.46% (15.44%) of memory on the PAN coordinator (regular node) remains available for other uses (e.g., applications). With sw-based cryptography (Figure 10-b), the amount of memory required by the security sub-layer 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 the memory occupancy is due to the IEEE 802.15.4 implementation and software implementation of the encryption algorithm. Also note that 13.68% (9.66%) of memory on the PAN coordinator (regular node) remains available for other uses (e.g., applications).

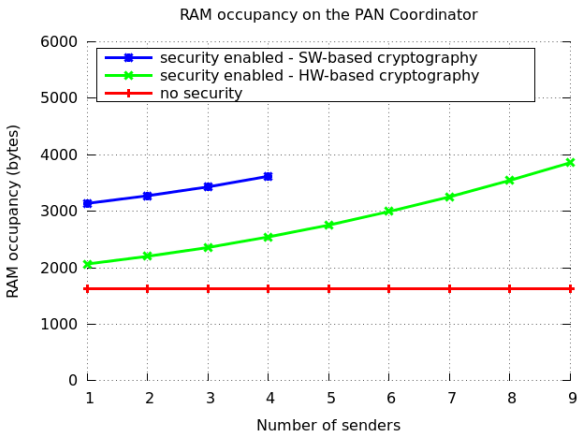


Figure 11. RAM memory overhead

However, the space necessary to allocate the TinyOS image is not the only storage cost that we have to pay in order to use the security sub-layer. As discussed in Section 3.1.1, the security sub-layer 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. Figure 11 shows the trend of RAM occupancy on the PAN coordinator when the number of nodes grows. In the case of hw-based cryptography, nine sender nodes require about 3858 bytes of RAM. 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, you have better leave at least 500 byte, or maybe 1 Kbytes, 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 higher 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 overall system scalability.

5 CONCLUSIONS

We have presented a performance evaluation of the IEEE 802.15.4 security sub-layer. 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 [14], 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 sub-layer 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 [21]. 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 sub-layer, but the possible different size of the Frame Counter field. Hence, the security cost model can be extended to encompass these changes as well.

ACKNOWLEDGEMENTS

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] C. Alcaraz, J. Lopez, R. Roman, H-H Chen, "Selecting key management schemes for WSN applications," Computers & Security, Vol. 31, No. 8, pp. 956–966, November 2012.

[2] F. Amini, M. Khan, J. Mišić and H. Pourreza, "Performance of IEEE 802.15.4 Clusters with Power Management and Key Exchange", *Journal of Computer Science and Technology*, 23 (3), pp. 377–388, May 2008.

[3] G. Anastasi, M. Conti, M. Di Francesco, "A Comprehensive Analysis of the MAC Unreliability Problem in 802.15.4 Wireless Sensor Networks", *IEEE Transactions on Industrial Informatics*, Vol.7, N.1, pp.52-65, February 2011.

[4] A. Aziz and W. Diffie, "Privacy and Authentication for Wireless Local Area Networks," *IEEE Personal Communications*, Vol.1, N.1, pp. 25–31, 1994

[5] B. Bougard, F. Catthoor, D. C. Daly, A. Chandrakasan, and W. Dehaene, "Energy Efficiency of the IEEE 802.15.4 Standard in Dense Wireless Microsensor Networks: Modelling and Improvement Perspectives", in *Proceedings of Conference on Design, Automation and Test in Europe* (DATE '05), Vol.1, March 7-11, 2005.

[6] S. Camtepe, B. Yener, "Key management in wireless sensor networks," in *Wireless sensor network security*, J. Lopez, J. Zhou (Eds.), IOS Press (2008)

[7] C.-C. Chang, D.J. Nagel, S. Muftic, "Balancing security and energy consumption in wireless sensor networks", *Mobile Ad-Hoc and Sensor Networks*, Lecture Notes in Computer Science Volume 4864, pp. 469–480, 2007.

[8] R. Daidone, G. Dini, and M. Tiloca, "IEEE 802.15.4 security sublayer implementation for TinyOS nesC", 2011. URL http://www.open-zb.net/downloads.php.

[9] R. Daidone, G. Dini, and M. Tiloca, "On experimentally evaluating the impact of security on IEEE 802.15.4 networks", in *Proceedings of the Third International Workshop on Performance Control in Wireless Sensor Networks* (PWSN 2011), pp 20-25, Barcelona, Spain, 27-29 June 2011.

[10] R. Daidone, G. Dini, M. Tiloca. "On preventing GTS-based Denial of Service in IEEE 802.15.4", in *Proceedings of the European Conference on Wireless Sensor Networks* (EWSN 2012), pp. 1–2, Trento, Italy, 15-17 February 2012.

[11] G. Dini and I.M. Savino, "S2RP: a Secure and Scalable Rekeying Protocol for Wireless Sensor Networks," *In Proceedings of the 2006 IEEE International Conference on Mobile Adhoc and Sensor Systems* (MASS'06), pp. 457–466, Vancouver, BC, Canada, October 09–12, 2006.

[12] G. Dini and I.M. Savino, "LARK: A Lightweight Authenticated ReKeying Scheme for Clustered Wireless Sensor Networks", *ACM Transactions on Embedded Computing Systems*, 10(4), November 2011.

[13] G. Dini and M. Tiloca, "HISS: A HIghly Scalable Scheme for Group Rekeying," *The Computer Journal*, Vol. 56, No. 4, pp. 508-525, 2013.

[14] Feng Chen, Xiaolong Yin, R. German, and F. Dressler, "Performance impact of and protocol interdependencies of IEEE 802.15.4 security mechanisms", in *Proceedings of the Fifth IEEE International Workshop on Wireless and Sensor Networks Security* (WSNS 2009), pp.1036-1041, Macau, China, 12-15 October 2009

[15] Federal Information Processing Standards Publication 197, *Specification for the Advanced Encryption Standard* (AES), National Institute of Standards and Technology (NIST), November 2001.

[16] 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 2nd ACM International Conference on Wireless Sensor Networks and Applications* (WSNA'03), pp. 151–159, San Diego, CA, USA, September 19, 2003.

[17] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," *In Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems* (SenSys '09). Berkeley, California - November 4–6, 2009.

[18] G. Guimaraes, E. Souto, D. Sadok, J. Kelner, "Evaluation of security mechanisms in wireless sensor networks," in *Proceedings of Systems Communications,* pp. 428–433, Montreal, Canada, 14-17 August, 2005.

[19] 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, October 2003.

[20] 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, September 2006.

[21] IEEE Std 802.15.4e-2012 , IEEE Standard for Local and metropolitan area networks--Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC sublayer," (Amendment to IEEE Std 802.15.4-2011) , April 2012.

[22] J.-H. Hauer, R. Daidone, R. Severino, J. Büsch, M. Tiloca and S. Tennina, "An Open-Source IEEE 802.15.4 MAC Implementation for TinyOS 2.1," *in Poster at the 8th European Conference on Wireless Sensor Networks* (EWSN 2011), Bonn, Germany, 23-25 February 2011.

[23] 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 4th International Conference on Information Systems Security* (ICISS 2008), pp. 258–272, Hyderabad, India, 16-20 December 2008.

[24] N. Karlof, C. Sastry and D. Wagner, "TinySec: A Link Layer Security Architecture for Wireless Sensor Networks", in *Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems* (SenSys'04), pp. 162–175, Baltimore, 3-5 November 2004.

[25] Y.W. Law, J. Doumen, P. Hartel, "Survey and benchmark of block ciphers for wireless sensor networks", *ACM Transactions on Sensor Networks*, 2(1), pp. 65–93, 2006.

[26] J.-S. Lee, "Performance evaluation of IEEE 802.15.4 for low-rate wireless personal area networks", *IEEE Transactions on Consumer Electronics*, vol. 52, no. 3, pp. 742–749, August 2006.

[27] J. Lee, K. Kapitanova, and S.H. Son, "The price of security in wireless sensor networks," *Computer Networks*, vol. 54, no. 17, pp. 2967–2978, 2010.

[28] A. Liu and P. Ning, "TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks", in *Proceedings of the Seventh International Conference on Information Processing in Sensor Networks* (IPSN 2008), pp. 245–256, St. Louis, Missouri, USA, April 2008.

[29] J. Mišić, S. Shafi and V.B. Mišić, "The impact of MAC parameters on the performance of 802.15.4 PAN", *Ad-Hoc Networks*, vol. 3, no. 5, pp. 509–528, 2005.

[30] J. Mišić, "Cost of secure sensing in IEEE 802.15.4 networks", *IEEE Transactions on Wireless Communications*, 8(5), pp. 2494–2504, 2009.

[31] Moteiv Corporation, Tmote Sky Datasheet, 2006 http://www.eecs.harvard.edu/~konrad/projects/shimmer/references/tmote-sky-datasheet.pdf.

[32] M. Passing and F. Dressler, "Experimental Performance Evaluation of Cryptographic Algorithms on Sensor Nodes", in *Proceedings of the Third IEEE International Conference on Mobile Ad-hoc and Sensor Systems* (MASS'06), pp.882-887, Vancouver, BC, Canada, 9-12 October, 2006.

[33] A. Perrig, J. Stankovic, and D. Wagner, "Security in Wireless Sensor Networks," Communications of the ACM, Vol.47, No.6, pp.53–57, June 2004.

[34] I. Ramachandran, A.K. Das and S. Roy, "Analysis of the contention access period of IEEE 802.15.4 MAC", *ACM Transactions on Sensor Networks*, vol. 3, no. 1, March 2007.

[35] N. Sastry and D. Wagner, "Security considerations for IEEE 802.15.4 networks", in *Proceedings of the 3rd ACM workshop on Wireless security* (WiSe '04), pp. 32–42, 1 October 2004, New York, NY, USA.

[36] Texas Instruments, CC2420 2.4 GHz IEEE 802.15.4/ZigBee ready RF Transceiver, 2012 http://www.ti.com/lit/ds/symlink/cc2420.pdf.

[37] The Network Simulator Ns2, http://www.isi.edu/nsnam/ns/.

[38] TinyOS IEEE 802.15.4 Working Group http://tinyos.stanford.edu:8000/15.4 WG

[39] TinyOS security algorithms repository http://sourceforge.net/projects/tinyossecurity/files/

[40] Y. Xiao, H.-H. Chen, B. Sun, R. Wang, and S. Sethi, "MAC security and security overhead analysis in the IEEE 802.15.4 wireless sensor networks", *EURASIP Journal on Wireless Communication and Networking*, pp. 81–81, April 2006.

[41] Y Xiao, V.K. Rayi, B. Sun, X. Du, F. Hu, and M. Galloway, "A survey of key management schemes in wireless sensor networks," *Computer Communications*, Vol. 30, No. 11-12, pp. 2314–2341, 2007.

[42] M. Zheng, J. Lee and M. Anshel, "Toward secure low rate wireless personal area networks", *IEEE Transactions on Mobile Computing*, vol. 5, no. 10, pp. 1361 –1373, October 2006.

[43] L. Zhu, J. Gao and X. Zhang, "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* (WiCOM '08), Dalian, China, October 12-17, 2008.

[44] S. Zhu, S. Setia and S. Jajodia, "LEAP+: Efficient security mechanisms for large-scale distributed sensor networks", *ACM Transactions on Sensor Networks*, Vol.2, No.4, 500–528, November 2006.

[45] ZigBee Alliance. ZigBee Document 053474r17, ZigBee Specification, January 2008 http://www.zigbee.org/.