Hindawi
Security and Communication Networks
Volume 2018, Article ID 5087065, 14 pages
https://doi.org/10.1155/2018/5087065

Research Article

Energy and Area Costs of Lightweight Cryptographic Algorithms for Authenticated Encryption in WSN

Carlos Andres Lara-Nino, Arturo Diaz-Perez, and Miguel Morales-Sandoval

CINVESTAV Tamaulipas, "Parque Científico y Tecnológico TECNOTAM", 87130 Victoria, TAMPS, Mexico

Correspondence should be addressed to Carlos Andres Lara-Nino; clara@tamps.cinvestav.mx

Received 27 April 2018; Revised 17 July 2018; Accepted 13 August 2018; Published 4 September 2018

Academic Editor: Wei Wang

Wireless Sensor Networks (WSN) aim at linking the cyber and physical worlds. Their security has taken relevance due to the sensitive data these networks might process under unprotected physical and cybernetic environments. The operational constraints in the sensor nodes demand security primitives with small implementation size and low power consumption. Authenticated encryption is a mechanism to provide these systems with confidentiality, integrity, and authentication of sensitive data. In this paper we explore hardware implementation alternatives of authenticated encryption through generic compositions, to assess the costs of this security approach in WSN. Two symmetric ciphers, AES and PRESENT, and two hash functions, SHA and SPONGENT, are used as the underlying primitives for the generic compositions. All the architectures studied in this work are implemented and evaluated in an FPGA-based WSN mote. The life time of the sensor node is used as the main evaluation metric but FPGA resources are also reported. From the experimental results obtained, it is shown how lightweight ciphers significantly contribute to reduce implementation area and energy consumption overheads, extending the lifetime of the sensor node.

1. Introduction

Wireless Sensor Network (WSN) is a relatively novel technology that has attracted attention by being one of the cornerstones for the Internet of Things (IoT) [1]. These networks are found in applications with deep social implications such as healthcare, military, industrial, and domotics. The primary elements of a WSN, denominated sensor nodes or *motes*, have special properties that impose special requirements of WSN applications. Most notably is that they communicate wirelessly, have small physical size, possess low computing capabilities, and operate using batteries.

The batteries of a sensor node cannot be replaced, and so their operational life is limited [2]. Significant effort has been invested to study the longevity of a WSN as a function of its energy profile [3, 4]. The study of the energy costs of different security mechanisms for WSN has also been of notable interest in the literature [5–7]. To reduce motes fabrication costs, hardware implementations must be small enough [8], and the energy consumption must be low to extend the operational life of the WSN [2]. Therefore, the implementation of cryptographic algorithms for WSN must be resource and energy aware.

In this work we study the energy and area costs associated with providing security services to WSN motes. The main goals of our research are protecting the sensitive data in a WSN, guaranteeing the user privacy, and ensuring the trustworthiness of a WSN.

In some WSN applications where sensitive data is collected and transmitted (i.e., wireless body area networks), it is compulsory to guarantee information security through protected communications. Particularly, the IEEE Standard for local and metropolitan area networks, IEEE 802.15.4 [9], recommends to guarantee the confidentiality and integrity/authentication services of transmitted data by means of the Advanced Encryption Standard (AES) [10] in Counter (CTR) with Code-Block Chain (CBC) Message Authentication Code (MAC) operation mode (CCM, CCM∗) [11].

AES CCM∗ is made up of two underlying primitives, AES in CTR mode for the confidentiality service and AES in CBCMAC mode for the integrity and authentication

services. AES is a strong symmetric key cryptosystem with well understood cryptographic properties. AES is widely used for general purpose applications with good performance, in both software and hardware. However, is AES the optimal primitive to be used in constrained environments such as sensor nodes?

In order to study alternatives to the use of AES we shall require alternative schemes which provide the same protections as CCM∗. The security services defined in IEEE 802.15.4 may also be provided by authenticated encryption (AE) compositions. These constructions allow for greater flexibility on the selection of the underlying cryptographic primitives.

An authenticated encryption scheme designed by "generic composition" is an operation mode which makes black-box use of a given symmetric encryption scheme and a given MAC algorithm. Three of such constructions have been studied in the literature: *Encrypt-and-MAC*, *MAC-then-encrypt*, and *Encrypt-then-MAC* [12]. The security of these schemes has been evaluated assuming that the given symmetric encryption transformation is secure against chosen-plaintext attacks (CPA) and the given MAC is unforgeable under chosen-message attacks (CMA) [12].

Given that it is possible to select different cryptographic primitives to realize authenticated encryption systems, what would be the benefit for sensor nodes if the implemented AE solutions are based on lightweight algorithms over generic alternatives?

The privacy-preserving CTR mode is CPA-secure. Under this operation mode, only the encryption procedure of the cipher is required to encrypt or decrypt the data. This is advantageous for constrained devices. When a MAC function is added to the scheme then it is possible to provide integrity and authentication. These MACs can be generated using block ciphers (CBCMAC) or hash functions (HMAC) as long as the operation mode is CMA secure. While hash-based tags provide desirable security features, their efficiency has been questioned [13], although not empirical evidence has been provided.

In the AE constructions evaluated we use generic and lightweight ciphers and hash functions. We study the use of block ciphers for symmetric encryption and MAC generation, as well as the use of hash functions in MAC algorithms. In this sense our goals are (a) to quantify the cost of lightweight algorithms compared against generic algorithms, both in the case of block ciphers and hash functions, and (b) to quantify the cost of hash functions compared against block ciphers. Authenticated encryption through generic composition provides us with the means to reach these goals.

Recently, modern operation modes for authenticated encryption have been proposed in the literature [14–18]. These proposals allow for single-pass encryption and integrity without the need of a MAC. Although the added efficiency is interesting for WSN, we have selected to use generic compositions as case study based on the following premises:

(i) The use of a generic composition method is advantageous in that it facilitates the design of an authenticated encryption with associated data (AEAD) construct by making it possible to choose both an appropriate symmetric encryption scheme and a message authentication scheme independently [19].

(ii) The flexibility on the selection of the underlying primitives is the enabler for the study of a broader range of alternatives in the use of encryption and MAC schemes.

(iii) The chosen composite schemes are usually already supported by existing security analyses; consequently tailored security analysis of the composed scheme is unnecessary [19].

(iv) The selected AE paradigm is well suited for WSN communications as no decryption is needed to verify the integrity of the message [19].

(v) Using generic compositions as case study allows flexibility in the selection of the underlying security primitives. In this sense, AE through generic compositions acts as an enabler for additional purposes such as: the cost analysis of (a) lightweight algorithms and (b) hash-based MAC functions. This assessment can be useful for different AE schemes which can adopt lightweight cryptographic algorithms or hash-based MACs.

We evaluate different configurations of AE over a battery powered sensor node prototyped in FPGA. This mote is enabled to perform basic tasks of sensing, processing, and transmission of messages. Additionally, the prototype incorporates a security core that implements the AE configurations evaluated in this work. The underlying symmetric encryption function can be implemented as a lightweight block cipher in a convenient confidentiality mode, and the underlying MAC function can also make use of lightweight algorithms, with cipher-based or hash-based schemes. In this work we aim at determining experimentally which of these options provide greater advantages for sensor nodes, regarding energy consumption and usage of hardware resources.

Our contributions can be summarized as follows:

(1) We quantify the overhead in the lifetime of WSN motes when implementing authenticated encryption through generic compositions.

(2) We prove empirically the advantage of using lightweight algorithms over generic alternatives in reducing the impact in the lifetime of WSN motes.

(3) We find the costs of using hash functions over block ciphers in hardware and the impact of those constructions in the lifetime of a WSN mote.

2. Materials and Methods

In this section we review the relevant works from the literature, provide some preliminary notions, describe our design methodology for the proposed solution, and present our experimental design.

2.1. Related Work. Some classical proposals for AE besides the generic compositions include the operation modes

CCM [11], GCM [20], and OCB [21]. The CCM mode of operation requires the use of a block cipher in CTR mode in order to provide confidentiality and CBCMAC mode in order to provide authentication and integrity. The counters used by the cipher are generated from an initialization value called nonce, which must be different for every secret key used. The Galois/Counter Mode (GCM) operates in the same way as the CCM for encryption. However, a system based on Galois field multiplications is used in order to provide integrity and authentication. The advantage of GCM over CCM is that the authentication tag can be computed in parallel. The Offset Codebook Mode (OCB) is a scheme for including a MAC into the operation of a block cipher and in this single pass provides confidentiality and authentication. Different versions of OCB have been proposed in order to include support for associated data and to improve performance. While CCM and GCM require $2n$ calls to the underlying block cipher, OCB requires only $n+2$ calls in order to provide the same security services, where n is the number of blocks in the plaintext. The drawback is that OCB is ruled by patents and licenses that limit its practical use; this has been one of the motivations for the wider adoption of CCM.

Other works have proposed AE schemes from the original generic compositions. The authors in [22] review different constructions for authenticated encryption using stream ciphers as underlying primitives. A qualitative analysis of the different alternatives is performed in order to identify the most suitable for energy, processing power, and memory constrained devices. The research focuses on the stream ciphers from the eSTREAM project. Three Dragon-MAC based constructions are proposed and analyzed, these are *Encrypt-then-MAC* compositions and thus considered secure. Nonetheless, no experimental data is provided to assess the suitability of these solutions. The work in [19] proposes an authenticated encryption operation mode called Joint Cipher Mode (JCM) based on the *Encrypt-then-MAC* approach. JCM provides an authenticated encryption with associated data (AEAD) cryptographic service in packet-based communication protocols. An additional AEAD construction named TinyAEAD is derived from JCM, which utilizes a block cipher and a Matyas-Meyer-Oseas (MMO) hash as underlying algorithms. Simulation results for TinyAEAD, CCM and EAX' in a PIC18F2320 clocked at 40 MHz are provided. Results of the software simulation benchmark indicate that TinyAEAD exhibits advantageous performance regarding processing latency, processing throughput, and processing efficiency for practical WSN communicated data frame lengths.

The need for real-time cryptographic solutions has also been expressed. In [16] the authors propose a real-time based AE scheme where the "real-time key stream" is generated from any secure block cipher like AES. The authors introduce two modes of operation: the integrity aware real-time based counter (IAR-CTR) and the cipher feedback (IAR-CFB) mode. Both the proposed modes of operation aim to offer both confidentiality and message integrity in a single pass without a tag. According to the authors, their real-time solutions are adapted to systems where parameters like integrity awareness, latency, jitter, and parallelism are critical. The work provides results simulated using the CryptoPP cryptographic library on an Intel Core 2 Duo 1.83-GHz machine with 512MB of RAM. Another related proposal for real-time encryption is found in [17]. In that work the authors present a real-time alternative for the CFB and OCFB modes called real-time based optimized cipher feedback mode (RT-OCFB). According to the authors, this proposal remedies the "stalling problem" and the inability to arrange the key stream in advance. Similarly as CFB and OCFB, this proposal does not provide integrity nor authentication. No implementation results are presented in that work.

Some additional proposals have focused on single-pass AE. Two stream-cipher modes of authenticated encryption, namely, PFC-CTR (counter-based authenticated encryption environment) and PFC-OCB (OCB-based authenticated encryption environment), are proposed in [14, 15]. The designs rely on a key stream generated from a block cipher. The schemes provide data confidentiality, authentication, and protect against replay attacks with reduced cipher calls. No experimental assessment of these modes of operation is presented. A new mode of operation for block encryption called plaintext feedback XORing (PFX) is presented in [18]. PFX provides strong integrity and used with other encryption modes which achieves confidentiality. Two of these schemes are presented in the work: PFX-CTR and PFX-INC. Implementation results are not provided.

In this work, we have selected generic compositions over schemes such as CCM, GCM, and OCB since we are interested in testing different MAC algorithms, such the ones based on hash functions. The modern operation modes reviewed, although efficient, were not considered on the same basis. Moreover generic compositions can allow us to include associated data which is an interesting feature for WSN. Nonetheless, our findings can be used in implementations of the aforementioned schemes since we are not only interested in evaluated the AE construction, but also its underlying primitives.

We analyze the suitability of authenticated encryption through generic compositions for WSN. Our goal is to construct such a solution which provides confidentiality, integrity, and authentication with reduced energy consumption and implementation area. Unlike previous proposals, our study provides sufficient experimental data to support our findings.

2.2. Preliminaries. Authenticated encryption refers to a shared-key based transformation with aims to provide both confidentiality and authentication of the underlying data [12]. These schemes use a key to transform a plaintext into a ciphertext; the inverse transformation requires the same key to convert the ciphertext again into a plaintext or into a symbol that warns if the ciphertext has been tampered with.

2.2.1. Security Notions. Indistinguishability has been proposed as the most important security goal for symmetric encryption [12]. This property refers to the inability of an attacker to distinguish between a ciphertext and a random string. It can be studied under either the chosen-plaintext (CPA) or the chosen-ciphertext (CCA) attacks to determine

Table 1: Security results for different authenticated encryption schemes under the assumption that $\mathcal{SE}$ is IND-CPA secure and that $\mathcal{MA}$ is SUF-CMA secure, retrieved from [12]. The notion of IND-CPA security for $\mathcal{SE}$ is used since it is the weakest security notion (compared with IND-CCA), yet it is sufficient to achieve IND-CCA security under $\mathcal{AE}$.

Composition Method	Confidentiality		Authentication	
	IND-CPA	IND-CCA	INT-PTXT	INT-CTXT
Encrypt-and-MAC	Insecure	Insecure	Secure	Insecure
MAC-then-encrypt	Secure	Insecure	Secure	Insecure
Encrypt-then-MAC	Secure	Secure	Secure	Secure

different security properties. The security notion for symmetric encryption based on indistinguishability is strong. If the attacker cannot find any distinguishers in the transformation, the probability of other attack models succeeding is negligible. The notions of security for indistinguishability under the chosen-plaintext and the chosen-ciphertext attacks are abbreviated as IND-CPA and IND-CCA, respectively [12].

The notions for authentication of symmetric encryption, as presented in [12] are (i) the integrity of plaintexts (INT-PTXT) and (ii) the integrity of ciphertexts (INT-CTXT). In the first one, the objective is to ensure that the decryption will not produce a message never encrypted by the sender. In the second one, the creation of ciphertexts that were not previously produced by the sender is prevented.

These security notions for confidentiality and authentication share well-known relations which are used to demonstrate the security of complex cryptographic schemes; for a detailed description of these relations the reader should refer to [12].

For MAC algorithms, the security notions involve the concept of *unforgeability*. This property evaluated under the chosen-message attack (CMA) can be weak (WUF-CMA) or strong (SUF-CMA). The first case implies that it should be infeasible for the adversary to find a message-tag pair for an unknown message, even after a chosen-message attack. In the second case it is required that it be computationally infeasible for the adversary to find a new message-tag pair even after a chosen-message attack. In [12] it is noted that any pseudorandom function (PRF) is a strongly unforgeable MAC, and most practical MACs seem to be strongly unforgeable.

2.2.2. Generic Compositions. A "generic composition" is a combination of a symmetric encryption scheme with a MAC algorithm in some way [12]. Let $\mathcal{SE}$ represent a symmetric encryption scheme with key K_e, specified by an encryption algorithm $\mathcal{E}$ and a decryption algorithm $\mathcal{D}$. Assume $\mathcal{MA}$ represents a message authentication scheme with key K_a, specified by a generation algorithm $\mathcal{T}$ and a verification algorithm $\mathcal{V}$. Let $\mathcal{M}$ represent a message that requires to be protected. The following compositions of $\mathcal{SE}$ and $\mathcal{MA}$ can create an authenticated encryption scheme $\mathcal{AE}$:

(i) *Encrypt-and-MAC*: $\mathcal{AE}_{K_e,K_a}(M) = \mathcal{E}_{K_e}(M) \parallel \mathcal{T}_{K_a}(M)$.

(ii) *MAC-then-encrypt*: $\mathcal{AE}_{K_e,K_a}(M) = \mathcal{E}_{K_e}(M \parallel \mathcal{T}_{K_a}(M))$.

(iii) *Encrypt-then-MAC*: $\mathcal{AE}_{K_e,K_a}(M) = C \parallel \mathcal{T}_{K_a}(C)$ where $C = \mathcal{E}_{K_e}(M)$.

Table 1 is obtained from the security analysis in [12]. Let $\mathcal{SE}$ be IND-CPA secure, and $\mathcal{MA}$ be SUF-CMA secure. Associated with them is $\mathcal{AE}$, an authenticated encryption scheme constructed according to one of the three generic compositions: *Encrypt-and-MAC*, *MAC-then-encrypt*, or *Encrypt-then-MAC*.

The three AE compositions are evaluated to prove whether they are secure under four different notions of security: IND-CPA, IND-CCA, INT-PTXT, and INT-CTXT. Formal demonstrations for the security of each scheme are provided in [12].

In this work, the *Encrypt-then-MAC* composition is used to provide confidentiality and authentication to the messages transmitted by the sensor node used as case study. The selected scheme is known to be secure under two assumptions: the underlying symmetric encryption scheme is IND-CPA secure, and the underlying MAC function is SUF-CMA secure.

2.2.3. Symmetric Encryption. For constrained environments, as in a WSN, reducing the number of algorithms required is one of many critical tasks. Hence, involutive cryptographic transformations are desired. An involutive symmetric cipher is one such that its encryption function $\mathcal{E}$ and its decryption function $\mathcal{D}$ are equivalent. Some Feistel constructions possess this property. However, this advantage implies security risks for block ciphers due to search space reduction. Involution is also found in operation modes such as CTR [23]. This encryption scheme enables encrypting and decrypting data using only the transformation $\mathcal{E}$ of the underlying symmetric cipher. Furthermore, the CTR operation mode is IND-CPA secure, as it has been demonstrated in [24, 25].

The CTR operation mode is illustrated in Figure 1. It has the advantage that it requires only the implementation of the encryption function $\mathcal{E}$ to both encrypt and decrypt a message M using a series of counters ctr and a secret key K_e. This effectively reduces the required resources for most implementations to half.

2.2.4. MAC Algorithms. A MAC is a tag computed from the message using a private key. As mentioned in [12], most practical MACs seem to be strongly unforgeable based on the assumption that they behave as a PRF. In this work we use CBCMAC [23] and HMAC [23] as MAC generation algorithms. The security of CBCMAC as a PRF is demonstrated in [24, 25] while the security of HMAC as a PRF is analyzed in [26].

CBCMAC is an operation mode that allows to use a symmetric cipher to generate MACs. This scheme is shown in Figure 2. It requires little additional resources to generate

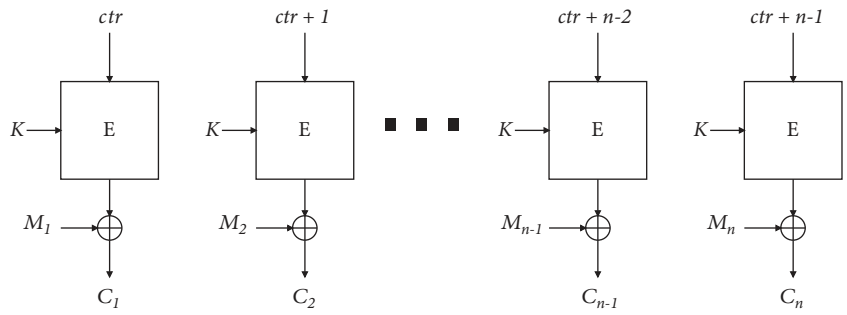

Figure 1: The CTR operation mode.

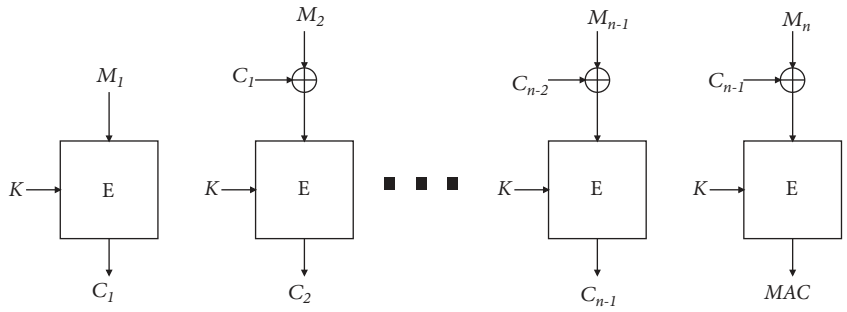

Figure 2: The CBCMAC operation mode.

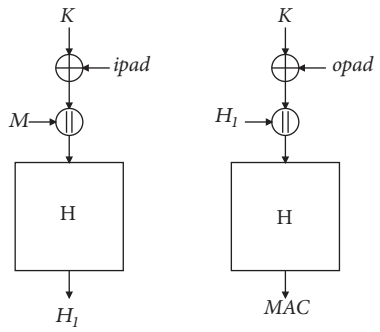

Figure 3: The HMAC operation mode.

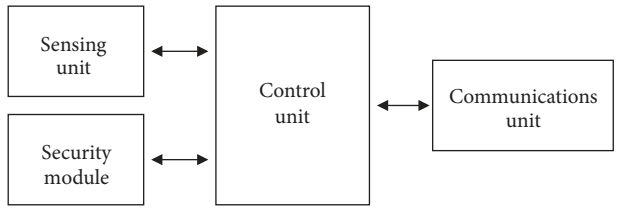

Figure 4: Components in the sensor node architecture.

a MAC with an underlying symmetric encryption scheme. Moreover, given the nature of MACs, it is only necessary to implement the encryption function of the block cipher. The strength of CBCMAC rests on two conditions: (i) the processed messages are of fixed length and (ii) the block size of the underlying cipher is adequate to represent the message space. Both of these considerations are present in our case study since we utilize fixed length messages, and the length of these messages is small.

HMAC was included in this study to evaluate the cost of hash algorithms in constrained environments. This operation mode generates the MAC of a message using a hash function and a private key as illustrated in Figure 3. In the literature it has been pointed out that hash functions are not suitable for constrained devices [13]; however, this claim has not been verified experimentally. Additionally, the emergence of lightweight hash functions reinforces the need to corroborate such assumption.

2.3. Sensor Node Architecture. The architecture of the sensor node prototype created includes a sensing unit and a communications unit. The *sensing unit* is intended to retrieve samples of a physical variation of its environment. All the collected data is transmitted to another entity using a *communications module*. Figure 4 illustrates the components considered in the sensor node architecture.

The sensor node includes a *control unit* capable of taking decisions, handling interruptions, exceptions, among others. These tasks can be managed by an automata or processor. For our case study, the automata is more convenient since it reduces resource consumption at the cost of reduced flexibility and longer development times. These, in contrast, are the main advantages of a processor. The selection of the most appropriate method will depend on the characteristics of the WSN.

Upon receiving a new message from the sensing unit, the automata is in charge of overseeing the processing and

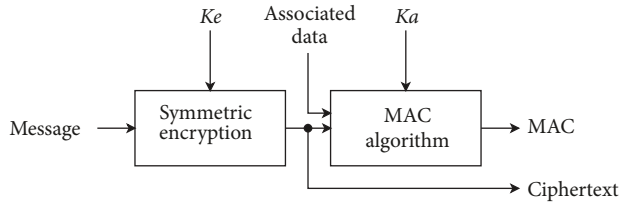

Figure 5: Architecture of the security module implementing the *Encrypt-then-MAC* composition. The message containing sensitive data is first encrypted using an encryption key to obtain a ciphertext, then the MAC is calculated for the ciphertext. Under a generic composition the associated data can be authenticated using the MAC as illustrated.

transmission of the message. In our study, the scenario where a message is received from a neighboring node was not considered.

In this sensor node architecture, the most critical component is the *security module*, developed to keep the sensor's transmission channel secure. This block implements the authenticated encryption composition *Encrypt-then-MAC*. It takes as input a message from the sensing unit and produces as output packets for the communication unit; these encapsulate the encrypted payload and a security header. Each configuration for the security module includes an automata to control the operation of the module, the encryption of the message, and the creation of the MAC. The architectures used to perform the encryption and calculating the MAC are discussed in detail below.

2.4. Security Module. The architecture of the security module is composed of two main elements: a submodule to encrypt/decrypt the message and a submodule in charge of generating and verifying the MAC of the message. In the case study proposed it is considered that the node only performs transmission tasks, which require only encryption and tag generation. The block diagram for the security module is illustrated in Figure 5.

2.4.1. Underlying Cryptographic Algorithms. The operation modes selected to provide confidentiality authentication require underlying block ciphers or hash functions. A general purpose algorithm and a lightweight alternative were studied for each type of cryptographic primitive.

Encryption Engine. Data confidentiality is achieved through encryption, provided by a block cipher in CTR operation mode. This scheme was implemented seeking minimal area overhead, mainly determined by the counter itself and an XOR layer. In the CTR mode, each consecutive counter's value is encrypted using an encryption algorithm and the result is then XOR-ed with the plaintext block. The result is the ciphertext.

Block Ciphers. The symmetric cipher algorithms used in the authenticated encryption scheme are AES and Present. In both cases the main design goal is area reduction, using a key size of 128 bits. AES was selected for being a well-known general purpose standard while Present was selected as the lightweight cipher to be used based on the results reported in the literature about its performance and implementation size [27].

Rijndael was proposed in 1998 by Joan Daemen and Vincent Rijmen and standardized as AES by NIST in 2001. Its a round based cipher built as a substitution-permutation network (SPN). Present was created in 2007 by Andrey Bogdanov et al. and standardized by ISO/IEC in 2012. Present also has a SPN structure and is based in rounds. In this work we evaluate the impact that these two ciphers have in the security module for the sensor node, from energy and area perspectives.

The hardware architectures for AES and Present used as the encryption engine in the security module are shown in Figure 6.

The AES architecture has a well-balanced trade-off between implementation size and performance. When the goal is to improve the energy consumption of a circuit it is not practical to sacrifice performance in aims of achieving minimal area footprint. Reduced latency and thus improved performance play a major role in reducing the energy usage.

The Present design also explores these ideas, featuring a reduced datapath architecture that does not compromise the latency of the datapath to the limit. The optimization in that architecture was carefully developed to carry out only the most effective area-reduction strategies. The AES architecture used in this work was derived from the implementation in [28] while the Present design utilized is the one reported in [27].

In the Present design used, the aim is to reduce the datapath width considering both the substitution layer (sBoxLayer) and the permutation layer (pLayer). In the reduction of the substitution layer the datapath can be adjusted to any width divisible by four. The reduction of width in the permutation layer is achieved thanks to a pattern in the structure of the function itself. That strategy consists in exploiting the regularity of the 64-bit permutation. Using said pattern it is possible to shrink down the width of the permutation layer from 64-bit to 16-bit. With that reduction, the substitution layer can take a width of 16-bit too, thus requiring only four substitution boxes.

As can be seen from Figure 6 only two data ports are needed in this design, 16 bits taken as the input (data_in), and 16 bits taken as the output (data_out). To input the data, 4 cycles are consumed, 124 cycles are then used to process the state, and finally 4 more cycles are required to produce the output, giving a total latency of 132 cycles.

The key schedule of the architecture works by recording all the keying materials in a module and allowing the synthesis tool to generate the combinational design that produces the round keys. This is interesting for this specific design since the width of the round key is reduced from 64 to 16 bits, which enables a reduction in the complexity of the combinational process. Under this approach, it is required to calculate the whole key set presynthesis and to describe it as a memory block. When the FPGA can not use memory blocks to implement this module, the synthesizer will be forced to use LUTs to create a combinatorial block capable of

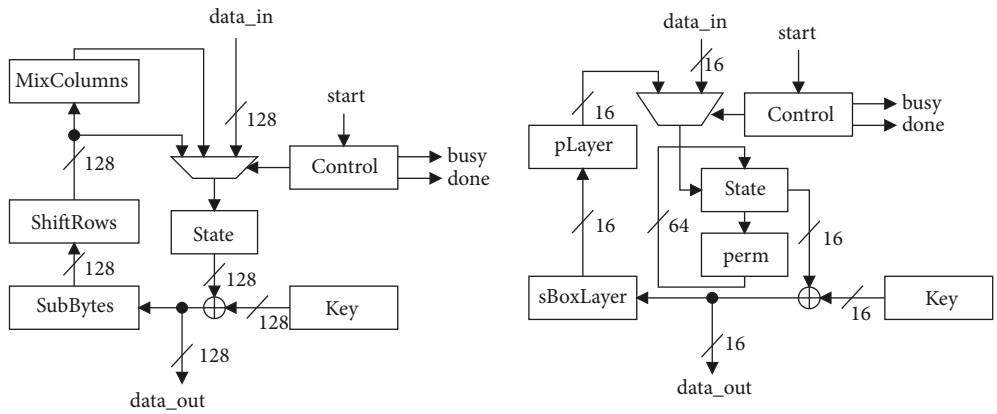

Figure 6: Block ciphers used to implement the authenticated encryption scheme. To the left, the architecture of AES; to the right, the architecture of Present.

generating each one of the round keys required by the cipher [29].

Authentication Engine. Authentication is guaranteed using a MAC. The final message packet contains the data and the MAC tag; hence data expansion is one of the downsides of the authenticated encryption method used. However, a MAC provides strong authentication, which is useful for messages with low lexicographic content, such the ones transmitted in WSN.

The keyed MAC algorithms selected use a key size of 128 bits. The implementation of CBCMAC requires an XOR layer at the input of the underlying block cipher. In an area optimized implementation, a single encryption core can process all the message, which is divided into several blocks. When the last block is processed, the result is the MAC of the message. In the implementation of HMAC two inner registers and an XOR layer are required, additionally to the hash core. The input message is divided into blocks of length determined by the underlying hash function. To reduce resource usage and utilize a single hash core it is necessary to utilize an extra register to store the intermediate results (H_1 in Figure 3). When the last message block has been processed, the MAC is available at the output of the hash module.

Hash Functions. We selected SHA-3 and spongent as the underlying hash functions for implementing the HMAC core. Of the SHA-3 family, SHA3-256 was used to test its efficiency in the presented case study. A review of the literature for lightweight hash functions revealed that the one with the smallest implementation sizes reported is the algorithm spongent; hence it was chosen.

Keccak was created in 2008 by Guido Bertoni et al. and standardized as SHA-3 in 2015 by NIST. This hash function was the first to use a sponge construction with a set of permutations as its internal function. SHA-3 is a family of four algorithms with hash sizes of 224, 256, 384, and 512 bits. Andrey Bogdanov et al. proposed spongent in 2011. This hash function follows a sponge construction with an internal function built as a Present-like SPN. This is also a family of five algorithms with hash lengths of 88, 128, 160, 224, and 256 bits.

SHA3-256 provides as default an output of 256 bits. This output is generated as 64-bit blocks, and according to NIST the MAC can be truncated to fit the application. Following the recommendations in [13] it was determined to use MACs of 64-bit length. spongent features different configurations for outputs of varying length, of which the one with length of 88-bit was selected. The disadvantage of this design is the number of rounds required to produce a digest, which has negative effects on the performance of the solution. The architectures for the hash functions utilized are shown in Figure 7.

The authors of SHA-3 developed three hardware implementations of the algorithms: a high performance core, a middle-range core, and a minimum area coprocessor [30]. The middle-range core is the best alternative for exploring area/performance trade-offs. The design is parametrized, which allows modifying the datapath width and achieving reduced area at the cost of increased latency. We empirically determined that the configuration which processes one-quarter of the state in parallel is the most adequate as regards area/latency ratio.

Reduced area is important to implement generic algorithms in WSN; however the performance cannot be compromised if the goal is to improve the energy consumption of the circuit. The spongent architecture used in this work was created in a straightforward manner. While this algorithm is lightweight by design, these results could be improved with area minimization strategies. However, also by design, the latency of spongent is high which detriments the use of area-reducing optimizations based on datapath reduction. This design differs from the one in [31] in that the architecture in this work includes the controls and registers required for the use in the HMAC mode.

The architecture developed for spongent-88 is based on a hardware loop with a single register. In the initial phase the state value is the all zeros word. At the first round the input block is XOR-ed with the value in the register. At each round the register is then XOR-ed with the contents of an LFSR and processed by the substitution and permutation layers.

The LFSR counter is generated by a 6-bit register clocked each active round. Its value is XOR-ed with the least significant bits of the state and its reversed value is XOR-ed

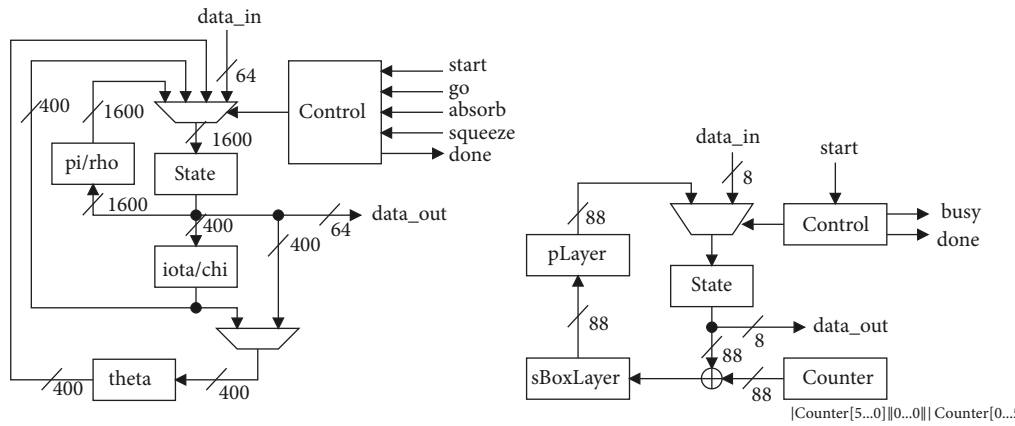

Figure 7: Hardware architectures for SHA3-256 (left) and spongent (right) used under the HMAC operation mode to generate MAC tags.

Table 2: Underlying cryptographic algorithms utilized in the different security module configurations.

Label	Algorithm	Type	Class	Operation mode	Service provided
AES-128	AES	Symmetric cipher	Generic	CTR	Confidentiality
				CBCMAC	Authentication
PRE-128	Present	Symmetric cipher	Lightweight	CTR	Confidentiality
				CBCMAC	Authentication
SHA-256	Keccak	Hash function	Generic	HMAC	Authentication
SPO-88	spongent	Hash function	Lightweight	HMAC	Authentication

with the state's most significant bits. The substitution layer is formed by 22 4-bit substitution boxes which process the state in parallel. The permutation layer is a simple wiring which can be straightforwardly implemented with little cost. The output is taken directly from the output of the register. To reduce the switching activity at the output, we opted to include a mask, thus improving the energy consumption at the cost of a few additional hardware resources.

2.5. Methods. This section describes the experimental method followed in this work, including the experimental setup, the configurations for the experiments, and the evaluation metrics.

2.5.1. Environment. The sensor node prototype presented in Section 2.4 was described using VHDL and implemented in a Xilinx Spartan-6 XC6LX16-CS324 FPGA. The prototype was equipped with a battery system to be autonomous. The study of the cryptographic primitives and their impact in the security module operation was carried out using this system as computing platform.

A lead acid battery of 6V and 1Ah was used in our experimentation. The FPGA was connected to the power supply through the regulators included in the Nexys 3 development board.

The operation of the sensing unit was emulated using a set of messages retrieved from a public available database of Intel (http://db.csail.mit.edu/labdata/labdata.html). This database contains messages from 54 sensor nodes deployed in the Intel Berkeley Research lab between February 28 and April 5, 2004. The motes utilized collected timestamped topology information, along with humidity, temperature, light, and voltage values every 31 seconds. The database includes a log of about 2.3 million readings collected from these sensors.

The communications unit was configured as a driver for 4215A XBee board using serial protocol in the data transmission from the FPGA to the XBee card. This module was set to receive and transmit data as 64-bit words which were then partitioned in bytes and transmitted via wireless communication.

To transmit messages, a basic scenario was considered where the sensor node sends the data directly to a base station using the XBee card connected to the FPGA. To reproduce identical conditions for all of the experiments the sensing unit was emulated using messages taken from the public database. From the database, 3600 messages corresponding to the sensor node with id=1 were extracted and converted to fixed point notation. The codification produced messages with constant length of 144 bits, which were stored in the sensor node implemented. The sensing unit was configured to produce as output one of these messages each second. The components of the environment are illustrated in Figure 8.

2.5.2. Configurations. The security module in the sensor node was implemented under different configurations, created from compositions of the cryptographic primitives shown in Table 2. Five configurations were studied; all of them follow the *Encrypt-then-MAC* paradigm.

The configurations derived from the use of these algorithms are described as follows:

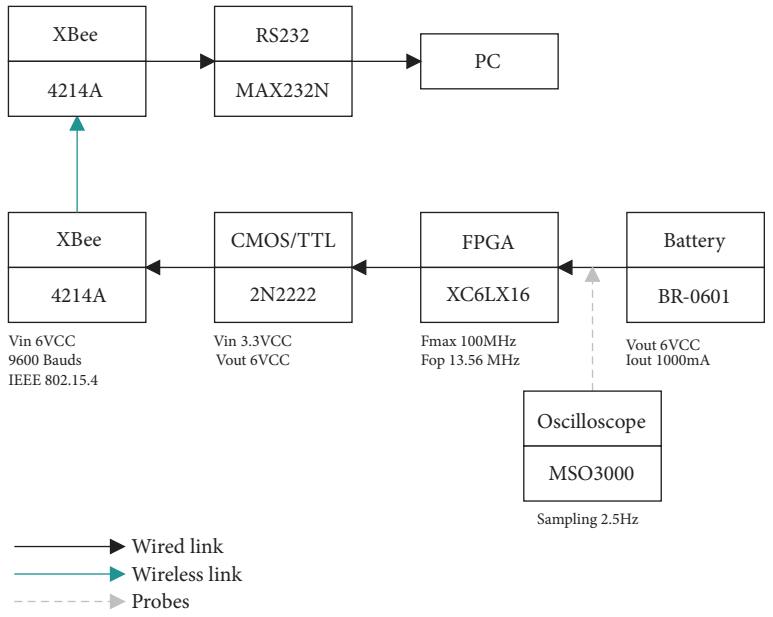

Figure 8: Experimental environment.

(1) No security services (C0). In this case the sensor node operates without providing security services. The results from this configuration can be used as a reference to measure the security-related overhead of the other configurations.

(2) Generic cipher (C1). In this configuration CTR was used to encrypt the data using AES-128, and CBC-MAC was constructed using AES-128 as well.

(3) Generic cipher and hash (C2). The message is encrypted with CTR mode using AES-128 and the MAC is generated with HMAC using SHA-256.

(4) Lightweight cipher (C3). This configuration uses CTR with PRE-128 to encrypt the message and CBCMAC with PRE-128 to generate the MAC.

(5) Lightweight cipher and hash (C4). The message is encrypted with PRE-128 in CTR mode and HMAC with SPO-88 is used to obtain the MAC.

All the previous configurations derived on a different construction for the security module in the sensor node under study. The modularity in the FPGA implementation was exploited to achieve easy replacement of each building block described with VHDL.

2.5.3. Metrics. The resource usage was studied using the FPGA units of slices (SLC), Look-Up Tables (LUT), and Flip-Flops (FF). The performance was studied as latency (LAT) and throughput (Thr). The different architectures were implemented using the ISE Design Suite System Edition 14.7 and the power analysis was performed using Xilinx XPower Analyzer. The synthesis processes were configured with *area* as optimization goal and *high* as optimization effort, while the usage of block RAMs and ROMs was disabled. The results were recorded after the place and route process.

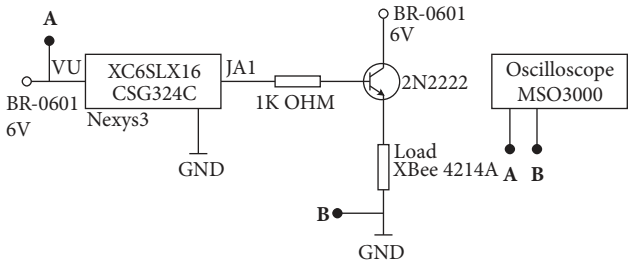

Figure 9: Experimental setup.

The life span of the sensor node is used to evaluate the energy consumption associated with each authenticated encryption composition under study. Each design was configured in the FPGA and set to operate until the energy of the battery ran out. The voltage level was monitored during the whole process and registered using a digital oscilloscope. The experimental setup is shown in Figure 9.

3. Results and Discussion

This sections presents our experimental results and our analysis derived from the data.

3.1. Results. Table 3 provides the area and power results for the different sensor node configurations. These architectures include the algorithms presented in Table 2 with the operational modes specified to provide security through authenticated encryption compositions. FPGA resources are provided in FF, LUT, and SLC after place and route. The latency cycles are reported for the encryption process (ENC), the calculation of the MAC, and the transmission of the message (COM).

Table 3: Implementation results for the different sensor node configurations. The XC6LX16-CS324 FPGA was used as implementation target with an operational frequency of 13.56 MHz.

Configuration	FPGA resources			Latency (Cycles)			t (ms)	POW (mW)	ENE (mJ)
	FF	LUT	SLC	ENC	MAC	COM			
C0	256	398	137	0	0	271200	20.00	23.19	0.4638
C1	2569	3944	1379	84	84	542400	40.01	27.27	1.0911
C2	3655	5033	1728	84	125	542400	40.01	31.01	1.2409
C3	884	1694	516	396	396	361600	26.72	23.63	0.6315
C4	1198	1899	571	396	3770	361600	26.97	23.45	0.6325

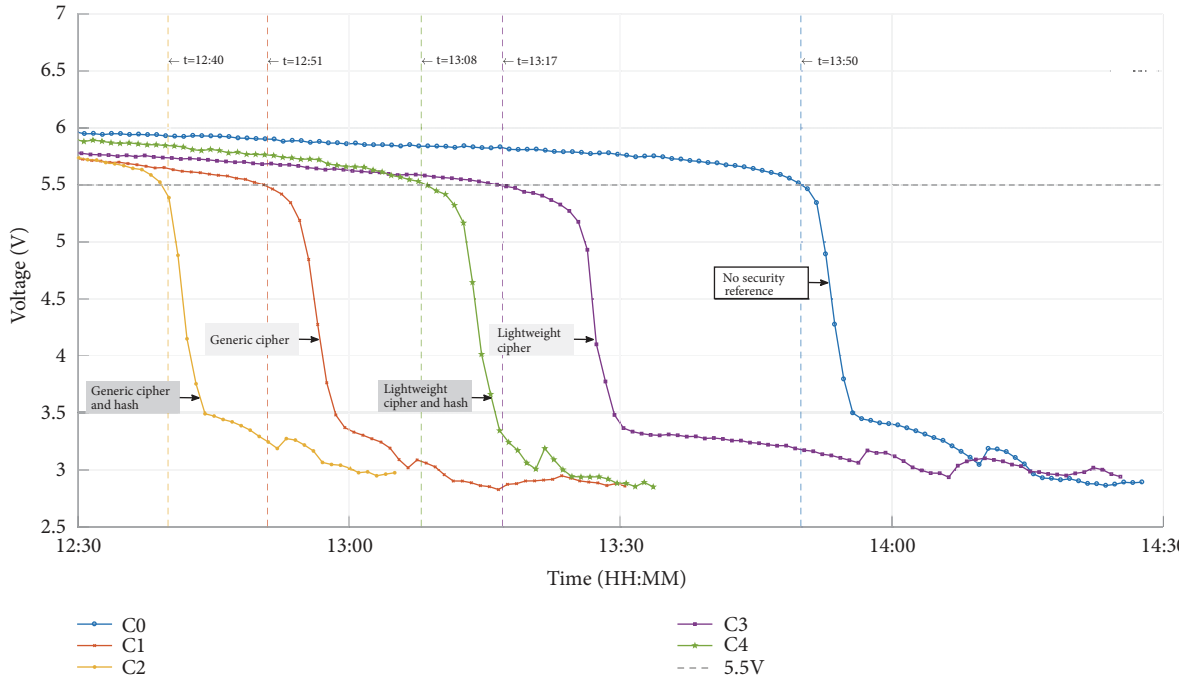

Figure 10: Operation time for the implemented sensor node configurations. It is important to remark that our experiment is a case study, it is expected that the lifetime of a real-world application would be longer, and thus the time differences would be greater. The sensor node prototypes were configured on the XC6LX16-CS324 FPGA embedded in the Nexys 3 development board. A 6V and 1Ah lead acid battery was used as power source. An XBee transmitter was connected to the FPGA board and also sourced from the 6V battery.

Figure 10 illustrates the voltage recordings for the different configurations of the sensor node. The horizontal axis represents the operation time since the sensor node was started. The vertical axis represents the voltage level of the battery recorded using a digital oscilloscope. The life span (t) is determined at the point where the voltage level drops below the operational threshold of 5.5V specified in the FPGA datasheet. Figure 10 has been adjusted to show the last couple hours of the experiment, which allows the reader to appreciate the point where the voltage for all the configurations drops below the operational threshold.

3.2. Discussion. The first important observation is that the energy estimations provided in Table 3 are consistent with the experimental results for the lifetime of the sensor node from Figure 10. An important remark is that even though the power is reduced thanks to area savings in the lightweight designs, the communications latency is the most significant contributor to the energy expenditure of the device. Another note to make is that even though there is a relation between the results in Table 3 and Figure 10, it is not proportional. Table 3 provides energy estimations for the time where the node is actively processing data, but most of the time the sensor node is idle and spending energy without performing any task. We believe this is the reason why, in Figure 10, the difference in the mote lifetime from one configuration to another is not more significant.

Our experiments were all conducted under the same conditions to achieve a fair comparison. However, the hours rate for the lead acid battery used was not specified in the product. This can lead to obtain different results if our experiment is replicated with the same conditions described in this work. Nonetheless we would expect that the behavior observed in Figure 10 can be duplicated easily. On a similar note, the fact that the hours rate for the battery used and its Peukert exponent are unknown limits us from estimating the lifespan of the sensor node based on its operational current.

Table 4: Reduction on the lifetime of the different configurations of the sensor node prototype.

Configuration	Impact
C0	0%
C1	7.1224%
C2	8.4090%
C3	3.9808%
C4	4.9935%

Several conclusions can be drawn from our experimentation concerning the life span of the sensor node configurations. As it was expected, the configuration which did not provide security for the messages (C0) reported the longest active time of the sensor node. This design (C0) achieved an active time of 13 hours and 50 minutes, which will be used as reference in further discussion.

If we group the configurations C1 (generic cipher) and C3 (lightweight cipher) and compare them to the configurations C2 (generic cipher and generic hash) and C4 (lightweight cipher and lightweight hash) it can be noted how the use of HMAC over CBCMAC has a negative impact on the life span of the prototype. This comparison can be appreciated in Figure 10. The advantages of using hash functions to generate MACs include elevated collision, preimage, and second preimage resistances. Nonetheless, since these algorithms require more cycles to generate a MAC, the energy expenditures are also increased. The cost on the life span of our case study associated with using HMAC over CBCMAC is 1.3% in average, which can be considered acceptable given the additional cryptographic strength associated with the former.

Now, compare the configurations C1 and C2 (generic algorithms) to the configurations C3 and C4 (lightweight algorithms). In this scenario the advantage of using lightweight algorithms is demonstrated. Compare C1 to C3 as shown in Figure 10, in both cases CTR is used to encrypt the data, and CBCMAC is used to generate the MAC. The difference in the life span of the configurations can be attributed to the use of Present over AES. Compare C2 to C4 as shown in Figure 10, in these instances the security services are provided by CTR and HMAC, and it is clear that the difference lies in the underlying algorithms. It can be concluded that the use of lightweight algorithms generates reduced area and energy overheads in the system. The average cost in the life span of using generic algorithms over lightweight solutions for our case study is in average 3.4%.

The impact of each authenticated encryption configuration on the life span of the sensor node utilized as study case is detailed in Table 4.

Regarding implementation area and performance, the results reveal that lightweight algorithms offer advantages in area by making performance trade-offs. These design considerations need to be reviewed carefully in the design of lightweight algorithms. While minimizing the resource usage and ergo the production costs of the system may be a primordial goal, this optimization should not represent great compromises to the performance of the algorithms. The performance impacts directly on the runtime of the architecture, if the architecture expends longer periods active then this affects the energy consumption. Energy-aware solutions require adequate compromises between the implementation size (to reduce the number of elements that must be powered) and the performance (to reduce the total time these elements must be powered on).

3.3. Comparisons. We were unable to find hardware implementations of authenticated encryption solutions in FPGA. For this reason we cannot provide a quantitative comparison with other works. What we provide, however, is a comparison in terms of qualitative characteristics, as well as quantitative comparisons, to some degree, of the underlying lightweight primitives used.

Table 5 presents our assessment of the characteristics for the different authenticated encryption solutions reviewed from the literature. The column "Calls" refers to the performance for the different schemes. As it can be appreciated, although modern operation modes are efficient they do not provide support for associated data (AD) or would not serve our purposes of evaluating the costs of MACs based on hash functions.

Multiple block ciphers are reported in the literature which could be used instead of Present. For example, in [32, 33] the authors specify the Simon and Speck families of block ciphers. The authors claim to balance their work around simplicity, security, and flexibility. As pointed out, both ciphers have good performance in several lightweight applications, but Simon is tuned for optimal performance in hardware and Speck for optimal performance in software. Based on this we focus our comparison in Simon. In [32], ten configurations of Simon are described; however we shall select the ones that match the Present version used, which has block size of 64 bits and key of 128 bits. It is important to match these criteria for fairness, since block size and cipher size determine the strength of a block cipher.

Our first criteria for selecting Present were its exceptional implementation size in ASIC. At first glance, the implementation results in [32] appear to outperform Present by about 25% (1000 GE versus 1339 GE in a 130 nm process). However, as the authors point out "[their] current hardware designs have not proceeded past the synthesis stage", which implies that they should be taken with caution. In contrast, the work in [34] provides actual implementation results, which point out the difference between Present and Simon for equivalent parameters is of about 7% (1458 GE versus 1560 GE in a 90 nm process). Moreover, the Simon 64/128 "area-minimizing implementations" provided in [32] do not appear to be effective (1000GE reduced to 958 GE). In our opinion this is due to the nature of the Feistel network used in the block cipher.

The second aspect to consider on the selection of a block cipher is its latency. As the works in [16, 17] point out, the delay in the processing of information by a mote is critical. For equivalent parameter sizes, Present requires 31 cycles, while Simon requires 44; this would carry over to the respective optimized implementations. This is another advantage for Present, since smaller latency implies higher

Table 5: Qualitative comparison of our proposed solutions with works from the literature.

Scheme	Ref.	Privacy	Integrity/Auth.	Supports AD	Patented	Calls
CCM	[11]	CTR	CBC-MAC	Yes	No	$2n$
GCM	[20]	CTR	GHASH	Yes	No	$2n$
OCB	[21]	Cipher	Checksum	Yes	Yes	$n+2$
SOSEMANUK-MAC	[22]	Stream cipher	Dragon-MAC	Yes	No	$2n$
HC 128-MAC	[22]	Stream cipher	Dragon-MAC	Yes	No	$2n$
Rabbit-MAC	[22]	Stream cipher	Dragon-MAC	Yes	No	$2n$
TinyAEAD	[19]	Cipher	MMO	Yes	No	$n+m$
PFC-CTR	[14, 15]	CTR	PFC	No	No	$n+1$
PFC-OCB	[14, 15]	OCB	PFC	No	No	$n+1$
IAR-CTR	[16]	CTR	PFC	No	No	$n+1$
IAR-CFB	[16]	CFB	PFC	No	No	$n+1$
RT-OCFB	[17]	OCFB	No	No	No	n
PFX-CTR	[18]	CTR	PFC	No	No	$n+1$
PFX-INC	[18]	CTR∗	PFC	No	No	$n+1$
CTR-CBCMAC	This work.	CTR	CBC-MAC	Yes	No	$2n$
CTR-HMAC	This work.	CTR	HMAC	Yes	No	$n+m$

n represents the number of message blocks; m represents the number of hash calls.
∗The authors indicate that this is an increment similar to the one used in GCM or CCM∗.

performance and lower energy consumption, and from that longer lifetime which is our ultimate goal.

Regarding the hash function selected, to the best of our knowledge, the basic implementation of SPONGENT-88 has the smallest results reported for both ASIC [35] and FPGA [31].

4. Conclusions

In this paper we have studied different alternatives to provide authenticated encryption for WSN applications under the *Encrypt-then-MAC* generic composition. We evaluated the impact of providing confidentiality, integrity, and authentication, in terms of energy consumption and area of the underlying cryptographic algorithms selected.

In our study, we considered general purpose and lightweight cryptographic algorithms to implement the building blocks of the *Encrypt-then-MAC* generic composition. Four different configurations of the security module of a sensor node prototype were evaluated in an FPGA. The overhead imposed by these compositions in the life span of the mote was quantified. This impact was associated with the energy consumption of a dedicated security module in the sensor node prototype.

As underlying cryptographic primitives for the *Encrypt-then-MAC* construction we analyzed different alternatives. The symmetric cipher AES was utilized to study the costs of using generic block ciphers. The hash function KECCAK was utilized to assess the impact of generic hash functions on constrained environments. The algorithms PRESENT and SPONGENT were studied to quantify the advantages of using lightweight cryptography.

From our experiments we observed that providing authenticated encryption through generic compositions represents an impact of ∼-6% in the lifetime of the sensor node, in the average. This is in line with our first contribution enumerated.

Our experimentation also demonstrated that the use of lightweight algorithms to enable authenticated encryption has favorable effects on the implementation size (∼-65% SLC in average) and lifetime (∼+3.4% in the average) of our WSN mote. Our second contribution is outlined with these findings. In particular, the use of a lightweight cipher under CTR mode to encrypt data and under CBCMAC mode to generate MACs achieved the best results.

The provided results also show evidence that using hash functions to generate MACs under the HMAC operation mode is less efficient than using block ciphers under the CBCMAC operation mode for the same task. This held true for both the generic and lightweight instances with impacts of ∼-1.3% on the lifetime of the sensor node and ∼+15% on the SLC count, in the average. Although the increased costs could discourage the use of hash functions, under some circumstances their added security benefits might outweigh the associated costs. With these results we conclude our third contribution.

Data Availability

The experimental data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by CONACyT [Grants nos. 393070 and 336750] and CINVESTAV. This work was also funded by "Fondo Sectorial de Investigación para la Educación", CONACyT Mexico, through the Project no. 281565.

References

[1] M. Luo, Y. Luo, Y. Wan, and Z. Wang, "Secure and Efficient Access Control Scheme for Wireless Sensor Networks in the Cross-Domain Context of the IoT," *Security and Communication Networks*, vol. 2018, 2018.

[2] T. Rault, A. Bouabdallah, and Y. Challal, "Energy efficiency in wireless sensor networks: a top-down survey," *Computer Networks*, vol. 67, pp. 104–122, 2014.

[3] B. C. Cheng, G. T. Liao, R. Y. Tseng, and P. H. Hsu, "Network lifetime bounds for hierarchical wireless sensor networks in the presence of energy constraints," *Computer Networks*, vol. 56, no. 2, pp. 820–831, 2012.

[4] D. Bruneo, S. Distefano, F. Longo, A. Puliafito, and M. Scarpa, "Evaluating wireless sensor node longevity through Markovian techniques," *Computer Networks*, vol. 56, no. 2, pp. 521–532, 2012.

[5] X. Zhang, H. M. Heys, and C. Li, "Energy efficiency of encryption schemes applied to wireless sensor networks," *Security and Communication Networks*, vol. 5, no. 7, pp. 789–808, 2012.

[6] K. Bok, Y. Lee, J. Park, and J. Yoo, "An energy-efficient secure scheme in wireless sensor networks," *Journal of Sensors*, vol. 2016, 2016.

[7] M. Elhoseny, X. Yuan, H. K. El-Minir, and A. M. Riad, "An energy efficient encryption method for secure dynamic WSN," *Security and Communication Networks*, vol. 9, no. 13, pp. 2024–2031, 2016.

[8] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks".

[9] "IEEE standard for local and metropolitan area networks–part 15.4: Low-rate wireless personal area networks (LR-WPANs)".

[10] "Specification for the Advanced Encryption Standard (AES)," in *Federal Information Processing Standards Publication*, vol. 197, 2001.

[11] M. J. Dworkin, "Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality," National Institute of Standards and Technology NIST SP 800-38c, 2007.

[12] M. Bellare and C. Namprempre, "Authenticated encryption: relations among notions and analysis of the generic composition paradigm," *Journal of Cryptology. The Journal of the International Association for Cryptologic Research*, vol. 21, no. 4, pp. 469–491, 2008.

[13] Z. Gong, P. Hartel, S. Nikova, and B. Zhu, "Towards Secure and Practical MACs for Body Sensor Networks," in *Progress in Cryptology - INDOCRYPT 2009*, vol. 5922 of *Lecture Notes in Computer Science*, pp. 182–198, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[14] T. Hwang and P. Gope, "PFC-CTR, PFC-OCB: Efficient stream cipher modes of authencryption," *Cryptologia*, vol. 40, no. 3, pp. 285–302, 2016.

[15] T. Hwang and P. Gope, "Robust stream-cipher mode of authenticated encryption for secure communication in wireless sensor network," *Security and Communication Networks*, vol. 9, no. 7, pp. 667–679, 2016.

[16] T. Hwang and P. Gope, "IAR-CTR and IAR-CFB: Integrity aware real-time based counter and cipher feedback modes," *Security and Communication Networks*, vol. 8, no. 18, pp. 3939–3952, 2015.

[17] T. Hwang and P. Gope, "RT-OCFB: Real-Time Based Optimized Cipher Feedback Mode," *Cryptologia*, vol. 40, no. 1, pp. 1–14, 2016.

[18] T. Hwang and P. Gope, "PFX: An essence of authencryption for block-cipher security," *Security and Communication Networks*, vol. 9, no. 10, pp. 1186–1197, 2016.

[19] A. A. Adekunle and S. R. Woodhead, "An AEAD cryptographic framework and TinyAEAD construct for secure WSN communication," in *Proceedings of the 2012 Wireless Advanced, WiAd 2012*, pp. 1–5, UK, June 2012.

[20] M. J. Dworkin, "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC," National Institute of Standards and Technology NIST SP 800-38d, 2007.

[21] P. Rogaway, M. Bellare, and R. S. Ferguson, "OCB: a block-cipher mode of operation for efficient authenticated encryption," *ACM Transactions on Information and System Security*, vol. 6, no. 3, pp. 365–403, 2003.

[22] S. Ahmad, A. Wahla, and F. Kausar, "Authenticated Encryption in WSN Using eSTREAM Ciphers," in *Advances in Information Security and Assurance*, vol. 5576 of *Lecture Notes in Computer Science*, pp. 741–749, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[23] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Boca Raton, FL, USA, 1st edition, 1996.

[24] M. Bellare and P. Rogaway, "Introduction to Modern Cryptography," http://web.cs.ucdavis.edu/ rogaway/classes/227/spring05/book/main.pdf.

[25] P. Rogaway, "Evaluation of Some Blockcipher Modes of Operation," http://web.cs.ucdavis.edu/ rogaway/papers/modes.pdf.

[26] M. Bellare, "New proofs for NMAC and HMAC: security without collision resistance," *Journal of Cryptology. The Journal of the International Association for Cryptologic Research*, vol. 28, no. 4, pp. 844–878, 2015.

[27] C. A. Lara-Nino, M. Morales-Sandoval, and A. Diaz-Perez, "Novel FPGA-Based Low-Cost Hardware Architecture for the PRESENT Block Cipher," in *Proceedings of the 19th Euromicro Conference on Digital System Design, DSD 2016*, pp. 646–650, Cyprus, September 2016.

[28] INMCM, *Basic AES-128 Encryption in VHDL Complete*, 2010, https://inmcm.wordpress.com/2010/03/14/basic-aes-128-encryption-in-vhdl-complete/.

[29] C. A. Lara-Nino, A. Diaz-Perez, and M. Morales-Sandoval, "Lightweight Hardware Architectures for the Present Cipher in FPGA," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 9, pp. 2544–2555, 2017.

[30] G. Bertoni, J. Daemen, M. Peeters, and G. van Assche, "Keccak implementation overview. Version 3.2," https://keccak.team/files/Keccak-implementation-3.2.pdf.

[31] C. A. Lara-Nino, M. Morales-Sandoval, and A. Diaz-Perez, "Small lightweight hash functions in FPGA," in *Proceedings of the 2018 IEEE 9th Latin American Symposium on Circuits & Systems (LASCAS)*, pp. 1–4, Puerto Vallarta, Feburary 2018.

[32] R. Beaulieu, S. Treatman-Clark, D. Shors, B. Weeks, J. Smith, and L. Wingers, "The SIMON and SPECK lightweight block ciphers," tech. rep, 2013.

[33] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK Lightweight Block Ciphers," in *in Proceedings of the 52nd Annual Design Automation Conference, DAC'15*, vol. 6, 175 pages, New York, NY, USA, 2015.

[34] S. Banik, S. K. Pandey, T. Peyrin, Y. Sasaki, S. M. Sim, and Y. Todo, "GIFT: A Small Present," in *Cryptographic Hardware and Embedded Systems – CHES 2017*, vol. 10529 of *Lecture Notes in Computer Science*, pp. 321–345, Springer International Publishing, Cham, 2017.

[35] A. Bogdanov, M. Knežević, G. Leander, D. Toz, K. Varıcı, and I. Verbauwhede, "spongent: A Lightweight Hash Function," in *Cryptographic Hardware and Embedded Systems – CHES 2011*, vol. 6917 of *Lecture Notes in Computer Science*, pp. 312–325, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.