Gong Z, Hartel P, Nikova S *et al.* TuLP: A family of lightweight message authentication codes for body sensor networks. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 29(1): 53–68 Jan. 2014. DOI 10.1007/s11390-013-1411-8

TuLP: A Family of Lightweight Message Authentication Codes for Body Sensor Networks

Zheng Gong[1] (龚 征), Pieter Hartel[2], Svetla Nikova[3], Shao-Hua Tang[4,*] (唐韶华), *Member, IEEE* and Bo Zhu[5] (朱 博)

[1] *School of Computer Science, South China Normal University, Guangzhou 510631, China*

[2] *Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, Enschede 7500AE The Netherlands*

[3] *Department of ESAT/SCD-COSIC, Katholieke Universiteit Leuven, Leuven, Belgium*

[4] *School of Computer Science and Engineering, South China University of Technology, Guangzhou 510641, China*

[5] *Department of Electrical and Computer Engineering, University of Waterloo, Waterloo N2L 3G1, Canada*

E-mail: cis.gong@gmail.com; pieter.hartel@utwente.nl; svetla.nikova@esat.kuleuven.be; shtang@ieee.org; zhubo03@gmail.com

Received January 25, 2013; revised August 16, 2013.

Abstract A wireless sensor network (WSN) commonly requires lower level security for public information gathering, whilst a body sensor network (BSN) must be secured with strong authenticity to protect personal health information. In this paper, some practical problems with the message authentication codes (MACs), which were proposed in the popular security architectures for WSNs, are reconsidered. The analysis shows that the recommended MACs for WSNs, e.g., CBC-MAC (TinySec), OCB-MAC (MiniSec), and XCBC-MAC (SenSec), might not be exactly suitable for BSNs. Particularly an existential forgery attack is elaborated on XCBC-MAC. Considering the hardware limitations of BSNs, we propose a new family of tunable lightweight MAC based on the PRESENT block cipher. The first scheme, which is named TuLP, is a new lightweight MAC with 64-bit output range. The second scheme, which is named TuLP-128, is a 128-bit variant which provides a higher resistance against internal collisions. Compared with the existing schemes, our lightweight MACs are both time and resource efficient on hardware-constrained devices.

Keywords message authentication code, body sensor network, low-resource implementation

1 Introduction

Nowadays, wireless sensor networks (WSNs) are more and more implemented to collect environmental information, e.g., temperature, humidity, and fire alarm. For realizing the Ambient Assisted Living (AAL) vision①, body sensor networks (BSNs, also called wireless medical sensor networks)[1] have attracted more attention for healthcare applications. Although the fact that large groups of patients already carry individually implantable or wearable monitoring equipments, a BSN offers a more accurate status than one isolated device. To offer more personalized healthcare to elderly or disabled patients, a BSN can instantly send personal health information to the server of a clinic or hospital. The gathered information will be monitored by doctors (or nurses) to prevent the occurrence of fatal events. Existing examples include CodeBlue[2], ALARM-NET[3], and DexterNet[4]. A system model of typical BSNs is illustrated in Fig.1.

Although BSNs are built from wireless sensor nodes, the system model of BSNs decides they are more restrictive in the aspect of resources. Firstly, since BSNs are either worn or implanted by patients, the batteries of body sensors should be as small as possible because they determine how "hidden" and "pervasive" the sensors are. Secondly, frequent battery changing or recharging activities are not acceptable for patients.

Regular Paper

This work is supported by the National Foundation of Netherlands with SenterNovem for the ALwEN project under Grant No. PNE07007, the National Natural Science Foundation of China under Grant Nos. 61100201, U1135004, and 61170080, the Universities and Colleges Pearl River Scholar Funded Scheme of Guangdong Province of China (2011), the High-Level Talents Project of Guangdong Institutions of Higher Education of China (2012), the Project on the Integration of Industry, Education and Research of Guangdong Province of China under Grant No. 2012B091000035, and the Project of Science and Technology New Star of Guangzhou Pearl River of China (2014).

*Corresponding Author

① European Union. http://ec.europa.eu/information_society/activities/einclusion/docs/ageing/aal_feb10_olsson.pdf, August 2013.

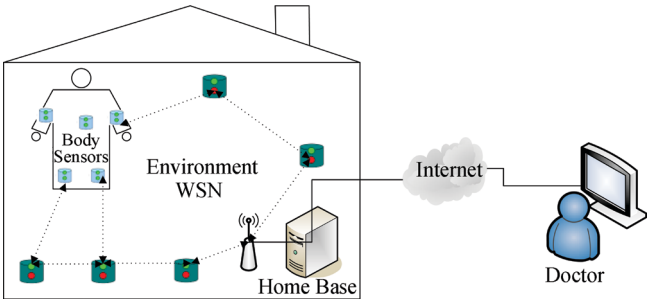

Fig.1. System model of a typical BSN.

Due to the above reasons, highly resource-constrained nodes are widely chosen for achieving energy-efficiency and lightweight. Table 1 shows the hardware specifications of typical BSN nodes used in practice. Besides the common usages of BSN nodes, the computational costs of the algorithms or protocols run on the nodes also must be as low as possible for the durability of BSN.

In WSNs, people usually accept low-level security requirements as trade-off of resource efficiency. However, BSNs are managed to monitor users' daily activities and health data, confidentiality and authenticity problems attract more concerns than WSNs. From the view of hospitals, it is the first priority that the BSN data should be collected from each patient with authenticity, so doctors can make a right decision on the exact case. Unfortunately, because of the heterogeneity of BSNs, the cryptographic schemes for static networks might not be applicable for BSNs. Also the schemes proposed for *ad hoc* networks such as asymmetric cryptography techniques would be costly for BSNs. Due to the constraints on power and computational ability, it seems only the well-known symmetric-key cryptographic algorithm, which is called Message Authentication Code (MAC), will be suitable for BSNs authenticity. MAC is a symmetric-key primitive that inputs a key-message pair to produce a unique tag. The integrity and the authenticity of the message are protected by the tag and the key respectively.

To ensure the authenticity of WSNs communication, security protocols via different MACs have been proposed. One widely used method is the Security Protocol for Sensor Networks (SPINS)[5], which consists of μTESLA (micro version of the Timed, Efficient, Streaming, Loss-tolerant Authentication) and SNEP (Secure Network Encryption Protocol) for broadcasting messages. Following SPINS, many lightweight security architectures have been proposed for WSN, e.g., TinySec[6], SenSec[7], and MiniSec[8]. All these architectures have considered which MAC will be suitable in the WSN packet/message authentication. For instance, TinySec and MiniSec recommend the well-known CBC-MAC[9] and OCB-MAC[10] respectively, whilst SenSec uses a novel scheme called XCBC-MAC[7]. All these MACs recommended for WSNs[6-8] are based on the operation modes of block cipher, and suggest 32-bit length tag for authenticity. Nevertheless, hash functions can be used to construct MACs as well. However, it was discovered that MACs based on dedicated hash functions (e.g., HMAC based on SHA-1[11]) are less competitive than block-cipher-based ones for highly constrained devices[12]. It is widely recognized by the BSN research community that authentication in BSN protocols is usually for short messages in network processing[1]. Therefore a lightweight MAC, which takes both the one-wayness and the collision resistance into account, will be more suitable for the BSN security.

To balance the security requirements and the constrained resources, a proper security level must be chosen for BSN authenticity. Intuitively, 32-bit security level for WSN is not suitable even for the one-wayness of BSN communication. As a comparable case for sensitive data authenticity, the authentication of Electronic Funds Transfer in the US Federal Reserve System uses a 64-bit CBC-MAC, and additionally a secret value for IV is daily changed and synchronized by the member banks. In other applications, certain authorities even recommended to implement a MAC with a longer length of 128-bit. Although a proper security level for a certain BSN application will be settled case by case, a 64-bit security bound is widely accepted for resisting practical threats in such hardware-limited devices. Since the power and RAM are highly constrained on a BSN node, a BSN-oriented MAC must take resource

Table 1. Specifications of Typical BSN Nodes

Node	CPU	RAM (KB)	Flash Memory (KB)	Voltage (V)	OS
TI Node②	16 bit, 8 MHz	2	64	1.8 ∼ 3.6	TinyOS
MICAz Node③	8 bit, 16 MHz	4	128	2.7 ∼ 3.3	TinyOS
MyriaNed④	16 bit, 32 MHz	8	128	1.6 ∼ 3.6	MyriaCore

②Texas Instruments. http://focus.ti.com/lit/ds/symlink/msp430f149.pdf, August 2013.

③Crossbow. http://www.openautomation.net/uploadsproductos/micaz_datasheet.pdf, August 2013.

④ALwEN project. http://www.alwen.nl, August 2013.

limitations into its design rationale as well. Without the loss of generality, the above conditions may not only be imposed on BSN applications, but also hold on authenticity-aware WSN applications. We note that the BSN circumstances are emphasized for the clarity of the motivation.

Our Contributions. The contributions of this paper are threefold. Firstly, the authentication modes for BSN are analyzed. We describe some practical problems of the MACs recommended in popular security architectures for WSN, such as TinySec (CBC-MAC), MiniSec (OCB-MAC) and SenSec (XCBC-MAC). In particular, we demonstrate an existential forgery attack on XCBC-MAC, which implies that the authenticity of SenSec is broken. Secondly, a performance comparison is presented on efficient MAC candidates from different design principles, e.g., CBC-MAC, OCB-MAC, ALPHA-MAC[13]. Thirdly, taking into account the requirements for BSN authenticity, we propose a tunable lightweight MAC based on the PRESENT block cipher[14], which is named TuLP. By extending the generic construction of ALRED[13], the structure of TuLP is designed by considering the potential security flaws[15-18]. A 128-bit variant TuLP-128 is also proposed for the higher resistance against internal collisions. Compared with the existing schemes including CBC-MAC, OCB-MAC, etc., our lightweight MACs show a better performance on MICAz node with less memory costs, and are also energy-efficient in the level of gate equivalents. We note that a preliminary version of this paper[19] has been published in the Proceedings of Indocrypt 2009. The major extension of this paper is listed as follows.

1) The introduction section is totally revised by considering the advices from the conference. Figures are added for a better presentation.

2) Authentication modes in BSN and the security definitions for MACs are added for a better understanding of the authentication issues in BSN.

3) The security analysis of TuLP is extended in formal proofs.

4) The software performance analysis includes more candidates for convincing comparison.

Organization. The remainder of this paper is organized as follows. In Section 2, we recall the necessary definitions and notions. The problems with the recommended MACs in the proposed security architectures for WSN are described in Section 3. Section 4 gives a performance comparison of some efficient MAC candidates for BSN authenticity. The designs of TuLP and TuLP-128 are presented in Section 5 along with a detailed analysis of the security and performance. Section 6 concludes the paper.

2 Preliminaries

Here we review some definitions and primitives which will be used in the following sections. Let $\oplus$ denote the bit-wise exclusive-or (XOR) operation. A message $M = a||b$ denotes the concatenation of two strings a and b. $\mathbb{M}$ and $\mathbb{K}$ denote the message space and the key space respectively.

2.1 Cryptographic Primitives

ALRED. The ALRED construction is a generic MAC design which was introduced by Daemen and Rijmen[13]. The ALRED construction consists of the following steps:

1) Initialization: fill the state with an all-zero block and encrypt it with a full encryption E with an authentication key k.

2) Chaining: for each message, iteratively perform an *injection layout* to map the i-th message block x_i to the same dimensions as a sequence of r round keys of E. By using the output of the injection layout as the round keys, apply a sequence of r times round function of E to the state.

3) Finalization: apply a full encryption E with the authentication key k to the final state. The tag is the first ℓ_m bits of the output.

Fig.2 depicts the ALRED construction with $r = 1$[13]. Since many block ciphers are designed with extra rounds for conservative security margins, ALRED actually uses such margins as a trade-off for performance advantages. By using AES as the underlying block cipher, Daemen and Rijmen also presented two paradigms of ALRED which are called ALPHA-MAC[13] and Pelican[20] respectively. In the literature,

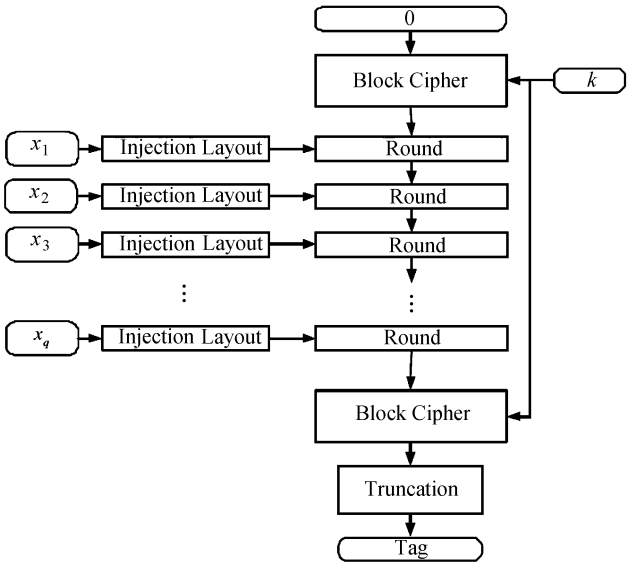

Fig.2. ALRED construction with $r = 1$.

many attacks show that ALPHA-MAC and Pelican might be threatened under the internal collisions[15], the side-channel attack[16] and the impossible differential analysis[17]. Nevertheless, Dunkelman *et al.*[18] showed that any ALRED-type MAC based on a keyless block cipher (e.g., ALPHA-MAC and Pelican use 1-round and 4-round keyless AES respectively) is vulnerable to time/memory trade-off attacks.

PRESENT. At CHES 2007, Bogdanov *et al.* proposed an ultra-lightweight block cipher which is named PRESENT[14]. PRESENT is an example of substitution-permutation network (SPN) and consists of 31 rounds. The block length is 64 bits and two key lengths of 80 and 128 bits are supported. The hardware requirements for PRESENT are competitive. Using the Virtual Silicon (VST) standard cell library based on UMC L180 0.18μm 1P6M Logic Process (UMCL18G212T3), the encryption-only PRESENT-80 and PRESENT-128 occupy 1 570 and 1 886 gate equivalents respectively[14]. Since Bogdanov *et al.* did not expect the 128-bit key version to be used until a rigorous analysis is given, the term PRESENT means 80-bit key version in hereafter. A high-level algorithm of the round function of PRESENT is depicted in Fig.3[14]. First, the 64-bit input of the round function is XORed with the subkey K^i. The total 32 subkeys (K^{32} for whitening after the final round) are derived from the key schedule algorithm over an 80-bit secret key. Next, 16 identical 4×4-bit S-boxes S are used in parallel as the non-linear substitution layer. Finally, a hardware-efficient bit-oriented permutation is executed to provide diffusion.

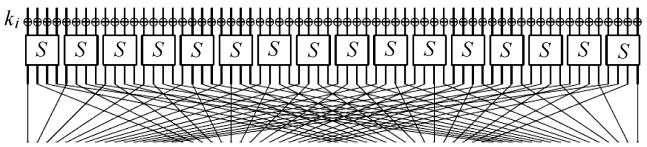

Fig.3. Round function of PRESENT.

PRESENT also has a hardware-efficient key schedule to avoid the weakness of related-key attacks. The user-supplied key is stored in a key register K and represented as $k_{79}k_{78}\cdots k_0$. At the i-th round, the leftmost 64-bit of the current key register becomes the subkey $K^i = k_{79}k_{78}\cdots k_{16}$. Subsequently, the key register K is updated as follows:

• cycling left shift 61 bits such that $[k_{79}k_{78}\cdots k_0] = [k_{18}k_{17}\cdots k_{20}k_{19}]$;

• the leftmost 4 bits are passed through PRESENT S-box such that $[k_{79}k_{78}k_{77}k_{76}] = S[k_{79}k_{78}k_{77}k_{76}]$;

• the round counter value is XORed with bits $k_{19}k_{18}k_{17}k_{16}k_{15}$.

Further details about the specification of PRESENT can be found in [14], including basic results of the differential and linear cryptanalyses, which can be summarized as follows.

Theorem 1. *Any* 5*-round differential characteristic of PRESENT has a minimum of* 10 *active S-boxes.*

Theorem 2. *Let* ϵ_{4R} *be the maximal bias of a linear approximation of four rounds of PRESENT. Then* $\epsilon_{4R} \leqslant 2^{-7}$.

Moreover, Bogdanov *et al.*[12] proposed some low-energy block-cipher-based hash functions (e.g., single and double block length constructions of DM-PRESENT and H-PRESENT respectively). At CHES 2011, Bogdanov *et al.*[21] proposed a new lightweight hash function based on PRESENT, which is called *SPONGENT*. The comparison on the hardware performances[12,21] shows that those PRESENT-based hash functions are more practical than dedicated or AES-based hash functions on highly constrained devices, such as RFID tags.

Many cryptanalysis results have been given on the PRESENT block cipher. Wang[22] presented a differential attack on 16-round PRESENT with the complexities of about 2^{64} chosen plaintexts, 2^{32} 6-bit counters, and 2^{64} memory accesses. Albrecht and Cid[23] introduced an algebraic differential attack on 19-round PRESENT-128. Besides the above basic attacks, some complicated attacks have been proposed based on preconditions. Collard and Standaert[24] described a statistical saturation attack against 24-round PRESENT. Besides the required plaintexts exceeds 2^{32}, the statistical saturation attack[24] still depends on the assumption that there exists an attack exploiting distributions of larger dimensions by combining multiple plaintexts. But it is still an open problem to calculate the effect of this assumption to the attack complexities. Özen *et al.*[25] proposed a related-key rectangle attack on 17-round PRESENT-128. However the known attacks on PRESENT with 80-bit keys, without any precondition, so far are bounded with 16 rounds[22].

2.2 Security Definitions for MACs

Before we formally present the security definitions for MACs, we first describe what a MAC is and how it is used. Assume there are users who require the communication data to be authenticated, they will securely generate and share an authentication key k with a key generation algorithm $\mathsf{GEN}(.)$. While a participant wants to send a message m to the other, he/she will compute a MAC tag (simply denotes by *tag*) based on the message m and the key k with a tag generation function $\mathsf{MAC}(.)$. By receiving *tag* and m, one can verify if the tag is valid for the message and the authentication key with a verification algorithm $\mathsf{VER}(.)$. The above descriptions can be formalized as the following definition[26].

Definition 1. *A message authentication code* $\mathcal{M} = \{\mathsf{GEN}, \mathsf{MAC}, \mathsf{VER}\}$ *is a tuple of probabilistic polynomial-time algorithms* $(\mathsf{GEN}, \mathsf{MAC}, \mathsf{VER})$ *such that*:

1) *The key-generation algorithm* GEN *takes the security parameter* 1^n *and outputs a secret key* k *with* $|k| \geqslant n$ *as the authentication key, such that* $k \leftarrow \mathsf{GEN}(1^n)$;

2) *The tag-generation algorithm* MAC *takes a key* k *and a message* m*, and outputs a tag* t*, such that* $t \leftarrow \mathsf{MAC}_k(m)$;

3) *The verification algorithm* VER *takes a key* k*, a message* m *and a tag* t*.* $\mathsf{VER}_k(m,t)$ *outputs* 1 *if* $t = \mathsf{MAC}_k(m)$*, otherwise it outputs* 0.

Deriving from the security model in [26], we define the following experiment for a MAC $\mathcal{M}$ under an adaptive adversary $\mathcal{A}$.

Definition 2. *The message authentication experiment* $\mathsf{MAC\text{-}Sec}_{\mathcal{A},\mathcal{M}}(n)$ *on a MAC* $\mathcal{M}$ *with the security parameter* n *under an adaptive adversary* $\mathcal{A}$ *is defined as follows*:

1) *An authentication key* k *is randomly generated by running* $\mathsf{GEN}(1^n)$.

2) *The adversary* $\mathcal{A}$ *is given input* 1^n.

3) $\mathcal{A}$ *has oracle accesses to the algorithms* MAC_k *and* VER_k *and their subfunctions, such as the compression function in* MAC.

Let $\mathcal{Q}$ be the set of all queries that $\mathcal{A}$ asked to the oracle. ε denotes a negligible probability. Based on the above experiment, the security properties of MACs can be defined as follows.

Definition 3. *A message authentication code* $\mathcal{M} = \{\mathsf{GEN}, \mathsf{MAC}, \mathsf{VER}\}$ *is existentially unforgeable under an adaptive chosen-message adversary* $\mathcal{A}$*, if and only if the probability that* $\mathcal{A}$ *can output a pair of message and tag* (m,t) $(m \notin \mathcal{Q})$ *is non-negligible after the experiment, such that*

$$\Pr[(m,t) \leftarrow \mathsf{MAC\text{-}Sec}_{\mathcal{A},\mathcal{M}}(n); \mathsf{VER}_k(m',t') = 1] \leqslant \varepsilon.$$

Definition 4. *A message authentication code* $\mathcal{M} = \{\mathsf{GEN}, \mathsf{MAC}, \mathsf{VER}\}$ *cannot be key-recovery attacked under an adaptive chosen-message adversary* $\mathcal{A}$*, if and only if the probability that* $\mathcal{A}$ *can output the authentication key* k *is non-negligible after the experiment, such that*

$$\Pr[k \leftarrow \mathsf{MAC\text{-}Sec}_{\mathcal{A},\mathcal{M}}(n); \\ t \leftarrow \mathsf{MAC}_k(m), \mathsf{VER}_k(m,t) = 1] \leqslant \varepsilon.$$

Note that a MAC is secure if no feasible adversary can successfully execute the above attacks with non-negligible probability in the above experiment.

2.3 Authentication Modes in BSN

A typical BSN involves three kinds of communication: off-body communication, on-body communication, and in-body communication. For protecting both authenticity and confidentiality, the data payload of each packet in a BSN should be encrypted, and then authenticated with the header (includes nonce, source, destination, and group ID, etc.) by the sender before it is sent to the receiver. Fig.4 shows two different paradigms to build up a secure packet with the properties of confidentiality, integrity and authenticity. The left paradigm, which is labeled as "short message efficient paradigm", is borrowed from Rogaway's Authenticated Encryption with Associate Data[27]. If the me-

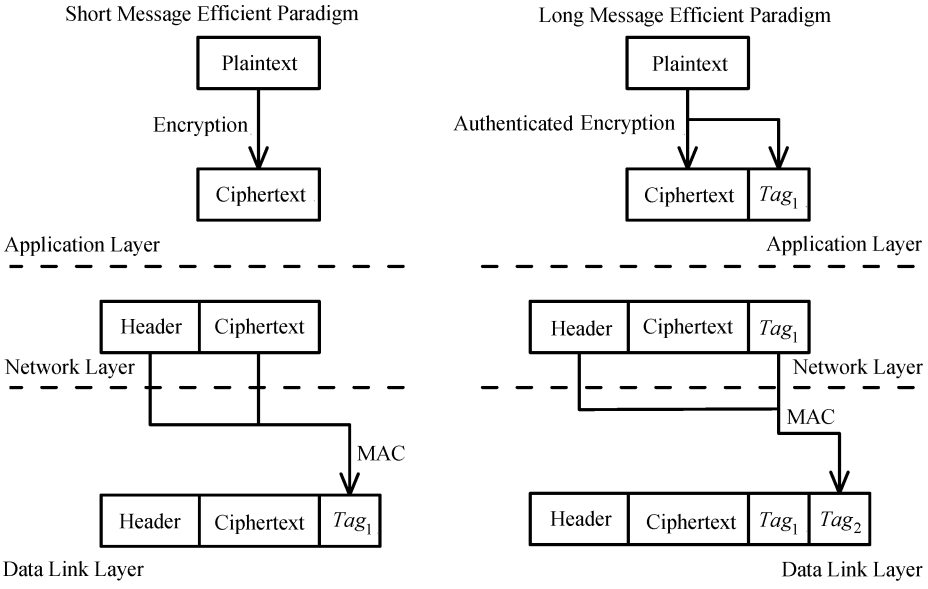

Fig.4. Two authentication paradigms for BSN.

ssage is long, the verification including the header and full ciphertext will be costly.

The right paradigm, which is labeled as "long message efficient paradigm", trades off the overhead on an extra tag to avoid the verification costs on the full ciphertext. Let k be an authentication key shared by a sender and a receiver. When a packet has arrived, the receiver first checks if $MAC(k, \text{Header}, Tag_1) = Tag_2$. If it does not hold, the receiver will just drop the packet without decryption. Otherwise the receiver checks the validity of the ciphertext through the equation $MAC(k, \text{Ciphertext}) = Tag_1$, and then decrypts it to obtain the original plaintext. If the underlying authentication encryption and MAC function are secure, Tag_2 protects the authenticity and integrity of the header and Tag_1, whilst Tag_1 protects the original plaintext.

A widely-known result[28] states that communicating a single bit of data consumes several orders of magnitude more power than executing a basic 32-bit arithmetic instruction. If a message is short, the communication cost on an extra tag might be larger than the verification which includes the full ciphertext. Otherwise, the verification might be higher than using an extra tag. One can choose the short or long message efficient paradigm by consideration of different applications. Generally, we consider the short message efficient paradigm is more suitable for BSN authentication. No matter which authentication paradigm is opted, the security and performance of the underlying MAC function will play a pivotal role for BSN authenticity.

3 Problems with MACs Recommended for WSN

For ensuring the security of the communication in WSN, many schemes have been proposed for the different layers of WSN. Basically, the security of data-link layer is fundamental for other security properties in the higher layers, e.g., secure routing in network layer and non-repudiation in application layer. In practice, there exist three widely-cited schemes for the security of data-link layer, which are TinySec[6], SenSec[7], and MiniSec[8]. For confidentiality, all the three schemes suggest using a lightweight block cipher for data encryption. But for authenticity, three totally different MAC functions are recommended, which are claimed to be suitable for WSN. In this section, we will give a comparative analysis of the three recommended MAC functions in the three schemes[6-8].

CBC-MAC. In TinySec[6], Karlof *et al.* suggested to use CBC-MAC[9] as the underlying MAC function. CBC-MAC uses a cipher block chaining construction for computing and verifying MACs. The first advantage of CBC-MAC is simplicity, as it relies on a block cipher which minimizes the number of cryptographic primitives that must be implemented on BSN nodes with a limited memory or gate equivalents. For BSN applications, the disadvantage of CBC-MAC is that independent keys should be used for encryption and authentication. Furthermore, it has been proven that the one-key CBC-MAC construction[29] is existentially forgeable when arbitrary length messages are allowed. To preserve the provable security for arbitrary length messages, a variant of CBC-MAC uses three different keys for the authentication[30]. Although the three-key construction solves the arbitrary length message problem and avoids unnecessary message padding, it raises another typical risk with respect to the key management in BSN. Compared with the one-key construction, the extra keys will impose a burden on key generation, distribution and storage. The risk of the key management indicates that a provably secure CBC-MAC might be less practical for BSN applications. As a direct alternative for CBC-MAC, we recommend the CMAC⑤ algorithm, which is submitted to NIST as a variation of CBC-MAC that Black and Rogaway proposed and analyzed[30]. Note that CMAC only using a single key with pre-computation would remove most of burdens on key generation and distribution.

XCBC-MAC. The XCBC-MAC algorithm, which was proposed by Li *et al.*, is part of the authenticated encryption mode for SenSec[7]. Let k_A and k_E be the authentication key and the encryption key, respectively. Let message $M = m_1||m_2||\cdots||m_t$. In general, the XCBC-MAC algorithm can be viewed as a variant of the two-key CBC mode. Fig.5 depicts the construction of XCBC-MAC.

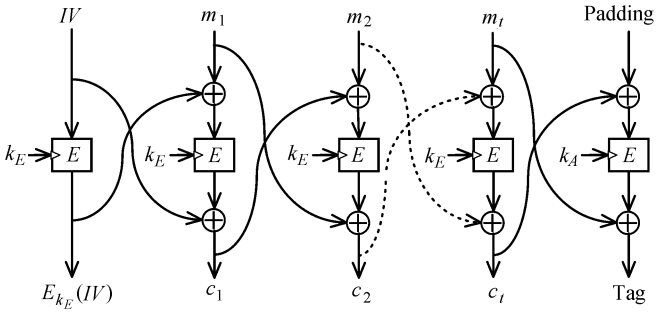

Fig.5. XCBC algorithm proposed in SenSec.

Unfortunately, we have found an existential forgery on XCBC-MAC by implementing adaptive chosen-message attack. One can easily build two different messages with the same tag under the XCBC mode. The forgery can be described in the following steps:

⑤NIST. http://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf, August 2013.

1) First, adversary $\mathcal{A}$ obtains initial value IV and $E_{k_E}(IV)$ from the first block of any former ciphertext under k_E.

2) Next, $\mathcal{A}$ requests the encryptions on the two different blocks $E_{k_E}(IV) \oplus m_1$ and $E_{k_E}(IV) \oplus m'_1$ in the XCBC mode. The ciphers will be $E_{k_E}(m_1) \oplus IV$ and $E_{k_E}(m'_1) \oplus IV$. $\mathcal{A}$ obtains $E_{k_E}(m_1)$ and $E_{k_E}(m'_1)$ by XORing the ciphers with IV.

3) Finally, $\mathcal{A}$ arbitrarily selects a message M', and then outputs two different messages M_1, M_2, where $M_1 = E_{k_E}(IV) \oplus m_1||E_{k_E}(m_1)||0||M'$ and $M_2 = E_{k_E}(IV) \oplus m'_1||E_{k_E}(m'_1)||0||M'$.

An illustration of our attack is depicted in Fig.6. It is straightforward that two different prefixes $E_{k_E}(IV) \oplus m_1||E_{k_E}(m_1)||0$ and $E_{k_E}(IV) \oplus m'_1||E_{k_E}(m'_1)||0$ will produce the same zero output to the next step. Thus the two different messages M_1 and M_2 will have the same tag. The attack is feasible since IV is a public-known value and the prefixes are computationally indistinguishable from a random query. Moreover, since XCBC-MAC has been proposed as an authenticated-encryption scheme, the encrypted IV can be obtained from the first block of the corresponding ciphertexts.

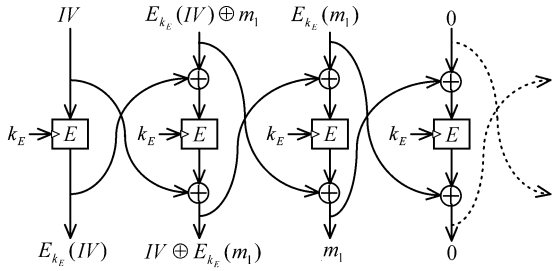

Fig.6. Existential forgery under XCBC-MAC.

Although our existential forgery on XCBC-MAC can be avoided by using a one-time randomized IV for each authentication, this protection might be impractical for sensor networks. If IV must be updated after one-time usage, at least all neighbor nodes need to be synchronized. Otherwise receivers cannot authenticate any packet from a sender. There are two methods for updating IV in a network. First is to add a fresh IV in every packet, which imposes an overhead on communications. The other is to synchronize IV with a predefined program in each node. Both solutions are costly in sensor networks. Therefore, it is less practical for an IV to be distributed just for one-time usages. Although other operation modes of block cipher also require a fresh IV for resisting statistical weakness (especially in encryption), the existential forgery of XCBC-MAC is a higher level security threat for protecting authenticity. For instance, if an IV is repeatedly used in CBC-MAC then only the same messages will produce the same MAC values. Even if IV is not changed, attackers still cannot existential forge a valid CBC-MAC value on a different IV or message. Due to the above reasons, the XCBC-MAC algorithm proposed in SenSec[7] is insecure under the chosen message attack and should be abandoned in any circumstance of sensor network authentication.

OCB-MAC. In MiniSec[8], Luk *et al.* suggested using the OCB mode[10], which is an efficient authenticated encryption scheme, as the MAC function for message authenticity and integrity. OCB uses a unique nonce to protect the authenticity, which will also increase an overhead of communication. Moreover, Ferguson[31] described a collision attack on OCB with variable-length messages. To keep adequate authentication security of OCB, one has to limit the amount of data that the MAC algorithm processes. Although variable-length message attacks seem rather harmless in practice, many MACs (e.g., CMAC) are provably secure without obeying this limitation.

4 Comparison of Some Practical MACs for BSN

We have shown that the MAC functions proposed for WSN in the literature are not exactly suitable for BSN. Many different MACs have been proposed in the past decades. Driven by the highly constrained resources of BSN node, the performance and security of those candidates should be rigorously examined before they are implemented. Basically, there are three approaches towards designing a MAC function. The first is to design a new primitive from scratch, such as UMAC[32]. The second is to define a new mode of operation for existing primitives, such as variants of encryption modes of block ciphers: CBC-MAC[9] and OCB-MAC[10], or mode variants of hash functions: HMAC/NMAC[11,33]. The third approach, which can be viewed as a hybrid of the above two approaches, is to design new MAC functions using components of existing primitives, such as ALPHA-MAC[13].

Based on the security and performance requirements of BSN, we will give a detailed comparison of some popular MAC candidates, which are claimed to be efficient by following the three different approaches. To be fair, all MACs use AES-128 as the underlying block cipher and allow arbitrary-length inputs. The timing of the keysetup and the message processing are estimated from the performance data given by the NESSIE consortium[34] (Pentium III/Linux Platform), such that the message processing time is measured in cycles/byte, while the keysetup and keysetup in addition to finalization are measured in cycles. The area in gate equivalents (GE) can be calculated from two parts: the area of the underlying component or primitive, and the area

for internal operations and storages. In order to compare the area requirements independently it is common to state the area in GE, where one GE is equal to the area which is required by two-input NAND gate with the lowest driving strength of the appropriate technology[35]. By following the same method[12,36], we also use the Virtual Silicon (VST) standard cell library based on UMC L180 0.18μm 1P6M Logic Process (UMCL18G212T3) to estimate each area in GE of the candidates. According to the related experiments[36], the area for AES-128 encryption is estimated to be 3 400 GE, as well as 64-bit storing and exclusive-or require 512 GE and 170 GE, respectively.

For chips built with CMOS technology, the power consumption is the sum of two parts: the static and the dynamic costs. The static part is roughly proportional to the area, namely, the more the size of the chip, the larger energy costs, whilst the dynamic part is proportional to the operating frequency. For the devices with a lower operating frequency, the static power consumption is the most significant. Based on this reason, the area of gate equivalents is often used as a simplified benchmark for energy efficiency. The comparison in Table 2 shows that ALPHA-MAC advances in both the message processing speed and the area of GE, which indicates that one could also build a time and energy efficient MAC from the ALRED construction by using a lightweight block cipher.

5 Two New Lightweight MACs from ALRED

In this section, we will propose a tunable lightweight MAC based on PRESENT, which is named TuLP. To raise the security bound of resisting internal collisions, we will also give a wide-pipe version of TuLP, which is called TuLP-128. Both of our schemes use the experiences of ALPHA-MAC[13] and Pelican[20]. Next, the security of our schemes will be analyzed. Finally, the performance of our lightweight schemes will be given. Compared with the results in Table 2, our new MAC functions are not only time-efficient with less memory usage, but also energy-efficient in the number of gate equivalents.

5.1 TuLP and TuLP-128

By using the round function of PRESENT[14], first a new MAC function TuLP is built from a modification of the ALRED construction. TuLP is a lightweight MAC function with an 80-bit key length at maximum and 64-bit block length, which consists of the following steps:

1) *Padding.* Let k be an authentication key such that $|k| \leqslant 80$ bits. If $|k|$ is less than 80 bits, it should be iteratively padded with 1 and 0 as $10101\cdots$. First pad M with $\lambda(M,k)$ where $\lambda(M,k)$ returns the concatenation of bitwise lengths of M and k. Then pad the concatenated string to a multiple of 64 bits, e.g., appending a single bit 1 followed by necessary d bits 0. Finally split the result $pad(M)$ into 64-bit blocks $m_1, m_2, \ldots, m_t$, $t = |pad(M)|/64$, such that $pad(M) = M||\lambda(M,k)||10^d$.

2) *Initialization.* Apply one full-round PRESENT encryption E to the initial value IV with the (padded) authentication key k, then obtain $s_0 = E_k(IV)$ as the initial state.

3) *Compression.* For each message block m_i where $i \in \{1, 2, \ldots, t\}$, XOR m_i with the current state s_i as the 64 most significant bits of the key k_i for current r times PRESENT round function ρ. The rest 16 bits of the key k_i is derived from the 16 most significant bits of the authentication key k (denote by $MSB^{16}(k)$). By executing the same key schedule algorithm of PRESENT, apply r times ρ on the state s_{i-1}, such that

$$k_i = m_i \oplus s_{i-1}||MSB^{16}(k), \quad s_i = \rho^r_{k_i}(s_{i-1}).$$

4) *Finalization.* Apply one full-round PRESENT encryption to the state s_t under the key k, and then truncate the least significant ℓ_m bits of the final state s_{t+1} as the tag of the message M. Therefore, $s_{t+1} = E_k(s_t)$, $tag_M = Trunc^{\ell_m}(s_{t+1})$.

Since the length of internal state is only 64 bits, TuLP is not strong enough to resist the birthday attack on internal states to find a collision. Although this "weakness" may not be fatal in some BSN applications, we still provide a wide-pipe version, which is called TuLP-128, to increase the state and the maximum tag lengths to be 128 bits. The key length of TuLP-128 is up to 160 bits. We note that the design principle is inspired by MDC-2[37] and the padding rule is identical to TuLP.

Table 2. Comparison of Some Practical MAC Functions

Functions	Design Method	Keysetup (Cycles/Byte)	Finalization (Cycles/Byte)	Message processing (Cycles/Byte)	Area in GE (Estimate)
CBC-MAC[9]	Cipher mode	616	1 440	26.0	4 764
OCB-MAC[10]	Cipher mode	644	1 444	30.0	6 812
ALPHA-MAC[13]	AES components	1 032	416	10.6	4 424
HMAC (SHA-1)[11]	Hash mode	1 346	3 351	15.0	8 120

1) *Padding.* Let k be an authentication key such that $|k| \leqslant 160$ bits. By using the same padding rule of TuLP, split the result $pad(M) = M||\lambda(M,k)||10^d$ into 64-bit blocks $m_1, m_2, \ldots, m_t$, $t = |pad(M)|/64$.

2) *Initialization.* Divide the (padded) authentication key k into two 80-bit keys $k_l||k_r$. Then apply one full-round PRESENT encryption to two different 64-bit initial values IV_1 and IV_2 under k_l and k_r, respectively. Obtain the outputs as the left and right initial states $s_{l,0}$ and $s_{r,0}$, such that $s_{l,0} = E_{k_l}(IV_1), s_{r,0} = E_{k_r}(IV_2)$.

3) *Compression.* For each message block m_i where $i \in \{1, 2, \ldots, t\}$, first split the last left and right states $s_{l,i-1}$ and $s_{r,i-1}$ into four 32-bit blocks. Then exchange the least significant 32 bits of the left state (denoted by $LSB^{32}(.)$) with the most significant 32 bits of the right state. The exchanged input states are denoted by $\hat{s}_{l,i-1}$ and $\hat{s}_{r,i-1}$. By following the same algorithm of the compression in TuLP, apply r PRESENT round functions on the exchanged input states $\hat{s}_{l,i-1}$ and $\hat{s}_{r,i-1}$ respectively.

$$\begin{aligned}
&\hat{s}_{l,i-1} = MSB^{32}(s_{l,i-1})||MSB^{32}(s_{r,i-1}),\\
&\hat{s}_{r,i-1} = LSB^{32}(s_{l,i-1})||LSB^{32}(s_{r,i-1});\\
&k_{l,i} = m_i \oplus s_{l,i-1}||MSB^{16}(k_l),\\
&k_{r,i} = m_i \oplus s_{r,i-1}||MSB^{16}(k_r);\\
&s_{l,i} = \rho^r_{k_{l,i}}(\hat{s}_{l,i-1}),\\
&s_{r,i} = \rho^r_{k_{r,i}}(\hat{s}_{r,i-1}).
\end{aligned}$$

4) *Finalization.* Apply one full-round PRESENT encryption to the left and the right states under the divided keys k_l and k_r respectively. Then truncate the least significant ℓ_m bits of the concatenation of the final states as the tag of the message M.

$$\begin{aligned}
&\hat{s}_{l,t} = MSB^{32}(s_{l,t})||MSB^{32}(s_{r,t}),\\
&\hat{s}_{r,t} = LSB^{32}(s_{l,t})||LSB^{32}(s_{r,t});\\
&s_{l,t+1} = E_{k_l}(\hat{s}_{l,t}),\ s_{r,t+1} = E_{k_r}(\hat{s}_{r,t});\\
&tag_M = Trunc^{\ell_m}(s_{l,t+1}||s_{r,t+1}).
\end{aligned}$$

Figs. 7 and 8 depict the high-level algorithms of TuLP and TuLP-128, respectively. Referring to the security issues of ALPHA-MAC and Pelican[12,16-17], the advantages of our schemes are as follows.

• In ALPHA-MAC[13], all message blocks directly become the round keys after the message injections, so the attacker can execute side-channel attacks in the known message scenario. Biryukov *et al.*[16] present a side-channel attack on ALPHA-MAC, which relies on the fact that the round keys of ALPHA-MAC are public-known by the attacker. Moreover, Dunkelman *et al.*[18] showed that ALRED is vulnerable if a keyless block cipher is used. In TuLP, round keys are not computed from a deterministic function of input message blocks but still use the original key scheduling algorithm. Unless the underlying reduced-round block cipher is broken, an attack is unlikely to make a hypothesis on any intermediate states of the algorithm. The XOR operation between the state and the input message block can resist the attacker to implement similar side-channel attacks[16] on TuLP and TuLP-128.

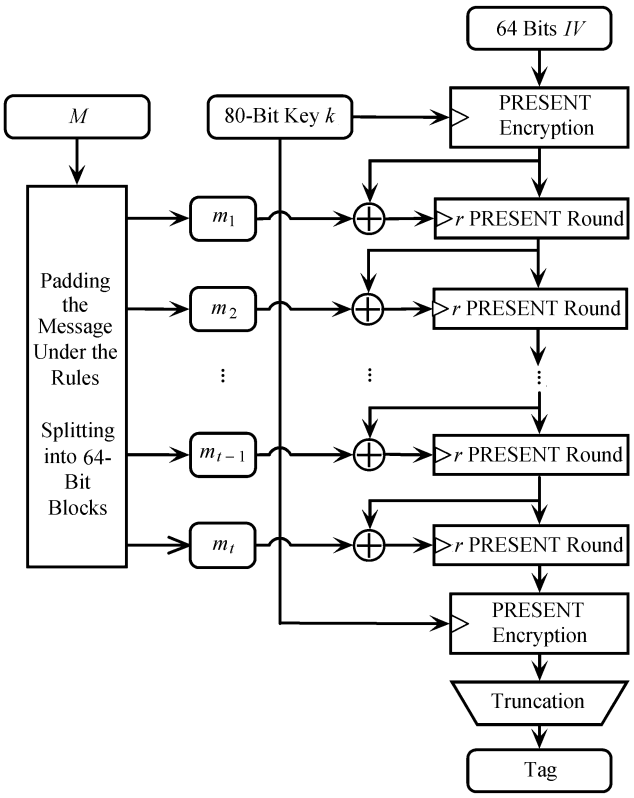

Fig.7. Illustration of TuLP.

• Like in Pelican[20], the message injection layer is also removed in TuLP and TuLP-128 for simplicity. Because it can hardly improve the resistance against linear and differential attacks. In Pelican, the message block is XORed with the last output state as the input state for current round. But in our schemes, the message block is XORed with the state as a part of the subkey for next round. We note that the iteration of $E_{k\oplus m}(k)$ is proven to be collision and preimage resistant in the black-box analysis of the PGV constructions[38].

• The bitwise lengths of the message and key are appended to the end of the message. Our message padding rule can avoid some trivial attacks, such as fixed-point, internal collision and extension attacks. ALPHA-MAC and Pelican only pad messages with a single 1 followed by the minimum number of 0 bits to suffice a block.

• Benefiting from the ALRED construction, the security of our schemes can be reduced to the security of PRESENT if internal collisions are not involved. The

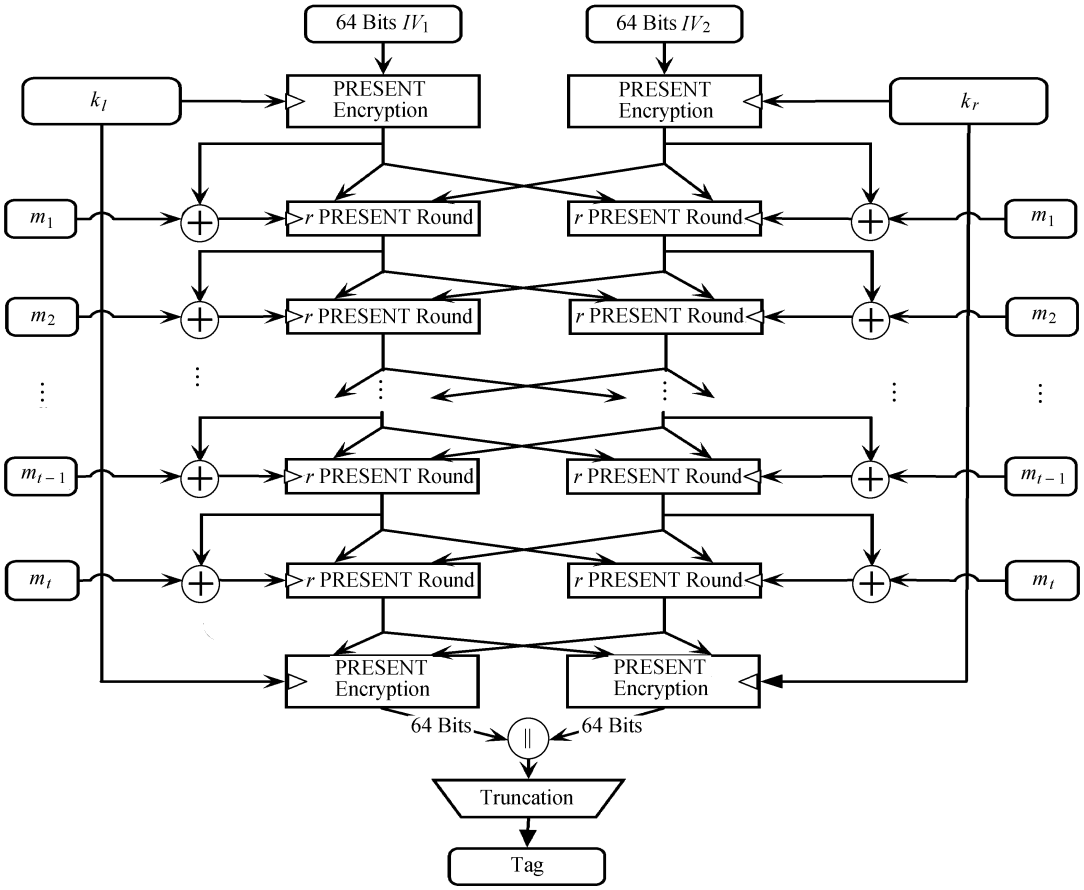

Fig.8. Illustration of TuLP-128.

proofs are provided in the security analysis of Subsection 5.2. Since the compressions in TuLP and TuLP-128 are different from the full-round PRESENT, authentication and encryption can use the same secret key.

• TuLP is designed for rapid message processing. The computational cost of the message processing is equivalent to $\frac{r}{31}$ of one PRESENT encryption. Whilst TuLP-128 provides a wider intermediate state and maximum 128-bit tag length for collision resistance, such that the cost of message processing only requires $2r/31$ of one PRESENT encryption.

• The choice of r rounds PRESENT in the compression is tunable by considering the practical balance of security and performance. Since key management in sensor network is expensive in computation and energy, the length of authentication key is tunable since the padding rules consider dynamic key length. To give practical instances for the analysis in the following subsection, we will consider r=16 in the compression of TuLP and TuLP-128, whilst $IV = IV_1 =$ `0123456789ABCDEF` and $IV_2 =$ `FEDCBA9876543210`. The test vectors of TuLP and TuLP-128 are provided in Appendix.

• Same as ALRED, one can replace PRESENT in the constructions of TuLP and TuLP-128 by any well-analyzed block cipher with a reasonable security margin, e.g., AES, Serpent, and Twofish. The extra rounds of the margin impose an upper bound to the trade-off between performance and security. Note that if the underlying block cipher is lightweight, the instantiation will also inherit its resource-efficient property.

5.2 Security Analysis

Based on the provability results of the ALRED construction in [13], it is straightforward to derive similar results on TuLP and TuLP-128. In this subsection, we first prove that TuLP is as strong as the PRESENT block cipher with respect to key recovery and existential forgery attacks without internal collisions. Then we analyze TuLP when internal collisions are considered. Finally, a similar security analysis is given on TuLP-128.

Theorem 3. *Any key recovery attack on TuLP requiring t (adaptively) chosen messages, can be converted to a key recovery attack on the PRESENT block cipher requiring $t+1$ adaptively chosen plaintexts.*

Proof. Let $\mathcal{A}$ be a successful attacker requiring t tag values corresponding to t (adaptively) chosen messages m_i yielding the key k, where $i \in \{1, 2, \ldots, t\}$. Then we derive a key recovery attack on the PRESENT block cipher as follows.

1) Request the first state $s_0 = E_k(IV)$.

2) For $i = 1$ to t, compute the intermediate state $s_i = \chi(s_0, m_i)$, where χ denotes the compression function of TuLP.

3) For $i = 1$ to t, request $tag_i = Trunc(E_k(s_i))$.

4) Submit t tag values to $\mathcal{A}$ to recover the key k.

The above attack requires t chosen messages and one chosen message on $E_k(IV)$. So the theorem follows. □

Similar to Theorem 3, the provability of TuLP can be extended to the existential forgery attack and the fixed point attack as follows.

Lemma 1. *Any existential forgery attack on TuLP without internal collisions requiring t (adaptively) chosen messages, can be converted to a ciphertext guessing attack on the PRESENT block cipher requiring $t + 1$ adaptively chosen plaintexts.*

Proof. Let $\mathcal{A}$ be a successful attacker requiring t tag values tag_i corresponding to t (adaptively) chosen messages m_i yielding another tag tag' under message m', where $i \in \{1, 2, \ldots, t\}$. Then we derive a ciphertext guessing attack on the PRESENT block cipher as follows.

1) Request the first state $s_0 = E_k(IV)$.

2) For $i = 1$ to t, compute $s_i = \chi(s_0, m_i)$, where χ denotes the compression function of TuLP.

3) For $i = 1$ to t, request $tag_i = Trunc(E_k(s_i))$.

4) Submit t tag values to $\mathcal{A}$ to obtain an existential forgery tag', which is a truncation of the valid ciphertext on the last internal state s_i.

The above attack requires t chosen messages and one chosen message on $E_k(IV)$. So the lemma follows. □

Lemma 2. *Any existential forgery attack on TuLP, requiring t (adaptively) chosen messages for a fixed point $\{(m, s)|E_{m\oplus s}(s) = s,\ m \in \mathcal{M},\ s \in \mathbb{K}\}$, can be converted to a fixed point attack $\{(m', k)|E_{m'}(k) = k,\ m \in \mathcal{M},\ k \in \mathbb{K}\}$ on the PRESENT block cipher requiring $t + 1$ adaptively chosen plaintexts.*

Proof. Let $\mathcal{A}$ be a successful attacker requiring t tag values corresponding to t (adaptively) chosen messages m_i yielding a fixed point fp, where $i \in \{1, 2, \ldots, t\}$. Then we derive a fixed point attack on the PRESENT block cipher as follows.

1) Request the first state $s_0 = E_k(IV)$.

2) For $i = 1$ to t, compute $s_i = \chi(s_0, m_i)$, where χ denotes the compression function of TuLP.

3) For $i = 1$ to t, request $tag_i = Trunc(E_k(s_i))$.

4) Submit t tag values to $\mathcal{A}$ to obtain a fixed point $fp = s_i$ such that $E_k(s_i) = s_i$.

The above attack requires t chosen messages and one chosen message on $E_k(IV)$. So the lemma follows. □

Now we analyze the security with respect to internal collisions.

Lemma 3. *Any existential forgery attack on TuLP with an internal collision requiring t (adaptively) chosen messages, can be converted to a collision attack on the r PRESENT round functions requiring $t + 1$ adaptively chosen plaintexts.*

Proof. Let $\mathcal{A}$ be a successful attacker requiring t tag values tag_i corresponding to t (adaptively) chosen messages m_i yielding another tag tag' under message m' with an internal collision, where $i \in \{1, 2, \ldots, t\}$. Then we derive a collision attack on the r PRESENT round functions as follows.

1) Request the first state $s_0 = E_k(IV)$.

2) For $i = 1$ to t, compute $s_i = \chi(s_0, m_i)$, where χ denotes the compression function of TuLP (i.e., the r PRESENT round functions).

3) For $i = 1$ to t, request $tag_i = Trunc(E_k(s_i))$.

4) Submit t tag values to $\mathcal{A}$ to obtain an existential forgery tag', and tag' should also be a valid ciphertext on the message m' with an internal collision $\chi(s_0, m_a) = \chi(s_0, m_b)$, where $a, b \in \{1, 2, \ldots, t\}$.

The above attack requires t chosen messages and one chosen message on $E_k(IV)$. So the lemma follows. □

The reasons why we choose r=16 in the compression of TuLP (and TuLP-128) to resist the internal collisions from the linear and differential cryptanalysis are briefly described as follows.

Theorem 4. *Consider $r = 16$ in the compression of TuLP. The minimum extinguishing differential in TuLP imposes a differential characteristic of about 2^{-64}. Whilst the maximum bias of the linear analysis has the probability of about 2^{-28} with 2^{56} known plaintext/ciphertext pairs.*

Proof. Based on the differential and the linear cryptanalyses that are given by Bogdanov *et al.*[14], any 5 rounds differential characteristic of PRESENT has a minimum of 10 active S-boxes. One round PRESENT has one S-box, all 31 rounds use the same. For differential cryptanalysis, we have:

1) For one active S-box, the maximum possibility for differential characteristic is 2^{-2}. Thus 16 rounds provide a lower bound $(2^{-2})^{r\times 10/5} = 2^{-64}$ for the probability of a characteristic. The probability is not greater than the birthday attack on the intermediate states (2^{-32} and 2^{-64} for TuLP and TuLP-128 respectively).

2) This differential cryptanalysis would require the memory complexity of about 2^{64} known plaintext/ciphertext pairs.

For linear cryptanalysis, we have:

1) Any 4 rounds provide the maximum bias of a

linear approximation $\epsilon_{4R} \leqslant 2^{-7}$. Hence 16 rounds provide the maximum bias of a linear approximation $(2^{-7})^{r/4} = 2^{-28}$.

2) The above linear cryptanalysis would require the memory complexity of about $1/(2^{-28})^2 = 2^{56}$ known plaintext/ciphertext pairs. □

Consider a typical BSN application consisting of 100 nodes, each node transfers an 8-byte message under the same authentication key per 15 seconds for monitoring. Although the above linear analysis has a non-negligible bias, the time and the memory complexities of obtaining 2^{56} plaintext/ciphertext pairs (about 2^{19} TB) would be impractical.

Subsequently, we consider the security of TuLP-128 without internal collisions.

Theorem 5. *Any key recovery attack on TuLP-128 requiring t (adaptively) chosen messages, can be converted to a key recovery attack on PRESENT requiring $t+2$ adaptively chosen plaintexts.*

Proof. Consider the situation that $k_l = k_r = k$. Let $\mathcal{A}$ be a successful attacker requiring t tag values corresponding to t (adaptively) chosen messages m_i yielding the key k, where $i \in \{1, 2, \ldots, t\}$. Let χ be the compression function of TuLP. $MSB^{32}(\cdot)$ and $LSB^{32}(\cdot)$ denote the truncation of the most and the least significant 32 bits, respectively. Then we derive a key recovery attack on the PRESENT block cipher as follows.

1) Request the initial left and right states $s_{l,0} = E_k(IV_1)$ and $s_{r,0} = E_k(IV_2)$.

2) For $i = 1$ to t, compute the left state

$$s_{l,i} = \chi(MSB^{32}(s_{l,i})||MSB^{32}(s_{r,i}), m_i)$$

and the right state

$$s_{r,i} = \chi(LSB^{32}(s_{l,i})||LSB^{32}(s_{r,i}), m_i).$$

3) For $i = 1$ to t, request $tag_i = Trunc(E_k(s_{l,i})||E_k(s_{r,i}))$.

4) Submit t tag values to $\mathcal{A}$ to obtain an existential forgery tag', which should also be a valid ciphertext on the message m'.

5) Submit t tag values to $\mathcal{A}$ to recover the key k.

The above attack needs t chosen messages except $E_k(IV_1)$ and $E_k(IV_2)$. So the theorem follows. □

Similar to Theorem 5, it is easy to obtain the following lemmas on TuLP-128.

Lemma 4. *Any existential forgery attack on TuLP-128 without internal collisions of requiring t (adaptively) chosen messages, can be converted to a ciphertext guessing attack on PRESENT requiring $t+2$ adaptively chosen plaintexts.*

Proof. Consider the situation that $k_l = k_r = k$. Let $\mathcal{A}$ be a successful attacker requiring t tag values tag_i corresponding to t (adaptively) chosen messages m_i yielding another tag tag' under message m', where $i \in \{1, 2, \ldots, t\}$. Then we derive a ciphertext guessing attack on the PRESENT block cipher as follows.

1) Request the initial left and right states $s_{l,0} = E_k(IV_1)$ and $s_{r,0} = E_k(IV_2)$.

2) For $i = 1$ to t, compute the left state $s_{l,i} = \chi(MSB^{32}(s_{l,i})||MSB^{32}(s_{r,i}), m_i)$ and the right state $s_{r,i} = \chi(LSB^{32}(s_{l,i})||LSB^{32}(s_{r,i}), m_i)$.

3) For $i = 1$ to t, request $tag_i = Trunc(E_k(s_{l,i})||E_k(s_{r,i}))$.

4) Submit t tag values to $\mathcal{A}$ to obtain an existential forgery tag', which should also be a valid ciphertext on the message m'.

The above attack needs t chosen messages except $E_k(IV_1)$ and $E_k(IV_2)$. So the lemma follows. □

Lemma 5. *Any existential forgery attack on TuLP-128 with a fixed point requiring t (adaptively) chosen messages, can be converted to a fixed point attack on PRESENT requiring $t+2$ adaptively chosen plaintexts.*

Proof. Consider the situation that $k_l = k_r = k$. Let $\mathcal{A}$ be a successful attacker requiring t tag values corresponding to t (adaptively) chosen messages m_i yielding a fixed point fp, where $i \in \{1, 2, \ldots, t\}$. We also choose the same left and right initial values such that $IV_1 = IV_2$. Then we can derive a fixed point attack on the PRESENT block cipher as follows.

1) Request the initial left and right states $s_{l,0} = E_k(IV_1)$ and $s_{r,0} = E_k(IV_2)$.

2) For $i = 1$ to t, compute the left state $s_{l,i} = \chi(MSB^{32}(s_{l,i})||MSB^{32}(s_{r,i}), m_i)$ and the right state $s_{r,i} = \chi(LSB^{32}(s_{l,i})||LSB^{32}(s_{r,i}), m_i)$.

3) For $i = 1$ to t, request $tag_i = Trunc(E_k(s_{l,i})||E_k(s_{r,i}))$.

4) Submit t tag values to $\mathcal{A}$ to obtain a fixed point fp such that the left state $s_{l,t+1} = \chi(MSB^{32}(s_{l,t+1})||MSB^{32}(s_{r,t+1}), m_{t+1})$, while the right state equals $s_{r,t+1} = \chi(LSB^{32}(s_{l,t+1})||LSB^{32}(s_{r,t+1}), m_{t+1})$.

Since the initial values are the same, and the intermediate values $s_{l,i}$ and $s_{r,i}$ are permutated by each round. A fixed point on TuLP-128 can easily be derived from the fixed point fp of TuLP-128 in the above attack. The attack requires t chosen messages except $E_k(IV_1)$ and $E_k(IV_2)$. So the lemma follows. □

By using multi-collisions, Knudsen *et al.*[39] provided a collision attack and preimage attacks on the MDC-2 construction with the time complexities of about $(\log_2(n)/n) \times 2^n$ and 2^n where the block length is n. The preimage attacks make new trade-offs so that the most efficient attack requires time and memory of about 2^n. Whilst the meet-in-the-middle attack on MDC-2[40] requires time and memory about $2^{3n/2}$ and 2^n. Based

on the security analysis of the MDC-2 construction and TuLP, the security of TuLP-128 with the internal collisions is as follows.

Theorem 6. *Consider $r = 16$ in the compression of TuLP-128. The internal collision and preimage attacks on TuLP-128 have the complexities of about $2^{61.3}$ and 2^{64}, respectively.*

Proof. The proof is based on the security that r=16 in the compression of TuLP-128. One S-box provides a maximum 2^{-2} possibility for differential characteristic, 16-round PRESENT provide a lower bound 2^{-64} for the probability of a characteristic. The minimum extinguishing differential in TuLP-128 imposes a differential characteristic of about 2^{-64} in the left state and the same in the right state. 16 rounds provide a maximum bias of a linear approximation 2^{-28}. But both the differential analysis and the linear cryptanalysis require a memory complexity no less than 2^{56} known plaintext/ciphertext pairs, which is impractical in BSN. Since PRESENT is an SP-network block cipher and the iteration of $E_{k\oplus m}(k)$ is proven to be collision and preimage resistant in the black-box analysis by Black *et al.*[38], and TuLP-128 has an MDC-2 like construction. Each round of the compression in TuLP-128 exchanges the right most 32 bits of the left state with the left-most 32 bits of the right state. Due to Knudsen *et al.*'s cryptanalysis of MDC-2[39], the internal collision attack and the preimage attack on TuLP-128 would require the time complexity of about $(\log_2(64)/64) \times 2^{64} \approx 2^{61.3}$ and 2^{64}, respectively. Therefore, the complexity of an internal collision is about $2^{-61.3}$ via the multi-collision attack with a negligible memory requirement. Whilst the preimage attack requires time and memory of about 2^{64}. □

Although TuLP-128 does not achieve the ideal upper bounds of collision and preimage resistances, the MDC-2 like structure can minimize the area in hardware, or the memory usage in software implementation. Nevertheless, a $2^{61.3}$ level of time complexity on finding an internal collision is still beyond the computational bound in practice.

6 Performance Evaluation

Before we study the performance of TuLP and TuLP-128, first we program an optimized implementation of PRESENT by using 1 KB look up table on MICAz nodes. From our performance tuning, we find that the bit permutation of PRESENT is costly in software implementation. Compared with the best known result of AES-128 software implementation on MICAz nodes[41], our optimized implementation of PRESENT still shows a competitive processing speed per block and promising lower memory costs. Since PRESENT has already been proven to be a better choice than AES in hardware implementation[12], our optimized implementation shows that PRESENT is still practical in software (shown in Table 3).

As a point of comparison, we select DM-PRESENT[12], which is derived from the Davies-Meyer construction and the PRESENT with an 80-bit key, as the underlying hash function for HMAC[11]. We also choose OCB-MAC and CBC-MAC (one-key) based on PRESENT as benchmarks. For comparability, AES-based ALPHA-MAC, OCB-MAC and CBC-MAC are also tested. The area in GE is estimated by using the Virtual Silicon (VST) standard cell library based on UMC L180 0.18 μm 1P6M Logic Process (UMCL18G212T3). All experiments are based the MICAz nodes (TinyOS version 2.10), which are popular in both WSN and BSN. The results in the entries of block processing speed (in milliseconds) are averaged by iterating 100 times experiments with/without the optimization in the keysetup.

If we choose $r = 16$ in the compression of TuLP, TuLP will be about 2 times faster than PRESENT encryption in message processing. Table 4 shows that the optimized TuLP approaches faster than PRESENT-based CBC-MAC (one-key), OCB-MAC and HMAC. The keysetup costs in our schemes, which require one (or two) PRESENT encryption(s) to generate an encrypted IV, mainly lack TuLP (or TuLP-128) in processing the short messages. We note that the keysetup can be optimized by precomputing the encrypted IV before the authentications with the same keys, and the values can be reused in the latter authentication with the same keys. Same optimization can be implemented in TuLP-128 to boost the processing of short messages. Although HMAC can also precompute the initialization values for optimization, the values must be treated and protected (128 bits for a certain key in DM-PRESENT) in the same manner as secret keys[11]. While the optimization for our schemes only increases a smaller storage (one encrypted IV is 64-bit) without need to be insulated. Although the lengths of internal state and tag are doubled, the performance of TuLP-128 is still

Table 3. Comparison of AES and PRESENT Implementations

Encryption	Software (MICAz)			Hardware		
	RAM (Byte)	ROM (Byte)	Processing Speed	Logic Process (μm)	Cycles per Block	Area (GE)
AES-128[41-42]	1 915	12 720	1.46 ms/16 Bytes	0.18	226	2 400
PRESENT-80[12]	1 040	1 926	1.82 ms/8 Bytes	0.18	32	1 570

Table 4. Comparison Amongst Some PRESENT-Based MAC Functions

	Key Length (Bit)	Block Size (Bit)	RAM (Byte)/ ROM (Byte)	Area in GE (Estimate)	Block Processing Speed (ms)
TuLP	80	64	1 048/3 302	2 252	4.46/6.63
TuLP-128	160	128	1 056/3 718	2 764	8.91/13.24
OCB-MAC (PRESENT)	80	64	1 048/3 362	2 252	6.56
CBC-MAC (PRESENT)	80	64	1 040/2 970	3 276	6.51
HMAC (DM-PRESENT)	80	64	1 056/3 484	2 213[12]	10.90
ALPHA-MAC (AES)	128	128	2 088/5 342	4 424	3.92
OCB-MAC (AES)	128	128	2 104/6 144	6 812	4.07
CBC-MAC (AES)	128	128	2 088/5 320	4 764	3.96

comparable to one-key CBC-MAC based on PRESENT. Obviously, TuLP-128 will be faster than HMAC with a double block length hash function based on PRESENT. Due to the hardware-oriented design of PRESENT, the block processing speeds of TuLP and TuLP-128 are slower than the MAC constructions based on AES. But the implementation costs of TuLP and TuLP-128 are much smaller than those of AES-based MACs, which will be more attractive in resource-constrained applications.

Nevertheless, if a higher security bound is required, one can tweak the rounds in the compressions of TuLP and TuLP-128. For instance, increase 16 rounds to 20 will decrease about $4/16 = 25\%$ performance in message processing. In return, a 20-round PRESENT will have a lower bound $(2^{-2})^{20\times 10/5} = 2^{-80}$ for a differential characteristic. And the maximal bias of a linear approximation is $(2^{-7})^{20/4} = 2^{-35}$, which requires 2^{70} known plaintext/ciphertext.

7 Conclusions

By considering the restrictions of BSN, we have proposed a new family of lightweight MACs that includes TuLP and TuLP-128. The key length and the number of round functions in the compression functions are tunable in our lightweight MACs, which support practical trade-offs between security and performance in BSN applications. The statistics strongly support that TuLP and TuLP-128 are promising on devices with constrained resources. The security of our schemes has been analyzed with respect to the cryptanalyses on ALRED and the results on PRESENT. Particularly, the construction of TuLP and TuLP-128 not only avoids the security threats which are discovered in ALRED variants, but also can instantiate with other lightweight block ciphers instead of PRESENT. Since both PRESENT and ALRED are new proposals, we suggest that rigorous analysis should be imposed to avoid any potential weakness inside the cryptosystems based on them.

References

[1] Yang G Z (eds.). Body Sensor Network. Springer London, 2006.

[2] Malan D, Fulford-Jones T, Welsh M, Moulton S. CodeBlue: An ad hoc sensor network infrastructure for emergency medical care. In *Proc. International Workshop on Wearable and Implantable Body Sensor Networks*, April 2004.

[3] Wood A, Virone G, Doan T, Cao Q, Selavo L, Wu Y, Fang L, He Z, Lin S, Stankovic J. ALARM-NET: Wireless sensor networks for assisted-living and residential monitoring. Technical Report, Department of Computer Science, University of Virginia, 2006.

[4] Kuryloski P, Giani A, Giannantonio R *et al.* DexterNet: An open platform for heterogeneous body sensor networks and its applications. In *Proc. the 6th International Workshop on Wearable and Implantable Body Sensor Networks*, June 2009, pp.92-97.

[5] Perrig A, Szewczyk R, Wen V, Culler D, Tygar J D. SPINS: Security protocols for sensor networks. In *Proc. the 7th Annual International Conference on Mobile Computing and Networking*, July 2001, pp.189-199.

[6] Karlof C, Sastry N, Wagner D. TinySec: A link layer security architecture for wireless sensor networks. In *Proc. the 2nd International Conference on Embedded Networked Sensor Systems*, November 2004, pp.162-175.

[7] Li T, Wu H, Wang X, Bao F. SenSec design. Technical Report, I^2R Sensor Network Flagship Project (SNFP: Security part), Technical Report-TR v1.0, February 2005.

[8] Luk M, Mezzour G, Perrig A, Gligor V. MiniSec: A secure sensor network communication architecture. In *Proc. the 6th IEEE International Conference on Information Processing in Sensor Networks* (*IPSN*), April 2007, pp.479-488.

[9] ISO. Information technology — Security techniques — Message authentication codes (MACs) — Part 1: Mechanisms using a block cipher. ISO9797-1, 1999. http://www.iso.org/iso/iso_catalogue/catalogue_tc, August 2013.

[10] Rogaway P, Bellare M, Black J. OCB: A block-cipher mode of operation for efficient authenticated encryption. *ACM Transactions on Information and System Security*, 2003, 6(3): 365-403.

[11] Information Technology Laboratory, National Institute of Standards and Technology of U.S. The keyed-hash message authentication code (HMAC). Federal Information Processing Standards Publication, FIPS PUB 198. http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf, Oct. 2013.

[12] Bogdanov A, Leander G, Paar C, Poschmann A, Robshaw M J B, Seurin Y. Hash functions and RFID tags: Mind the gap. In *Lecture Notes in Computer Science 5154*, Oswald E, Rohatgi P (eds.), Springer-Verlag, 2008, pp.283-299.

[13] Daemen J, Rijmen V. A new MAC construction ALRED and a specific instance ALPHA-MAC. In *Lecture Notes in Computer Science 3557*, Gilbert H, Handschuh H (eds.), Springer-Verlag, 2005, pp.1-17.

[14] Bogdanov A, Knudsen L R, Leander G *et al.* PRESENT: An ultra-lightweight block cipher. In *Lecture Notes in Computer Science 4727*, Paillier P, Verbauwhede I (eds.), Springer Heidelberg, 2007, pp.450-466.

[15] Huang J, Seberry J, Susilo W. On the internal structure of ALPHA-MAC. In *Lecture Notes in Computer Science 4341*, Nguyen P Q (ed.), Springer-Verlag, 2006, pp.271-285.

[16] Biryukov A, Bogdanov A, Khovratovich D, Kasper T. Collision attacks on AES-based MAC: ALPHA-MAC. In *Lecture Notes in Computer Science 4727*, Paillier P, Verbauwhede I (eds.), Springer-Verlag, 2007, pp.166-180.

[17] Wang W, Wang X, Xu G. Impossible differential cryptanalysis of Pelican, MT-MAC-AES and PC-MAC-AES. *Cryptology ePrint Archive*, http://eprint.iacr.org/2009/005, August 2013.

[18] Dunkelman O, Keller N, Shamir A. ALRED blues: New attacks on AES-based MAC's. *Cryptology ePrint Archive*, http://eprint.iacr.org/2011/095, August 2013.

[19] Gong Z, Hartel P, Nikova S, Zhu B. Towards secure and practical MACs for body sensor networks. In *Lecture Notes in Computer Science 5922*, Roy B K, Sendrier N (eds.), Springer-Verlag, 2009, pp.182-198.

[20] Daemen J, Rijmen V. The Pelican MAC function. *Cryptology ePrint Archive*, http://eprint.iacr.org/2005/088, August 2013.

[21] Bogdanov A, Knežević M, Leander G, Toz D, Varici K, Verbauwhede I. SPONGENT: A lightweight hash function. In *Lecture Notes in Computer Science 6917*, Preneel B, Takagi T (eds.), Springer-Verlag, 2011, pp.312-325.

[22] Wang M. Differential cryptanalysis of reduced-round PRESENT. In *Lecture Notes in Computer Science 5023*, Vaudenay S (ed.), Springer-Verlag, 2008, pp.40-49.

[23] Albrecht M, Cid C. Algebraic techniques in differential cryptanalysis. In *Lecture Notes in Computer Science 5665*, Dunkelman O (ed.), Springer-Verlag, 2009, pp.193-208.

[24] Collard B, Standaert F X. A statistical saturation attack against the block cipher PRESENT. In *Lecture Notes in Computer Science 5473*, Fischlin M (ed.), Springer-Verlag, 2009, pp.195-210.

[25] Özen O, Varici K, Tezcan C, Kocair Ç. Lightweight block ciphers revisited: Cryptanalysis of reduced round PRESENT and HIGHT. In *Lecture Notes in Computer Science 5594*, Boyd C, Nieto J G (eds.), Springer-Verlag, 2009, pp.90-107.

[26] Katz J, Lindell Y. Introduction to Modern Cryptography (Chapman & Hall/CRC Cryptography and Network Security Series). Chapman & Hall/CRC, 2007.

[27] Rogaway P. Authenticated-encryption with associated-data. In *Proc. the 9th ACM Conference on Computer and Communications Security*, November 2002, pp.98-107.

[28] Barr K C, Asanović K. Energy-aware lossless data compression. *ACM Transactions on Computer Systems*, 2006, 24(3): 250-291.

[29] Bellare M, Kilian J, Rogaway P. The security of the cipher block chaining message authentication code. *Journal of Computer and System Sciences*, 2000, 61(3): 362-399.

[30] Black J, Rogaway P. CBC MACs for arbitrary-length messages: The three-key constructions. *Journal of Cryptology*, 2005, 18(2): 111-131.

[31] Ferguson N. Collision attacks on OCB. http://csrc.nist.gov, August 2013.

[32] Black J, Halevi S, Krawczyk H, Krovetz T, Rogaway P. UMAC: Fast and secure massage authentication. In *Lecture Notes in Computer Science 1666*, Wiener M (ed.), Springer-Verlag, 1999, pp.216-233.

[33] Bellare M, Canetti R, Krawczyk H. Keying hash functions for message authentication. In *Lecture Notes in Computer Science 1109*, Koblitz N (ed.), Springer-Verlag, 1996, pp.1-15.

[34] Preneel B, van Rompay B, Örs S B *et al.* Performance of optimized implementations of the NESSIE primitives (v2.0 edition). In *The NESSIE Consortium*, http://www.cosic.esat.kuleuven.be/nessie/deliverables/D21-v2.pdf, August 2013.

[35] Paar C, Poschmann A, Robshaw M J B. New designs in lightweight symmetric encryption. In *RFID Security: Techniques, Protocols and System-on-Chip Design*, Kitsos P, Zhang Y (eds.), Springer, 2008, pp.349-371.

[36] Feldhofer M, Rechberger C. A case against currently used hash functions in RFID protocols. In *Lecture Notes in Computer Science 4277*, Meersman R, Tari Z, Herrero P (eds.), Springer-Verlag, 2006, pp.372-381.

[37] ISO. Information technology – Security techniques – Hash-functions – Part 2: Hash-functions using an n-bit block cipher algorithm. ISO/IEC10118-2, 2010. http://www.iso.org/iso/home/store/catalogue_tc, August 2013.

[38] Black J, Rogaway P, Shrimpton T. Black-box analysis of the block-cipher-based hash-function constructions from PGV. In *Lecture Notes in Computer Science 2442*, Yung M (ed.), Springer, 2002, pp. 320-335.

[39] Knudsen L, Mendel F, Rechberger C, Thomsen S. Cryptanalysis of MDC-2. In *Lecture Notes in Computer Science 5479*, Joux A (ed.), Springer, 2009, pp.106-120.

[40] Lai X, Massey J. Hash functions based on block ciphers. In *Lecture Notes in Computer Science 658*, Rueppel R A (ed.), Springer, 1993, pp.55-70.

[41] Healy M, Newe T, Lewis E. Analysis of hardware encryption versus software encryption on wireless sensor network motes. In *Lecture Notes in Electrical Engineering 20*, Mukhopadhyay S C, Gupta G S (eds.), Springer, 2008, pp.3-14.

[42] Moradi A, Poschmann A, Ling S, Paar C, Wang H. Pushing the limits: A very compact and a threshold implementation of AES. In *Lecture Notes in Computer Science 6632*, Paterson K G (ed.), Springer-Verlag, 2011, pp.69-88.

Zheng Gong received a Ph.D. degree in computer science from Shanghai Jiao Tong University, China, in 2008. From Dec. 2008 to Jan. 2012, he was a postdoctoral researcher in the DIES group of Twente University, Netherland. Currently he is an associate professor of computer science at South China Normal University, Guangzhou. His recent research directions are cryptography and provable security, including the design and analysis of block ciphers, hash functions and message authentication codes.

Pieter Hartel received his Ph.D. degree in computer science from the University of Amsterdam in 1989. He has worked at CERN in Geneva (Switzerland), the Universities of Nijmegen, Amsterdam (The Netherlands), and Southampton (UK). He is currently a full professor of computer science at the University of Twente. His research interest is information security.

Svetla Nikova got her Master degree in mathematics in Sofia University, Bulgaria, and her Ph.D. degree in Eindhoven University of Technology, The Netherlands. From 1999 till 2008 she was a postdoctoral researcher in ESAT/COSIC, Katholieke Universiteit Leuven (KU Leuven), Belgium. From 2008 till 2012 she worked as an assistant professor in the University of Twente. Since March 2012, she is a senior researcher in COSIC, KU Leuven. Svetla Nikova's research expertise is in cryptography, in particular lightweight cryptographic algorithms, side-channel resistant implementations, symmetric crypto primitives.

Shao-Hua Tang received the B.Sc. and M.Sc. degrees in applied mathematics in 1991 and 1994, respectively, and the Ph.D. degree in communication and information system, in 1998, all from South China University of Technology. He was a visiting scholar with North Carolina State University, USA, in 2001∼2002, and a visiting professor with University of Cincinnati, USA, in 2009∼2010. He has been a full professor with the School of Computer Science and Engineering, South China University of Technology since 2004. His current research interests include information security, networking, and information processing. He is a member of the IEEE.

Bo Zhu received his M.Sc. degree in computer science from Shanghai Jiao Tong University, China, in 2010, and his B.Eng. degree in electrical and computer engineering from Tsinghua University, Beijing, in 2007. Currently he is a Ph.D. candidate in electrical and computer engineering at University of Waterloo. His research interests lie in analysis and design of block ciphers, stream ciphers, and modes of operations.