

Towards Secure and Practical MACs for Body Sensor Networks*

Zheng Gong¹, Pieter Hartel¹, Svetla Nikova^{1,2} and Bo Zhu³

¹Distributed and Embedded Security Group, Faculty of EWI,
University of Twente, The Netherlands
{z.gong, pieter.hartel, s.nikova}@utwente.nl

² Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC,
Kasteelpark Arenberg 10, B-3001 Heverlee, Belgium

³Department of Computer Science and Engineering,
Shanghai Jiaotong University, China
zhuo03@gmail.com

Abstract

Wireless sensor network (WSN) commonly requires lower level security for public information gathering, whilst body sensor network (BSN) must be secured with strong authenticity to protect personal health information. First in this paper, some practical problems with the Message Authentication Codes (MACs), which are suggested in the current security architectures for WSN, are reconsidered. The analysis exploits the fact that the recommended MACs for WSN, e.g., TinySec (CBC-MAC), MiniSec (OCB-MAC), and SenSec (XCBC-MAC), are not exactly suitable for BSN. Particularly a dedicated attack is elaborated on the XCBC-MAC. Considering the hardware limitations of BSN, we propose a tunable lightweight MAC based on the PRESENT block cipher, which is named TuLP. A 128-bit variant TuLP-128 is proposed for a higher resistance against internal collisions. Compared to the existing schemes, our lightweight MACs show better performance and higher energy-efficiency at lower memory usage.

Key words. Message authentication code, Body sensor network, Resource-constrained implementation.

1 Introduction

Body sensor network (BSN, also called wireless medical sensor network) [34], which can be developed from wireless sensor network (WSN), is a key technology for long term monitoring of biological events or any abnormal condition of patients for realizing the Ambient Assisted Living (AAL) vision [1]. Although the fact that large groups of patients already carry individually implantable or wearable monitoring equipments, a biosensor array (also implantable or wearable) offers a more accurate status than one isolated device. To offer more personalized health care to elderly or disabled patients, the gathered health information will instantly be sent to the server of a clinic or hospital, which will be monitored by doctors (or nurses) to prevent the occurrence of fatal events. Since BSN nodes are either

*The first author acknowledges the financial support of SenterNovem for the ALwEN project, grant PNE07007. The last author is supported by NSFC (No.60803146) and National Basic Research Program (973) of China (No.2007CB311201).

worn or implanted by a patient, the power consumption should be low to minimize radiation and maximize durability. Moreover, BSN sensors also have limited computational ability and memory. These factors are important not only in the implantable but also in the external sensor settings because they determine how “hidden” and “pervasive” the sensors are. Table 1 shows the specifications of typical BSN nodes in practice.

Traditional WSNs are used to collect public information in the environment, such as temperature, humidity, fire alarm, etc. Since monitored health data from a person with BSN will be a part of personal Electronic Health Record (EHR), a higher level of assessment and protection is required for BSN communications. The existing EHR standards (ISO 27001, 27799, openEHR/ISO 18308, etc.) oblige BSN to be secured with strong cryptography. However, strong cryptography entails more resources. Considering the highly constrained resources that a BSN node can have, a better trade-off has to be found such that the security is maximized, while minimizing the resources. Unfortunately, because of the heterogeneity of BSN, the secure protocols for static networks might not be applicable for BSN. Also the methods proposed for *ad hoc* networks such as asymmetric cryptography techniques would be costly for BSN applications. Due to the constraints in power consumption and computational ability, it remains a great challenge to design secure and practical cryptographic primitives which are both time and resource efficient for BSN applications.

	TI Node ¹	MICAz Node ²	MyriaNed ³
CPU	16bit, 8MHz	8bit, 16MHz	16bit, 32MHz
RAM	2KB	4KB	8KB
Flash Memory	64KB	128KB	128KB
Voltage	1.8 ~ 3.6v	2.7 ~ 3.3v	1.6 ~ 3.6v
OS	TinyOS	TinyOS	MyriaCore

Table 1: The specifications of typical BSN nodes.

Related Work. To ensure the authenticity and integrity of WSN communication, security protocols via different Message Authentication Codes (MACs, different from the term “Medium Access Control”) are proposed. 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. One widely used method is the Security Protocol for Sensor Networks (SPINS) [29], which consists of μ TESLA (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 [22], SenSec [25] and MiniSec [26]. All these architectures considered which MAC will be suitable in the WSN packet/message authentication. For instance, TinySec and MiniSec recommend the well-known CBC-MAC [22] and OCB-MAC [26] respectively, whilst SenSec uses a novel scheme called XCBC-MAC [25]. All the recommended MACs are based on the operation modes of block cipher, and suggest 32-bit length tag for WSN. In contrast, since dedicated hash functions (such as MD5 and SHA-1) are primarily designed to be collision resistant for preventing forgery of digitally signed documents, it was exploited that MACs based on hash functions (e.g., HMAC [3]) might be less competitive than

¹Texas Instruments. <http://focus.ti.com/lit/ds/symlink/msp430f149.pdf>.

²Crossbow. http://www.xbow.com/products/Product_pdf_files/Wireless_pdf/MICAZ_Datasheet.pdf.

³ALWEN project. http://www.atmel.com/products/AVR/default_xmega.asp.

block-cipher-based ones for highly constrained devices [14]. Nevertheless, it is recognized by the BSN research community that authentication in BSN protocols is usually for short messages [18] in network processing. This property implies that the candidates of MACs, which focus more on the one-wayness than on the collision-resistance, will be more practical for BSN applications.

Since typical BSN nodes have limited resources, an appropriate security level should be imposed to realize authenticity and confidentiality in applications. Intuitively, 32-bit security level for WSN is not suitable even for the one-wayness of the transmitted data in BSN. 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 safely kept by the member banks. In other applications, certain authorities even recommended to implement a MAC with a longer length of 128-bit. Although an appropriate security level for BSN applications will be ensured case by case, but 64-bit security bound is widely-accepted for resisting sensible threats in such constrained devices. As power and RAM are normally the most constrained resources on a BSN node, the design of a MAC should consider applicable trade-offs towards time and resource efficiency in practice.

Our Contributions. The contributions of this work are three-fold. Firstly, 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). Especially 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 MACs from different design principles, e.g., CBC-MAC, OCB-MAC, ALPHA-MAC [16]. Thirdly, taking into account the requirements for authenticity in BSN, we propose a tunable lightweight MAC based on the PRESENT block cipher [13], which is named TuLP. The structure of TuLP is inspired by the generic construction ALRED [16]. A 128-bit variant TuLP-128 is proposed for the higher resistance against internal collisions. Compared to the existing schemes, our lightweight MACs show a better performance on MICAz node with less memory usage in ROM/RAM, and also energy-efficient in the level of gate equivalents.

Organization. The remainder of this paper is organized as follows. Section 2 describes some definitions and notions which will be used throughout the paper. The problems with the MACs recommended in the proposed security architectures for WSN are described in Section 3. Section 4 gives a performance comparison of some efficient MACs for BSN authenticity. The designs of TuLP and TuLP-128 follow in Section 5 along with a detailed analysis of the security and the performance. Section 6 concludes the paper.

2 Preliminaries

Here we review some definitions and primitives which will be used in the following sections. Exclusive-or (xor) will be denoted by \oplus . $a||b$ denotes the concatenation of two strings a and b . Let \mathcal{M} and \mathcal{K} be the message and key spaces respectively.

2.1 Cryptographic Primitives

ALRED. The ALRED construction is a generic MAC design introduced by Daemen and Rijmen [16]. 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 bits of the message to the same dimensions as a sequence of r round keys of E . Then apply a sequence of r times round function of E to the state by using the output of the injection layout as the round keys.
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.

Based on the ALRED construction, Daemen and Rijmen also presented two paradigms called ALPHA-MAC [16] and Pelican [17], by using AES as the underlying block cipher. Recently, many papers exploited that ALPHA-MAC and Pelican might be threatened under the internal collisions [21], the side-channel attack [9] and the impossible differential analysis [33]. We note that all those cryptanalyses are based on the internal structures of ALPHA-MAC and Pelican, which do not endanger the security of the generic construction of ALRED.

PRESENT. At CHES 2007, Bogdanov *et al.* proposed an ultra-lightweight block cipher which is named PRESENT [13]. PRESENT is an example of an SP-network 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 0.18 μ m 1P6M Logic Process (UMCL18G212T3)*, PRESENT-80 and PRESENT-128 are estimated to require 1570 and 1886 gate equivalents, respectively [13]. Since Bogdanov *et al.* do 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 Figure 1: First, 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 bit-oriented permutation, which will be extremely hardware-efficient, is executed to provide diffusion.

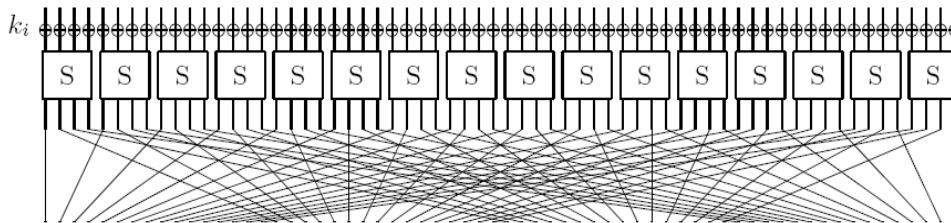


Figure 1: Round function of PRESENT [13].

Further details about the specification of PRESENT can be found in Bogdanov *et al.* [13], including basic results of the differential and linear cryptanalyses, which can be summarized as follows.

Theorem 1 *Any five-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 $\epsilon_{4R} \leq 2^{-7}$.

Based on PRESENT, Bogdanov *et al.* [14] propose some low-energy block-cipher-based hash functions (e.g., single and double block length construction DM-PRESENT and H-PRESENT respectively) which are more practical than dedicated or AES-based hash functions in highly constrained devices, such as RFID tags. Recently, many cryptanalysis results have been given on the PRESENT block cipher. Wang [32] presents a new 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. Collard and Standaert [15] show a statistical saturation attack against 24 PRESENT-80. The saturation attack depends on a simplified key schedule algorithm such that the same subkey should be used in each round. Özen *et al.* [27] provide a related-key rectangle attack on 17-round PRESENT-128. However the known attacks on full PRESENT with 80-bit keys, without any simplification, so far are bounded with 16 rounds [32].

2.2 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. Figure 2 shows two different paradigms to build up a secure packet with the properties of authenticity and confidentiality. The left paradigm is borrowed from Rogaway’s *Authenticated Encryption with Associate Data* [30]. If the message is long, the authentication includes the full ciphertext will be costly.

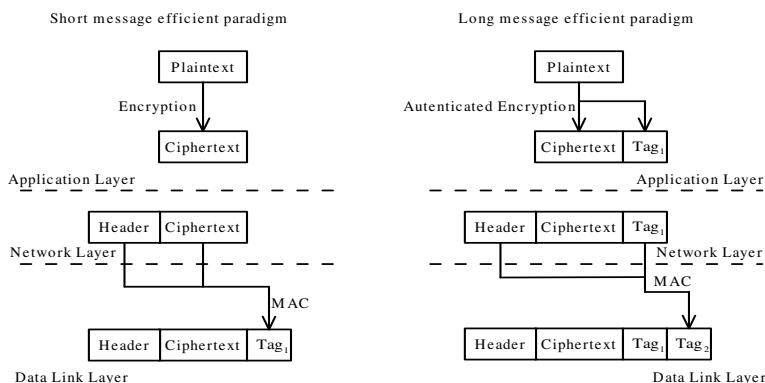


Figure 2: Two authentication paradigms for BSN.

The right paradigm trades off the overhead on an extra tag to avoid the authentication 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 $\text{MAC}(k, \text{Header}, \text{Tag}_1) = \text{Tag}_2$. If it does not hold, the receiver will just ignore the packet without decryption. Otherwise the receiver checks the validity of the ciphertext through the equation $\text{MAC}(k, \text{Ciphertext}) = \text{Tag}_1$, and then decrypts it to obtain the valid 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 [6] states that communicating a single bit of data consumes several orders of magnitude more power than executing a basic 32-bit arithmetic instruction. If the message is short, one can choose Rogaway’s *Authenticated Encryption with Associate Data* paradigm since the communication costs on the extra tag will be larger than the authentication which includes the full ciphertext. Otherwise, the right paradigm might be efficient since both the sender and the receiver require less computation on authenticating with a long ciphertext. No matter what authentication mode is opted, the security and performance of the underlying MAC function will play a pivotal role for BSN security.

3 Problems with the 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, data link layer security 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 [22], SenSec [25], and MiniSec [26]. 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 [22, 25, 26].

CBC-MAC. In TinySec [22], Karlof *et al.* suggest to use CBC-MAC [4] 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. CBC-MAC is provably secure as well [8]. For BSN applications, the disadvantage of CBC-MAC is that independent keys should be used for encryption and authentication. Furthermore, the standard CBC-MAC construction is not secure for variable length messages. Adversaries can forge a MAC for certain messages. To preserve the provable security for variable length messages, a variant of CBC-MAC uses three different keys for the authentication [11]. The three-key construction solves the variable length message problem, and avoids unnecessary message padding, but it raises another typical risk with respect to the key management in BSN. Compared to the one-key constructions, the extra keys will impose a heavy burden on the key generation, distribution and storage, which indicates that CBC-MAC might be less practical for BSN applications.

XCBC-MAC. The XCBC-MAC algorithm proposed by Li *et al.* in [25] is part of the authenticated encryption mode for SenSec. Let k_A and k_E be the authentication key and the encryption key, respectively. Let message $M = m_1 || m_2 || \dots || m_t$. Figure 3 depicts the construction of XCBC-MAC. In general, the XCBC-MAC algorithm can be viewed as a variant of the two-key CBC mode. Unfortunately, we have found a practical existential forgery on XCBC-MAC by implementing a 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:

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

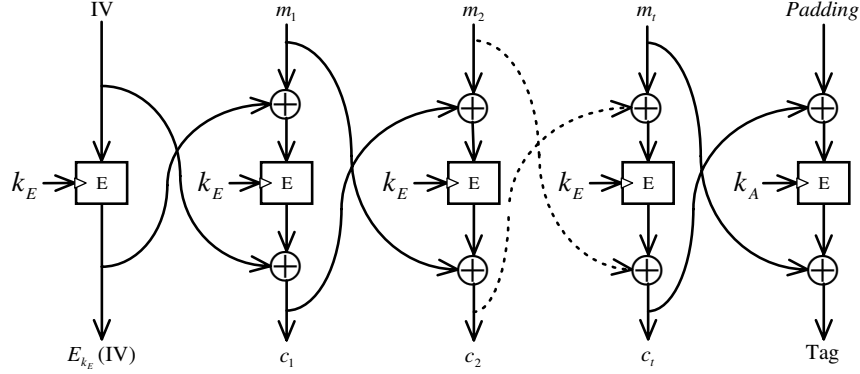


Figure 3: The XCBC algorithm proposed in SenSec [25].

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 chooses a padding message M' , and then outputs two different messages M_1, M_2 , where $M_1 = E_{k_E}(IV) \oplus m_1 || E_{k_E}(m_1) || M'$ and $M_2 = E_{k_E}(IV) \oplus m'_1 || E_{k_E}(m'_1) || M'$.

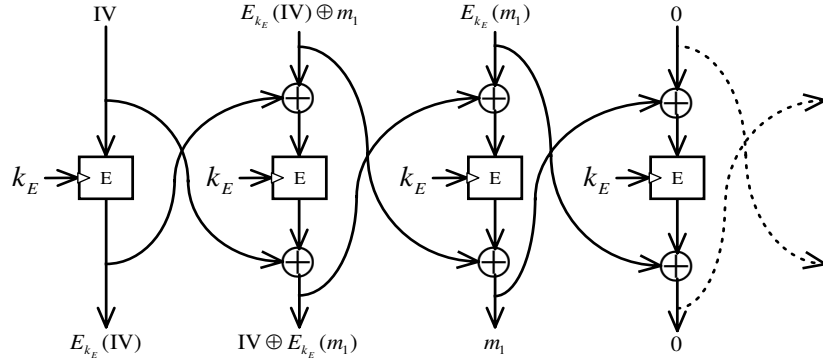


Figure 4: An existential forgery under the XCBC-MAC.

Based on the above attack, the two different messages M_1 and M_2 will have the same MAC since both the prefixes $E_{k_E}(IV) \oplus m_1 || E_{k_E}(m_1)$ and $E_{k_E}(IV) \oplus m'_1 || E_{k_E}(m'_1)$ produce the same zero output to the next step. The attack is difficult to detect since the prefixes are computationally indistinguishable from a randomized query. The attack on XCBC-MAC cannot be avoided by a randomized IV, since IV should be a public-known value and all nodes use the same IV. If IV is frequently changed, all nodes should be synchronized, otherwise the receiver cannot correctly decrypt the packet from the sender. Since synchronization is costly in BSN, it is impractical for an IV to be distributed just for one-time usage. Due to the above attack, the XCBC-MAC algorithm proposed in SenSec [25] is insecure under the chosen message attack and should be abandoned in any circumstance of authentication.

OCB-MAC. In MiniSec [26], Luk *et al.* suggest using the OCB mode [31], which is an efficient authenticated encryption scheme, as the MAC function for message authenticity and integrity. Since its publication OCB has received some attention, but little cryptanalysis. We believe this has two reasons. First, a proof of security seems to imply that cryptanalysis is useless. The proof is quite complicated and analysis of the proof details is restricted to those people who are well-versed in formal proof techniques. Second, the OCB mode has been patented. There is a significant cost, both directly and indirectly, associated with using a patented algorithm. The last reason is the main reason for the lack of rigorous cryptanalysis. Spending time on OCB will only help the patent-holders to sell their licenses without any further compensation to the cryptanalyst. Moreover, Ferguson also presents a collision attack on OCB with arbitrary length messages [19]. To keep adequate authentication security of OCB, one has to limit the amount of data that the MAC algorithm processes. Since the offset values used in OCB require extra time/memory costs with respect to the message length, the area and the power consumption will be increased for the computation and storage. The above reasons are relevant to real-life applications on BSN, and cast doubts on the wisdom of using OCB.

4 A 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 MAC Functions 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 MAC functions. The first is to design a new primitive from scratch, such as UMAC [10]. The second is to define a new mode of operation for existing primitives. Such as variants of encryption modes of block ciphers: CBC-MAC [4] and OCB-MAC [31]; Or variants mode of hash functions: HMAC/NMAC [3, 7]. The third approach, which can be viewed as a hybrid of the first and the second approach, is to design new MAC functions using components of existing primitives, such as ALPHA-MAC [16].

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 from the three different approaches. To be fair, all MACs based on block cipher use AES-128 as the underlying block cipher, as well as input messages can be of arbitrary length. The timing of the keysetup and the message processing are estimated from the performance data given by the NESSIE consortium [5] (Pentium III/Linux Platform), such that the message processing time is measured in cycles/byte, while the key-setup and keysetup + 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 [28]. By following the same method [14, 18], we also use the *Virtual Silicon* (VST) standard cell library based on *UMC L180 0.18 μ m IP6M Logic Process* (UMCL18G212T3) to estimate each area in GE of the candidates. According to the related experiments [18], the area GE for AES-128 encryption is estimated to be 3400 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 larger size of the chip the larger energy costs, whilst the dynamic part is proportional to the operating frequency. For

	CBC-MAC [11]	OCB-MAC [31]	ALPHA-MAC [16]	HMAC/SHA-1 [3]
Based on	cipher mode	cipher mode	AES components	hash mode
Keysetup	616	644	1032	1346
Finalization	1440	1444	416	3351
Message processing	26	31	10.6	15
Area in GE (estimate)	4764	6812	4424	8120 [18]

Table 2: The comparison of some practical MAC functions.

the devices with a lower operating frequency, the static power consumption is the most significant. For 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 has merits on both of 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, first we will propose a lightweight MAC variant from a modification of the ALRED construction, which is named TuLP. To raise the security bound of resisting internal collisions, we also give a wide-pipe version of TuLP, which is called TuLP-128. Both our schemes use the experiences of ALPHA-MAC [16] and Pelican [17]. Next, a security analysis of our schemes will be given. Finally, the performance of our lightweight schemes will be analyzed. Compared to the results in Table 2, our new MAC functions are time-efficient with less memory usage, and also energy-efficient in the number of gate equivalents.

5.1 TuLP and TuLP-128

By using the components of the block cipher PRESENT [13], first a new MAC function TuLP is built from 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 $|k| \leq 80bits$. If $|k|$ is less than 80 bits, it should be iteratively padded with 1 or 0 as 10101... 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 $m_1, m_2, \dots, m_t, t = \frac{|pad(M)|}{64}$, 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 , such that

$$s_0 = E_k(IV).$$

Obtain the output s_0 as the initial state.

3. **Compression.** For each message block m_i where $i \in \{1, 2, \dots, t\}$, 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 $\text{MSB}^{16}(k)$). By executing the same key schedule algorithm of PRESENT, apply r PRESENT round functions on the state s_{i-1} , such that

$$s_i = \rho_{m_i \oplus s_{i-1} || \text{MSB}^{16}(k)}^r(s_{i-1}).$$

4. **Finalization.** Apply one full-round PRESENT encryption to the state s_t under the key k , and then truncate the first ℓ_m bits of the final state s_{t+1} as the tag of the message M .

$$s_{t+1} = E_k(s_t), \text{tag}_M = \text{Trunc}^{\ell_m}(s_{t+1}).$$

Since the width of the intermediate state of TuLP is only 64 bits, it is not enough to resist the birthday attack on internal states for key non-recovery forgery. Although this “weakness” is not fatal in some applications of BSN, we still provide a wide-pipe version, which is called TuLP-128, to increase the state and the maximum tag length to be 128-bit length. The key length of TuLP-128 is up to 160 bits. We note that the design principle is inspired by MDC-2 and the padding rule is identical to TuLP.

1. **Padding.** Let k be an authentication key such that $|k| \leq 160\text{bits}$. If $|k|$ is less than 160 bits, it should be iteratively padded with 1 or 0 as 10101 \dots . 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. Split the result $\text{pad}(M)$ into 64-bit blocks $m_1, m_2, \dots, m_t, t = \frac{|\text{pad}(M)|}{64}$, such that

$$\text{pad}(M) = M || \lambda(M, k) || 10^d.$$

2. **State Initialization.** Splitted the (padded) authentication key k into two 80-bit key $k_l || k_r$. Then apply one full-round PRESENT encryption to two 64-bit different 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 $s_{r,0}$.

$$s_{l,0} = E_{k_l}(\text{IV}_1), s_{r,0} = E_{k_r}(\text{IV}_2).$$

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

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

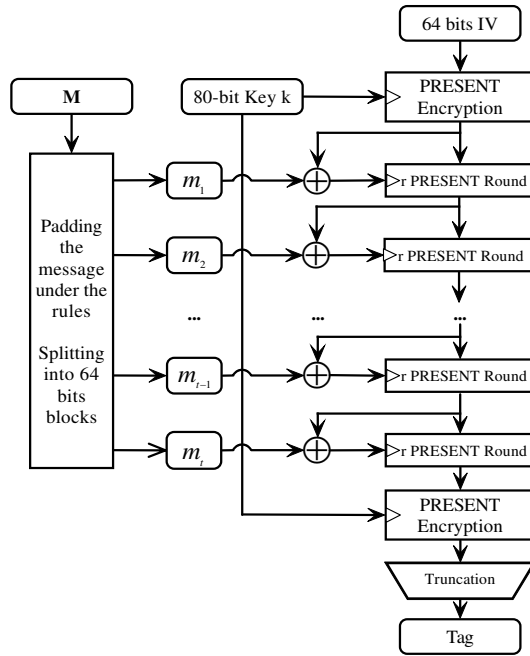


Figure 5: The illustration of TuLP.

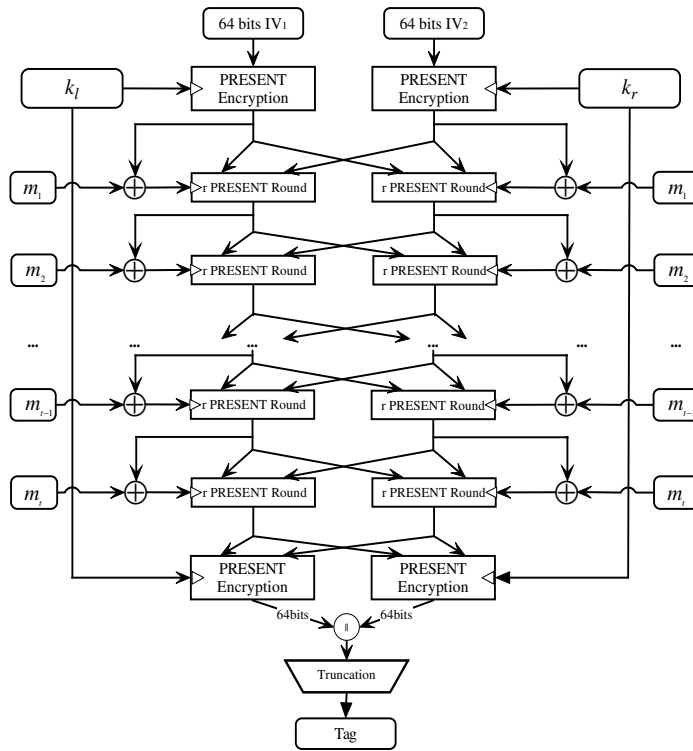


Figure 6: The illustration of TuLP-128.

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

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

Figure 5 and 6 depict the high-level algorithms of TuLP and TuLP-128, respectively. Referring to the issues of ALPHA-MAC and Pelican [9, 14, 33], the advantages of our schemes are as follows.

- In ALPHA-MAC [16], 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.* [9] 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. In TuLP, round keys are not computed from a deterministic function of input message blocks. Thus, a side-channel 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 [9] on TuLP and TuLP-128.
- Like in Pelican [17], 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 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 by Black *et al.* [12].
- The bitwise lengths of message and key are appended to the end of the message. This message padding rule can avoid some trivial attacks on the internal collision and the extension. ALPHA-MAC and Pelican only pad message with a single 1 followed by the minimum number of 0 bits to suffice a block.
- Benefit from the ALRED construction, the security of our schemes can be reduced to the security of PRESENT if internal collisions are not involved. The proofs are provided in the security analysis of Section 5.2. Since the compressions in TuLP and TuLP-128 are different from the PRESENT encryption, encryption and authentication can use the same secret key.
- TuLP is designed for rapid message processing. The computational costs 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 costs of message processing only requires $\frac{2 \cdot r}{31}$ of one PRESENT encryption.
- The choice of r rounds PRESENT in the compression is *tunable* by consideration of the practical balance of security and performance. Since key management in sensor network is expensive on computation and energy, the length of authentication key is *tunable* since the message padding rules considered dynamic key length. To give practical instances for the analysis in the following section, we will consider $r = 16$ in the compression of TuLP and TuLP128, where $\text{IV} = \text{IV}_1 = 0123456789\text{ABCDEF}$ and $\text{IV}_2 = \text{FEDCBA9876543210}$.

5.2 Security Analysis

In this section, 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 give a synthetic analysis of TuLP when internal collisions are considered.

Since the ALRED construction has a similar internal structure with the CBC mode, which typically implies the security between the construction and the underlying cryptographic primitives. Derived from the provability results of the ALRED construction in [16], it is easy to derive a similar result on TuLP as follows.

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 $i \in \{1, 2, \dots, t\}$. Then we derive a key recovery attack on the PRESENT block cipher as follows.

1. Request the first state $s_0 = E_k(\text{IV})$.
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 $\text{tag}_i = \text{Trunc}(E_k(s_i))$.
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 $E_k(\text{IV})$. So the theorem follows. \square

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 $i \in \{1, 2, \dots, t\}$. Then we derive a ciphertext guessing attack on the PRESENT block cipher as follows.

1. Request the first state $s_0 = E_k(\text{IV})$.
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 $\text{tag}_i = \text{Trunc}(E_k(s_i))$.
4. Submit t tag values to \mathcal{A} to obtain an existential forgery tag' , tag' should also be a valid ciphertext on the message m' .

The above attack requires t chosen messages and one chosen message on $E_k(\text{IV})$. So the lemma follows. \square

Lemma 2 *Any existential forgery attack on TuLP with a fixed point $\{(m, s) | E_{m \oplus s}(s) = s, m \in \mathcal{M}, s \in \mathcal{K}\}$ requiring t (adaptively) chosen messages, can be converted to a fixed point attack $\{(m', k) | E_{m'}(k) = k, m \in \mathcal{M}, k \in \mathcal{K}\}$ 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 $i \in \{1, 2, \dots, t\}$. Then we derive a fixed point attack on the PRESENT block cipher as follows.

1. Request the first state $s_0 = E_k(\text{IV})$.
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 $tag_i = \text{Trunc}(E_k(s_i))$.
4. Submit t tag values to \mathcal{A} to obtain a fixed point such that $E_k(s_i) = s_i$.

The above attack requires t chosen messages and one chosen message on $E_k(\text{IV})$. So the lemma follows. \square

Now we analyze the security with respect to internal collisions. The reason 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 with 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.* [13], 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. One S-box provides maximum 2^{-2} possibility for differential characteristic, thus 16 rounds provide a lower bound $(2^{-2})^{r*10/5} \approx 2^{-64}$ 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 maximal bias of a linear approximation $\epsilon_{4R} \leq 2^{-7}$. Hence 16 rounds provide maximum bias of a linear approximation $(2^{-7})^{r/4} = 2^{-28}$.

2. This linear cryptanalysis would require the memory complexity of about $1/(2^{-28})^2 = 2^{56}$ (2^{19} TB) known plaintext/ciphertext pairs.

So the theorem follows. □

Consider a typical BSN application consisting of 100 nodes. If each node transfers an 8-byte message per 15 seconds for monitoring, the time consumption of obtaining 2^{56} plaintext/ciphertext pairs for the linear analysis would be impractical.

By using multi-collisions, Knudsen *et al.* [23] provide a collision attack and preimage attacks on the MDC-2 construction with the time complexities of about $(\log_2(n)/n) \cdot 2^n$ 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 [24] 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 5 *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 of $r = 16$ in the compression of TuLP-128. One S-box provides a maximum 2^{-2} possibility for differential characteristic, 16 PRESENT round functions 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.* [12], and TuLP-128 has a 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 [23], the internal collision attack and the preimage attack on TuLP-128 would require the time complexity of about $(\log_2(64)/64) \cdot 2^{64} \approx 2^{61.3}$ 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} . So the theorem follows. □

Although TuLP-128 doesn't achieve the ideal upper bounds of collision and preimage resistances, the MDC-2 like structure in TuLP-128 also yields practical advantages. For example, symmetric left and right pipes can minimize the area, and the simple permutation layer between left and right states saves extra logical gates. Nevertheless, a $2^{61.3}$ level of time complexity on finding an internal collision is still beyond the computational bound in practice. Now we consider the security of TuLP-128 without internal collisions.

Theorem 6 *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 $i \in \{1, 2, \dots, t\}$. Let

ρ be the compression function of TuLP. $\text{MSB}^{32}(\cdot)$ and $\text{LSB}^{32}(\cdot)$ 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 first left and right states $s_{l,0} = E_k(\text{IV}_1)$ and $s_{r,0} = E_k(\text{IV}_2)$.
2. For $i = 1$ to t , compute the left intermediate state $s_{l,i} = \chi(\text{MSB}^{32}(s_{l,i}) || \text{MSB}^{32}(s_{r,i}), m_i)$, and the right intermediate state $s_{r,i} = \chi(\text{LSB}^{32}(s_{l,i}) || \text{LSB}^{32}(s_{r,i}), m_i)$, where χ denotes the compression function of TuLP.
3. For $i = 1$ to t , request $\text{tag}_i = \text{Trunc}(E_k(s_{l,i}) || E_k(s_{r,i}))$.
4. Submit t tag values to \mathcal{A} to recover the key k .

The above attack needs t chosen messages except $E_k(\text{IV}_1)$ and $E_k(\text{IV}_2)$. So the theorem follows. \square

Based on the proofs of Lemma 1 and Lemma 2, it is easy to obtain the following lemmas on TuLP-128. The proofs are omitted here for brevity.

Lemma 3 *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.*

Lemma 4 *Any existential forgery attack on TuLP-128 with a fixed point of requiring t (adaptively) chosen messages, can be converted to a fixed point attack on PRESENT requiring $t + 2$ adaptively chosen plaintexts.*

5.3 Performance

Before we study the performance of TuLP and TuLP-128, first we realize an optimized implementation of PRESENT by using 1K bytes look-up table on MICAz nodes. Our implementation of PRESENT is the first software result in the literature. From our performance experiments, we find that the bit permutation of PRESENT is costly in software implementation. Compared to the best known result of AES software implementation on MICAz nodes [20], our optimized implementation of PRESENT still shows a competitive speed and lower memory costs. Since PRESENT has already been proven to be a better choice than AES in hardware implementation [14], our optimized implementation shows that PRESENT is also practical in software.

Algorithm	Software (MICAz)			Hardware [14]		
	RAM	ROM	Processing speed per block	Logic process	Cycles per block	Area in GE
AES [14, 20]	1915	12720	1.46ms	0.35 μm	1032	3400
PRESENT-80	1040	1926	1.82ms	0.18 μm	32	1570

Table 3: The comparison of the implementations of AES and PRESENT.

Subsequently, we choose the DM-PRESENT [14], which is derived from the Davies-Meyer construction and the PRESENT with an 80-bit key, as the underlying hash function to build up a comparable algorithm from the HMAC [3]. We also choose PRESENT-based CBC-MAC as a benchmark

for comparability. The area in GE is estimated by using the *Virtual Silicon* (VST) standard cell library based on *UMC L180 0.18 μ m 1P6M Logic Process* (UMCL18G212T3). All experiments are based the MICAz nodes (*TinyOS version 2.10*), which are popular in both of WSN and BSN. The results in the entries of processing speed (in milliseconds) are averaged by iterating 100 times experiments with/without the optimization in the keysetup.

	TuLP	TuLP-128	CBC-MAC (PRESENT)	HMAC (DM-PRESENT)
Key Length (bits)	80	160	80	80
Intermediate State (bits)	64	128	64	64
RAM / ROM Costs (bits)	1048 / 3302	1056 / 3718	1040 / 2970	1056 / 3484
Area in GE (estimate)	2250	2566	2252	2213 [14]
Processing Speed (ms)	TuLP	TuLP-128	CBC-MAC (PRESENT)	HMAC (DM-PRESENT)
8 bytes	4.46 / 6.63	8.91 / 13.24	6.51	10.90
16 bytes	5.59 / 7.75	11.17 / 15.49	8.70	13.08
32 bytes	7.87 / 10.03	15.72 / 20.05	13.05	17.43
64 bytes	12.39 / 14.56	24.76 / 29.09	21.77	23.97
128 bytes	21.43 / 23.59	42.84 / 47.17	39.20	37.04
256 bytes	39.50 / 41.67	79.00 / 83.33	74.06	65.35
512 bytes	75.65 / 77.81	151.53 / 155.66	143.78	122.01
1024 bytes	147.94 / 150.10	295.97 / 300.31	283.21	233.04

Table 4: The comparison amongst some PRESENT-based MAC functions.

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 TuLP approaches 1.6 and 1.8 times faster than HMAC with DM-PRESENT and PRESENT-based CBC-MAC respectively, where message length from 8 bytes to 1024 bytes. The keysetup costs in our schemes, which require one (or two) PRESENT encryption(s) to generate an encrypted IV, mainly lacks TuLP (or TuLP-128) in processing the messages shorter than 32 bytes. We note that the keysetup costs 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. We note that HMAC also can precompute the initialization values for optimization, but the values must be treated and protected (128 bits for a certain key in DM-PRESENT) in the same manner as secret keys [3]. While the optimization for our schemes only increases a smaller storage (one encrypted IV is 64-bit) and no need to be insulated. Although the lengths of intermediate state and output are doubled, the performance of TuLP-128 is still comparable to PRESENT-based CBC-MAC. Obviously, TuLP-128 will be faster than HMAC with PRESENT in a double block length construction. Nevertheless, if a higher security bound is required, one can tune the rounds in the compressions of TuLP and TuLP-128. For instance, increase 16 rounds to 20 will cost about $4/16 = 25\%$ performance in message processing. Whilst a 20-round differential characteristic of PRESENT still has a probability 2^{-50} , and the linear approximation would require of about 2^{70} known plaintext/ciphertext with a bias 2^{-35} .

6 Conclusion

By considering the restrictions of BSN, two lightweight MACs TuLP and TuLP-128 have been proposed. The security of our schemes is analyzed with respect to the cryptanalyses on ALRED and

the results on PRESENT. The statistics strongly support that TuLP and TuLP-128 will be efficient and low-energy costs in devices with constrained resources. The provability of our schemes depends on the quality of the cryptanalysis that has been previously performed on those building blocks. The key length and the number of round functions in the compression are parameterized in our lightweight MACs, which support tunable tradeoffs between security and performance in BSN applications. 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.

Acknowledgement. We would like to thank Vicent Rijmen, Xuejia Lai for their advice on the design of TuLP and TuLP-128. And also thanks many anonymous reviews for their valuable comments to improve the presentation of this paper.

References

- [1] The Ambient Assisted Living (AAL) Joint Programme. European Union. <http://www.aal-europe.eu/about-aal>. Jan 2008.
- [2] M. Albrecht and C. Cid. Algebraic Techniques in Differential Cryptanalysis. In: O. Dunkelman (Ed.) *FSE 2009*, LNCS 5665, Springer, pp. 193-208, 2009.
- [3] Federal Information Processing Standard 198, The Keyed-Hash Message Authentication Code (HMAC), NIST, U.S. Department of Commerce, March 2002.
- [4] ISO/IEC 9797-1, Information technology - Security Techniques - Message Authentication Codes (MACs) - Part 1: Mechanisms using a block cipher, ISO 1999.
- [5] Performance of optimized implementations of the NESSIE primitives, version 2.0, The NESSIE Consortium, 2003, <https://www.cosic.esat.kuleuven.be/nessie/deliverables/D21-v2.pdf>.
- [6] K.C. Barr and K. Asanovic. Energy aware lossless data compression. *ACM Transactions on Computer Systems*, Volume 24, Issue 3, pp. 250-291, August 2006.
- [7] M. Bellare, R. Canetti and H. Krawczyk. Keying Hash functions for message authentication. *Advances in Cryptology-Crypto'96*, LNCS 1109, Springer-Verlag, pp. 1-15, 1996.
- [8] M. Bellare, J. Kilian and P. Rogaway. The security of the cipher block chaining message authentication code. *Journal of Computer and System Sciences*, 61(3):362-399, December 2000.
- [9] A. Biryukov, A. Bogdanov, D. Khovratovich and T. Kasper. Collision Attacks on AES-Based MAC: Alpha-MAC. In: P. Paillier and I. Verbauwhede (Eds.) *CHES 2007*, LNCS 4727, pp. 166-180, 2007.
- [10] J. Black, S. Halevi, H. Krawczyk, T. Krovetz and P. Rogaway. UMAC: Fast and Secure Message Authentication. *Advances in Cryptology-Crypto'99*, LNCS 1666, Springer-Verlag, pp. 216-233, 1999.
- [11] J. Black and P. Rogaway. CBC MACs for Arbitrary-Length Messages: The Three-Key Constructions. *Journal of Cryptology*, vol. 18, no. 2, pp. 111-131, 2005.

- [12] J. Black, P. Rogaway and T. Shrimpton. Black-Box Analysis of the Black-Cipher-Based Hash-Function Constructions from PGV. In *Advances in Cryptology-CRYPTO'02*, LNCS 2442, pp. 320-335, 2002.
- [13] A. Bogdanov, L.R. Knudsen, G. Leander, C. Paar, A. Poschmann, M.J.B. Robshaw, Y. Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In: P. Paillier and I. Verbauwhede (Eds.) *CHES 2007*. LNCS 4727, Springer, Heidelberg, pp. 450-466, 2007.
- [14] A. Bogdanov, G. Leander, C. Paar, A. Poschmann, M.J.B. Robshaw, and Y. Seurin. Hash Functions and RFID Tags: Mind the Gap. *CHES 2008*, LNCS 5154, Springer, pp. 283-299, 2008.
- [15] B. Collard and F.-X. Standaert. A Statistical Saturation Attack against the Block Cipher PRESENT. In: M. Fischlin (Ed.) *CT-RSA 2009*, LNCS 5473, pp. 195-210, 2009.
- [16] J. Daemen and V. Rijmen. A New MAC Construction ALRED and a Specific Instance ALPHA-MAC. *FSE'05*, LNCS 3557, Springer, pp. 1-17, 2005.
- [17] J. Daemen and V. Rijmen. The Pelican MAC Function. Unpublished manuscript, Available at <http://eprint.iacr.org/2005/088>.
- [18] M. Feldhofer and C. Rechberger. A Case Against Currently Used Hash Functions in RFID Protocols. In *OTM 2006 Workshops*, LNCS 4277, Springer, pp. 372-381, 2006.
- [19] N. Ferguson. Collision attacks on OCB. Feb 2002.
- [20] M. Healy, T. Newe and E. Lewis. Analysis of Hardware Encryption Versus Software Encryption on Wireless Sensor Network Motes. In: S.C. Mukhopadhyay, G.S. Gupta (Eds.) *Smart Sensors and Sensing Technology*, Springer, 2008.
- [21] J. Huang, J. Seberry and W. Susilo. On the internal Structure of ALPHA-MAC. In: P.Q. Nguyen (Ed.) *VIETCRYPT 2006*, LNCS 4341, pp. 271-285, 2006.
- [22] C. Karlof, N. Sastry and D. Wagner. TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. In *SenSys'04*, November 3-5, 2004, Baltimore, Maryland, USA.
- [23] L. Knudsen, F. Mendel, C. Rechberger and S. Thomsen. Cryptanalysis of MDC-2. In *EUROCRYPT 2009*, LNCS 5479, pp. 106-120, 2009.
- [24] X. Lai and J. Massey. Hash Functions Based on Block Ciphers. In *EUROCRYPT'92*, LNCS 658, pp. 474-494, 1993.
- [25] T. Li, H. Wu, X. Wang and F. Bao. SenSec Design. *I²R Sensor Network Flagship Project (SNFP: security part): Technical Report-TR v1.0*, February, 2005.
- [26] M. Luk, G. Mezzour, A. Perrig and V. Gligor. MiniSec: A Secure Sensor Network Communication Architecture. In *IPSN'07*, April 25-27, 2007, Cambridge, Massachusetts, USA.
- [27] O. Özen, K. Varici, C. Tezcan and Ç. Kocair. Lightweight Block Ciphers Revisited: Cryptanalysis of Reduced Round PRESENT and HIGHT. In *ACISP 2009*, LNCS 5594, pp. 90-107, 2009.

- [28] C. Paar, A. Poschmann and M. Robshaw. New Designs in Lightweight Symmetric Encryption. In: P. Kitsos, Y. Zhang (Eds.) *RFID Security: Techniques, Protocols and System-on-Chip Design*. Springer, pp. 349-371, 2008.
- [29] A. Perrig, R. Szewczyk , V. Wen , D. Culler and J. D. Tygar. SPINS: security protocols for sensor networks. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pp.189-199, July 2001, Rome, Italy.
- [30] P. Rogaway. Authenticated-Encryption with Associated-Data. In *ACM CCS'02*, pp.98-107, ACM, 2002.
- [31] P. Rogaway, M. Bellare and J. Black. OCB: A block-cipher mode of operation for efficient authenticated encryption. *ACM Transactions on Information and System Security (TISSEC)*, Volume 6, Issue 3, pp.365-403, August 2003.
- [32] M. Wang. Differential Cryptanalysis of Reduced-Round PRESENT. In *AFRICACRYPT 2008*, LNCS 5023, pp. 40-49, 2008.
- [33] W. Wang, X. Wang and G. Xu. Impossible Differential Cryptanalysis of PELICAN, MT-MAC-AES and PC-MAC-AES. Available at <http://eprint.iacr.org/2009/005>.
- [34] G. Z. Yang (Ed). *Body Sensor Network*. Springer London, ISBN-10: 1-84628-272-1, 2003.