TuLP: A Family of Secure and Practical Message Authentication Codes for Body Sensor Networks

Zheng Gong[†], Pieter Hartel[†], and Svetla Nikova[†,‡]
[†]University of Twente
‡Katholieke Universiteit Leuven
and
Bo Zhu
Shanghai Jiaotong University

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 exploits the fact that the recommended MACs for WSNs, e.g., CBC-MAC (TinySec), OCB-MAC (MiniSec), and XCBC-MAC (SenSec), are not 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 to the existing schemes, our lightweight MACs are both time and resource efficient on hardware-constrained devices.

Categories and Subject Descriptors: C.2.0 [**Computer-Communication networks**]: Security and protection; D.4.6 [**Security and Protection**]: Authentication; D.4.8 [**Performance**]: Measurements; I.1.2 [**Algorithms**]: Analysis of algorithms

General Terms: Algorithms, Design, Security, Performance
Additional Key Words and Phrases: Authenticity, Message authentication code, Body sensor network, Low-resource implementation

Zheng Gong[†], Pieter Hartel[†], Svetla Nikova[†,‡] and Bo Zhu[§]
[†]University of Twente, Faculty of EWI, Enschede, Netherlands
‡Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC, Leuven, Belgium
{z.gong, pieter.hartel, s.nikova}@utwente.nl
§Shanghai Jiaotong University, Department of Computer Science and Engineering, China
zhubo03@gmail.com

A preliminary version of this paper is published in the proceedings of Indocrypt 2009 [Gong et al. 2009]. The first author acknowledges the financial support of SenterNovem for the ALwEN project, grant PNE07007. The last author is supported by NSFC (No.60573032, 60773092, 60803146), National "863" Program of China (No. 2009AA01Z418) and National "973" Program of China (No.2007CB311201).

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) vision [AAL 2008], body sensor networks (BSNs, also called wireless medical sensor networks) [Yang 2003] has attracted more attentions 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. Since BSNs are either worn or implanted by patients, highly resource-constrained nodes are widely chosen for achieving energy-efficiency and lightweight. Existing examples include CodeBlue [Malan et al. 2004], ALARM-NET [Wood et al. 2006] and DexterNet [Kuryloski et al. 2009]. Table I shows the hardware specifications of typical BSN nodes used in practice.

	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	$1.8 \sim 3.6v$	$2.7 \sim 3.3v$	$1.6 \sim 3.6v$
OS	TinyOS	TinyOS	MyriaCore

Table I. The specifications of typical BSN nodes.

In WSNs, people usually accept low-level security requirements as trade-offs of usability. However, BSNs are managed to monitor users' daily activities and health data, security and privacy 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 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 a 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) [Perrig et al. 2001], which consists of μTESLA (micro version of the Timed, Efficient, Streaming, Loss-tolerant Authentication) and

[1]Texas Instruments. http://focus.ti.com/lit/ds/symlink/msp430f149.pdf.
[2]Crossbow. http://www.xbow.com/products/Product_pdf_files/Wireless_pdf/MICAZ_Datasheet.pdf.
[3]ALwEN project. http://www.atmel.com/products/AVR/default_xmega.asp.

SNEP (Secure Network Encryption Protocol) for broadcasting messages. Following SPINS, many lightweight security architectures have been proposed for WSN, e.g., TinySec [Karlof et al. 2004], SenSec [Li et al. 2005] and MiniSec [Luk et al. 2007]. 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 [ISO9797-1 1999] and OCB-MAC [Rogaway et al. 2003] respectively, whilst SenSec uses a novel scheme called XCBC-MAC [Li et al. 2005]. All these MACs recommended for WSNs [Karlof et al. 2004; Li et al. 2005; Luk et al. 2007] 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 [FIPS198 2002]) are less competitive than block-cipher-based ones for highly constrained devices [Bogdanov et al. 2008]. It is widely recognized by the BSN research community that authentication in BSN protocols is usually for short messages in network processing [Yang 2003]. 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, first 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 power and RAM are highly constrained on a BSN node, a BSN-oriented MAC must take resource limitations into its design rationale as well.

Our Contributions. The contributions of this paper are three-fold. 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 [Daemen and Rijmen 2005a]. Thirdly, taking into account the requirements for BSN authenticity, we propose a tunable lightweight MAC based on the PRESENT block cipher [Bogdanov et al. 2007], which is named TuLP. The structure of TuLP is inspired by the generic construction ALRED [Daemen and Rijmen 2005a]. Moreover, a 128-bit variant TuLP-128 is also 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 costs, and also energy-efficient in the level of gate equivalents.

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 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. Let $\oplus$ denote the bit-wise exclusive-or (XOR) operation. A message $M = a||b$ denotes the concatenation of two strings a and b. $\mathcal{M}$ and $\mathcal{K}$ 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 [Daemen and Rijmen 2005a]. 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 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.

Figure 1 depicts the ALRED construction with $r = 1$ [Daemen and Rijmen 2005a]. 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 [Daemen and Rijmen 2005a] and Pelican [Daemen and Rijmen 2005b]. Recently, many papers exploited that ALPHA-MAC and Pelican might be threatened under the internal collisions [Huang et al. 2006], the side-channel attack [Biryukov et al. 2007] and the impossible differential analysis [Wang et al. 2009]. We note that all those cryptanalyses are based on the internal structures of ALPHA-MAC and Pelican, which do not endanger the security of ALRED.

PRESENT. At CHES 2007, Bogdanov *et al.* have proposed an ultra-lightweight block cipher which is named PRESENT [Bogdanov et al. 2007]. PRESENT is an example of a 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

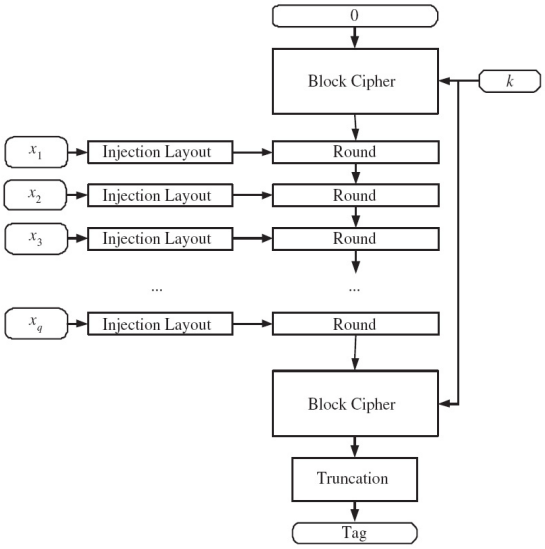

Fig. 1. The ALRED Construction with $r = 1$.

1570 and 1886 gate equivalents respectively [Bogdanov et al. 2007]. 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 2 [Bogdanov et al. 2007]. 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 hardware-efficient bit-oriented permutation is executed to provide diffusion.

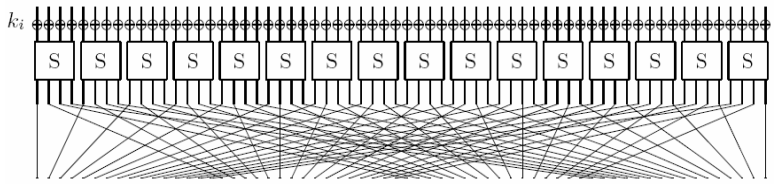

Fig. 2. Round function of PRESENT.

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 Bogdanov *et al.* [Bogdanov et al. 2007], including basic results of the differential and linear cryptanalyses, which can be summarized as follows.

THEOREM 2.1. *Any five-round differential characteristic of PRESENT has a minimum of 10 active S-boxes.*

THEOREM 2.2. *Let ϵ_{4R} be the maximal bias of a linear approximation of four rounds of PRESENT. Then $\epsilon_{4R} \leq 2^{-7}$.*

Moreover, Bogdanov *et al.* [Bogdanov et al. 2008] have proposed some low-energy block-cipher-based hash functions (e.g., single and double block length constructions of DM-PRESENT and H-PRESENT respectively). The comparison on the hardware performances [Bogdanov et al. 2008] 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.

Recently, many cryptanalysis results have been given on the PRESENT block cipher. Wang [Wang 2008] 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 [Albrecht and Cid 2009] 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 [Collard and Standaert 2009] described a statistical saturation attack against 24-round PRESENT. Besides the required plaintexts exceeds 2^{32}, the statistical saturation attack [Collard and Standaert 2009] still depends on the assumption that there exists an attack exploits 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.* [Özen et al. 2009] 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 [Wang 2008].

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 [Karlof et al. 2004], SenSec [Li et al. 2005], and MiniSec [Luk et al. 2007]. 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 [Karlof et al. 2004; Li et al. 2005; Luk et al. 2007].

CBC-MAC. In TinySec [Karlof et al. 2004], Karlof *et al.* suggest to use CBC-MAC [ISO9797-1 1999] 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, the one-key CBC-MAC construction [Bellare et al. 2000] is not secure for arbitrary length messages, which allows adversaries can forge a tag for certain messages. To preserve the provable security for arbitrary length messages, a variant of CBC-MAC uses three different keys for the authentication [Black and Rogaway 2005].

Although the three-key CBC-MAC 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 to the one-key construction, the extra keys will impose a burden on key generation, distribution and storage. If the underlying key management is centralized, those extra costs can be removed by a central device with pre-computation. But in BSN applications, nodes might be added and removed from a settled BSN frequently for changing its functionality. If the key management is distributed and adaptive, which is a highly possible situation in BSNs, the generation and the distribution costs of extra keys are non-negligible. The burden 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 CMAC algorithm, which is submitted to NIST [NIST 2005] as a variation of CBC-MAC that Black and Rogaway proposed and analyzed [Black and Rogaway 2005]. Note that CMAC only uses a single key with pre-computation would remove most of burdens on key generation and distribution.

XCBC-MAC. The XCBC-MAC algorithm, which has been proposed by Li *et al.*, is part of the authenticated encryption mode for SenSec [Li et al. 2005]. Let k_A and k_E be the authentication key and the encryption key, respectively. Let message $M = m_1||m_2||...||m_t$. In general, the XCBC-MAC algorithm can be viewed as a variant of the two-key CBC mode. Figure 3 depicts the construction of XCBC-MAC.

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 attack can be described in the following steps:

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

(2) Next, $\mathcal{A}$ requests the encryptions on the two different blocks $E_{k_E}(\text{IV}) \oplus m_1$ and $E_{k_E}(\text{IV}) \oplus m_1'$ in the XCBC mode. The ciphers will be $E_{k_E}(m_1) \oplus \text{IV}$ and $E_{k_E}(m_1') \oplus \text{IV}$. $\mathcal{A}$ 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 $M_1 = E_{k_E}(\text{IV}) \oplus m_1||E_{k_E}(m_1)||0||M'$ and $M_2 =$

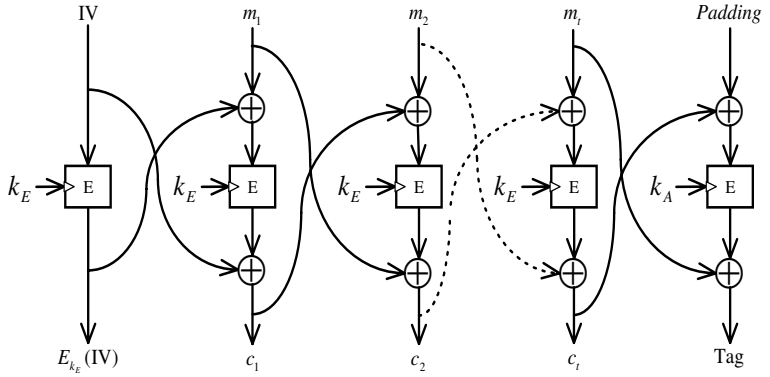

Fig. 3. The XCBC algorithm proposed in SenSec.

$E_{k_E}(\mathrm{IV}) \oplus m_1' || E_{k_E}(m_1') || 0 || M'.$

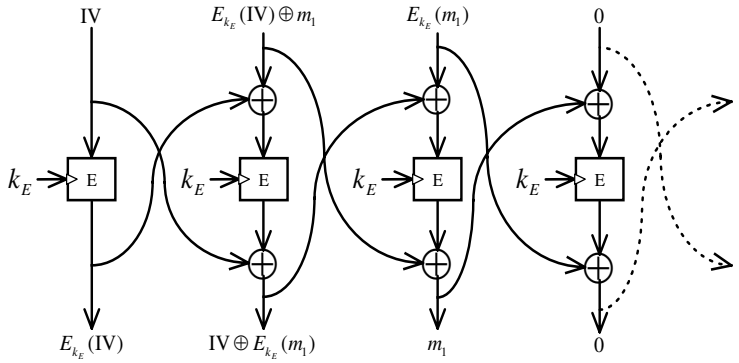

Fig. 4. An existential forgery under XCBC-MAC.

An illustration of our attack is depicted in Figure 4. It is straightforward that two different prefixes $E_{k_E}(\mathrm{IV}) \oplus m_1 || E_{k_E}(m_1) || 0$ and $E_{k_E}(\mathrm{IV}) \oplus m_1' || E_{k_E}(m_1') || 0$ 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 under the XCBC-MAC. 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.

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 impractical for an IV to be distributed just for one-time usage. 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 can not existentially forge a valid CBC-MAC value on a different IV or message. Due to the above reasons, the XCBC-MAC algorithm proposed in SenSec [Li et al. 2005] is insecure under the chosen message attack and should be abandoned in any circumstance of sensor network authentication.

OCB-MAC. In MiniSec [Luk et al. 2007], Luk *et al.* suggest using the OCB mode [Rogaway et al. 2003], which is an efficient authenticated encryption scheme, as the MAC function for message authenticity and integrity. Unlike other MAC candidates, OCB is a patented algorithm. The patented OCB raises two issues for its practical implementation, which have been emphasized by Ferguson [Ferguson 2002]. First, it might cause the intellectual property problem associated with using a patented algorithm in a product. On the other hand, less cryptanalysis has been given on OCB except the security proof from the original authors [Rogaway et al. 2003]. It is widely accepted in the cryptanalyst community that spending time on a patented algorithm might only be helpful to the patent-holders for selling their licenses. Moreover, Ferguson [Ferguson 2002] also described a collision attack on OCB with arbitrary length messages. To keep the authenticity of OCB, one has to limit the amount of data that the MAC algorithm processes. Although OCB is attractive as an efficient authenticated-encryption scheme, the above reasons cast doubt on using OCB for BSN applications.

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 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 [Black et al. 1999]. The second is to define a new mode of operation for existing primitives. Such as variants of encryption modes of block ciphers: CBC-MAC [ISO9797-1 1999] and OCB-MAC [Rogaway et al. 2003]; Or variants mode of hash functions: HMAC/NMAC [Bellare et al. 1996; FIPS198 2002]. 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 [Daemen and Rijmen 2005a].

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 [NESSIE 2003] (Pentium III/Linux Platform), such that the message processing time is measured in cycles/byte, while the keysetup 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 [Paar et al. 2008]. By following the same method [Bogdanov et al. 2008; Feldhofer and Rechberger 2006], 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 [Feldhofer and Rechberger 2006], the area for AES-128 encryption is estimated to be 3400 GE, as well as 64-bit storing and exclusive-or circuits require 512 GE and 170 GE, respectively.

	CBC-MAC [ISO9797-1 1999]	OCB-MAC [Rogaway et al. 2003]	ALPHA-MAC [Daemen and Rijmen 2005a]	HMAC (SHA-1) [FIPS198 2002]
Design method	cipher mode	cipher mode	AES components	hash mode
Keysetup	616	644	1032	1346
Finalization	1440	1444	416	3351
Message processing	26	30	10.6	15
Area in GE (estimate)	4764	6812	4424	8120

Table II. The comparison of some practical MAC functions.

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 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 II shows that ALPHA-MAC advances 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, 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 [Daemen and Rijmen 2005a] and Pelican [Daemen and Rijmen 2005b]. Next, the security of our schemes will be analyzed. Finally, the performance of our lightweight schemes will be given. Compared to the results in Table II, 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 [Bogdanov et al. 2007], 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 $|k| \leq 80$ 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 $m_1, m_2, \cdots, m_t, t = \frac{|pad(M)|}{64}$, 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, then obtain $s_0 = E_k(\text{IV})$ as the initial state.

(3) **Compression.** For each message block m_i where $i \in \{1, 2, \cdots, t\}$, 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 times ρ on the state s_{i-1}, such that

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

(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.

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

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 collsion. 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 [ISO/IEC10118-2 2000] and the padding rule is identical to TuLP.

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

(2) **Initialization.** Divide the (padded) authentication key k into two 80-bit key $k_l||k_r$. 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

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

(3) **Compression.** For each message block m_i where $i \in \{1, 2, \cdots, t\}$, 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 $\mathrm{LSB}^{32}(.)$) 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.

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

(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.

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

Figure 5 and 6 depict the high-level algorithms of TuLP and TuLP-128, respectively. Referring to the security issues of ALPHA-MAC and Pelican [Biryukov et al. 2007; Bogdanov et al. 2008; Wang et al. 2009], the advantages of our schemes are as follows.

—In ALPHA-MAC [Daemen and Rijmen 2005a], 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.* [Biryukov et al. 2007] 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 [Biryukov et al. 2007] on TuLP and TuLP-128.

—Like in Pelican [Daemen and Rijmen 2005b], 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 [Black et al. 2002].

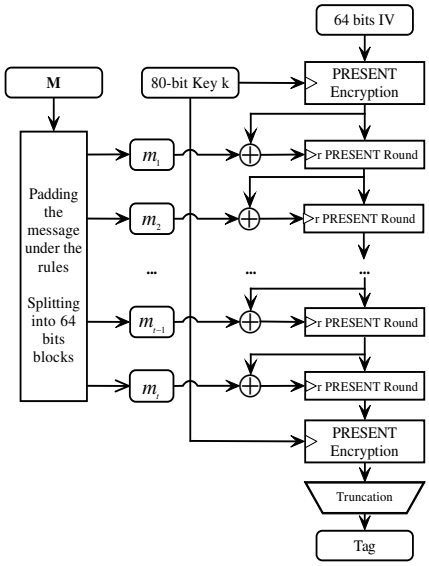

Fig. 5. The illustration of TuLP.

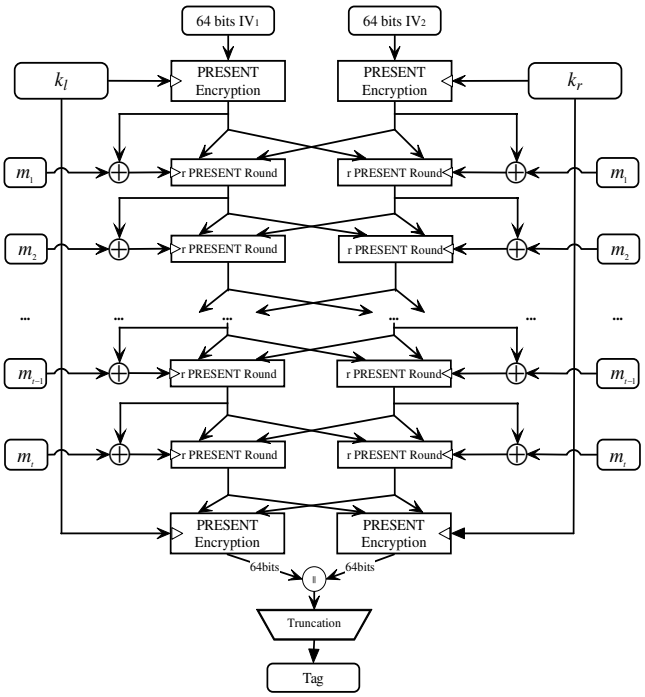

Fig. 6. The illustration of TuLP-128.

—The bitwise lengths of 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.

—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 full-round PRESENT, authentication and encryption can use the same secret key.

—TuLP is designed for rapid message processing. The computational costs of the message processing are 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 require $\frac{2 \cdot r}{31}$ 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 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 TuLP-128, whilst IV = IV_1 = `0123456789ABCDEF` and IV_2 = `FEDCBA9876543210`. The test vectors of TuLP and TuLP-128 are provided in Appendix.

—Same to 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 inherent its resource-efficient property.

5.2 Security Analysis

Based on the provability results of the ALRED construction in [Daemen and Rijmen 2005a], it is straightforward to derive similar results on TuLP and TuLP-128. 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 analyze TuLP when internal collisions are considered. Finally, a similar security analysis is given on TuLP-128.

THEOREM 5.1. *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 $i \in \{1, 2, \cdots, t\}$. 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 $tag_i = \text{Trunc}(E_k(s_i))$.
(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. □

Similar to Theorem 5.1, the provability of TuLP can be extended to the existential forgery attack and the fixed point attack as follows.

LEMMA 5.2. *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 $i \in \{1, 2, \cdots, t\}$. 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 $tag_i = \text{Trunc}(E_k(s_i))$.
(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 $E_k(\text{IV})$. So the lemma follows. □

LEMMA 5.3. *Any existential forgery attack on TuLP, requiring t (adaptively) chosen messages for a fixed point $\{(m, s)|E_{m\oplus s}(s) = s, m \in \mathcal{M}, s \in \mathcal{K}\}$, 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 $i \in \{1, 2, \cdots, t\}$. 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 $fp = s_i$ such that $E_k(s_i) = s_i$.

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

Now we analyze the security with respect to internal collisions.

LEMMA 5.4. *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 $i \in \{1, 2, \cdots, t\}$. Then we derive a collision attack on the r PRESENT round functions 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 (i.e., the r PRESENT round functions).
(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 an existential forgery tag', 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 $a, b \in \{1, 2, \cdots, t\}$.

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

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 5.5. *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.* [Bogdanov et al. 2007], 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 $(2^{-2})^{r*10/5} = 2^{-64}$ 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 the maximum bias of a linear approximation $(2^{-7})^{r/4} = 2^{-28}$.
(2) This linear cryptanalysis would require the memory complexity of about $1/(2^{-28})^2 = 2^{56}$ known plaintext/ciphertext pairs.

So the theorem follows. □

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 5.6. *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 $i \in \{1, 2, \cdots, t\}$. Let χ be the compression function of TuLP. $\mathrm{MSB}^{32}(\cdot)$ and $\mathrm{LSB}^{32}(\cdot)$ 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 $s_{l,0} = E_k(\mathrm{IV}_1)$ and $s_{r,0} = E_k(\mathrm{IV}_2)$.
(2) For $i = 1$ to t, compute the left state $s_{l,i} = \chi(\mathrm{MSB}^{32}(s_{l,i})||\mathrm{MSB}^{32}(s_{r,i}), m_i)$ and the right state $s_{r,i} = \chi(\mathrm{LSB}^{32}(s_{l,i})||\mathrm{LSB}^{32}(s_{r,i}), m_i)$.
(3) For $i = 1$ to t, request $tag_i = \mathrm{Trunc}(E_k(s_{l,i})||E_k(s_{r,i}))$.
(4) Submit t tag values to $\mathcal{A}$ to obtain an existential forgery tag', tag' 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 $E_k(\mathrm{IV}_1)$ and $E_k(\mathrm{IV}_2)$. So the theorem follows. □

Similar to Theorem 5.6, it is easy to obtain the following lemmas on TuLP-128.

LEMMA 5.7. *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 $i \in \{1, 2, \cdots, t\}$. Then we derive a ciphertext guessing attack on the PRESENT block cipher as follows.

(1) Request the initial left and right states $s_{l,0} = E_k(\mathrm{IV}_1)$ and $s_{r,0} = E_k(\mathrm{IV}_2)$.
(2) For $i = 1$ to t, compute the left state $s_{l,i} = \chi(\mathrm{MSB}^{32}(s_{l,i})||\mathrm{MSB}^{32}(s_{r,i}), m_i)$ and the right state $s_{r,i} = \chi(\mathrm{LSB}^{32}(s_{l,i})||\mathrm{LSB}^{32}(s_{r,i}), m_i)$.
(3) For $i = 1$ to t, request $tag_i = \mathrm{Trunc}(E_k(s_{l,i})||E_k(s_{r,i}))$.
(4) Submit t tag values to $\mathcal{A}$ to obtain an existential forgery tag', tag' should also be a valid ciphertext on the message m'.

The above attack needs t chosen messages except $E_k(\mathrm{IV}_1)$ and $E_k(\mathrm{IV}_2)$. So the lemma follows. □

LEMMA 5.8. *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.*

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 $i \in \{1, 2, \cdots, t\}$. 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 $s_{l,0} = E_k(\mathrm{IV}_1)$ and $s_{r,0} = E_k(\mathrm{IV}_2)$.

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

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

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

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 $E_k(\mathrm{IV}_1)$ and $E_k(\mathrm{IV}_2)$. So the lemma follows. □

By using multi-collisions, Knudsen *et al.* [Knudsen et al. 2009] 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 [Lai and Massey 1993] 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 5.9. *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.* [Black et al. 2002], 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 [Knudsen et al. 2009], 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 does not achieve the ideal upper bounds of collision and preimage resistances, the MDC-2 like structure in TuLP-128 still yields many practical advantages. For example, symmetric left and right pipes can minimize the

area in hardware, or the memory usage in software implementation. And the simple permutation layer between left and right states saves redundant logical gates. 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 1K bytes look-up table on MICAz nodes. From our performance tuning, we find that the bit permutation of PRESENT is costly in software implementation. Compared to the best known result of AES-128 software implementation on MICAz nodes [Healy et al. 2008], 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 [Bogdanov et al. 2008], our optimized implementation shows that PRESENT is still practical in software.

	Software (MICAz)			Hardware [Bogdanov et al. 2008]		
Encryption	RAM (byte)	ROM (byte)	Processing speed	Logic process	Cycles per block	Area
AES-128 [Healy et al. 2008]	1915	12720	1.46ms / 16Bytes	0.35μm	1032	3400 GE
PRESENT-80	1040	1926	1.82ms / 8Bytes	0.18μm	32	1570 GE

Table III. The comparison of AES and PRESENT implementations.

As a point of comparison, we select DM-PRESENT [Bogdanov et al. 2008], which is derived from the Davies-Meyer construction and the PRESENT with an 80-bit key, as the underlying hash function for HMAC [FIPS198 2002]. 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 0.18μm 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 block processing speed (in milliseconds) are averaged by iterating 100 times experiments with/without the optimization in the keysetup[1].

If we choose r=16 in the compression of TuLP, Table IV 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

[1]The performance test codes can be found at `http://eprints.eemcs.utwente.nl/15369/`

	Key length (bit)	Block size (bit)	RAM / ROM (byte)	Area in GE (estimate)	Block Processing Speed (ms)
TuLP	80	64	1048 / 3302	2252	4.46 / 6.63
TuLP-128	160	128	1056 / 3718	2764	8.91 / 13.24
OCB-MAC (PRESENT)	80	64	1048 / 3362	3276	6.56
CBC-MAC (PRESENT)	80	64	1040 / 2970	2252	6.51
HMAC (DM-PRESENT)	80	64	1056 / 3484	2213 [Bogdanov et al. 2008]	10.90
ALPHA-MAC (AES)	128	128	2088 / 5342	4424	3.92
OCB-MAC (AES)	128	128	2104 / 6144	6812	4.07
CBC-MAC (AES)	128	128	2088 / 5320	4764	3.96

Table IV. The comparison amongst some PRESENT-based MAC functions.

in DM-PRESENT) in the same manner as secret keys [FIPS198 2002]. 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 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 software speeds of TuLP and TuLP-128 are slower than the MAC constructions based on AES. According to the hardware performances of AES and PRESENT [Bogdanov et al. 2008], it is straightforward that TuLP and TuLP-128 are efficient than AES-based MACs in low-resource implementations. Note that both the hardware and the software implementation costs of TuLP and TuLP-128 are much smaller than 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 $(2^{-2})^{20*10/5} = 2^{-80}$ for a differential characteristic. And the maximal bias of a linear approximation $(2^{-7})^{20/4} = 2^{-35}$, which requires 2^{70} known plaintext/ciphertext.

7. CONCLUSION

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.

ACKNOWLEDGMENTS

We would like to thank Vincent Rijmen and Xuejia Lai for their helpful advice. And also thank many anonymous reviewers for their valuable comments.

REFERENCES

AAL. 2008. European union. the ambient assisted living (aal) joint programme. http://www.aal-europe.eu/about-aal.

Albrecht, M. and Cid, C. 2009. Algebraic techniques in differential cryptanalysis. In *Fast Software Encryption - FSE 2009*, O. Dunkelman, Ed. Vol. LNCS 5665. Springer, 193–208.

Bellare, M., Canetti, R., and Krawczyk, H. 1996. Keying hash functions for message authentication. In *Advances in Cryptology-Crypto'96*, N. Koblitz, Ed. Vol. LNCS 1109. Springer-Verlag, 1–15.

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

Biryukov, A., Bogdanov, A., Khovratovich, D., and Kasper, T. 2007. Collision attacks on aes-based mac: Alpha-mac. In *Cryptographic Hardware and Embedded Systems - CHES 2007*, P. Paillier and I. Verbauwhede, Eds. Vol. LNCS 4727. Springer-Verlag, 166–180.

Black, J., Halevi, S., Krawczyk, H., Krovetz, T., and Rogaway, P. 1999. Umac: Fast and secure massage authentication. In *Advances in Cryptology-Crypto'99*, M. Wiener, Ed. Vol. LNCS 1666. Springer-Verlag, 216–233.

Black, J. and Rogaway, P. 2005. Cbc macs for arbitrary-length messages: the three-key constructions. *Journal of Cryptology 18(2)*, 111–131.

Black, J., Rogaway, P., and Shrimpton, T. 2002. Black-box analysis of the block-cipher-based hash-function constructions from pgv. In *Advances in Cryptology-Crypto 2002*, M. Yung, Ed. Vol. LNCS 2442. Springer, 320–335.

Bogdanov, A., Knudsen, L., Leander, G., Paar, C., Poschmann, A., Robshaw, M., Seurin, Y., and Vikkelsoe, C. 2007. Present: An ultra-lightweight block cipher. In *Cryptographic Hardware and Embedded Systems - CHES 2007*, P. Paillier and I. Verbauwhede, Eds. Vol. LNCS 4727. Springer Heidelberg, 450–466.

Bogdanov, A., Leander, G., Paar, C., Poschmann, A., Robshaw, M., and Seurin, Y. 2008. Hash functions and rfid tags: Mind the gap. In *Cryptographic Hardware and Embedded Systems - CHES 2008*, R. Safavi-naini, Ed. Vol. LNCS 5154. Springer, 283–299.

Collard, B. and Standaert, F.-X. 2009. A statistical saturation attack against the block cipher present. In *CT-RSA 2009*, M. Fischlin, Ed. Vol. LNCS 5473. 195–210.

Daemen, J. and Rijmen, V. 2005a. A new mac construction alred and a specific instance alpha-mac. In *FSE 2005*, H. Gilbert and H. Handschuh, Eds. Vol. LNCS 3557. Springer, 1–17.

Daemen, J. and Rijmen, V. 2005b. The pelican mac function. Unpublished. Available at http://eprint.iacr.org/2005/088.

Feldhofer, M. and Rechberger, C. 2006. A case against currently used hash functions in rfid protocols. In *OTM Workshops 2006*, R. Meersman, Z. Tari, and P. Herrero, Eds. Vol. LNCS 4277. Springer, 372–381.

Ferguson, N. 2002. Collision attacks on ocb. Preprint. Available at http://csrc.nist.gov.

FIPS198. 2002. *Federal Information Processing Standard 198, The Keyed-Hash Message Authentication Code (HMAC)*. NIST, U.S. Department of Commerce.

Gong, Z., Hartel, P. H., Nikova, S., and Zhu, B. 2009. Towards secure and practical macs for body sensor networks. In *Progress in Cryptology - INDOCRYPT 2009*, B. K. Roy and N. Sendrier, Eds. Lecture Notes in Computer Science, vol. 5922. Springer, 182–198.

Healy, M., Newe, T., and Lewis, E. 2008. Analysis of hardware encryption versus software encryption on wireless sensor network motes. In *Smart Sensors and Sensing Technology 2008*, S. Mukhopadhyay and G. Gupta, Eds. Vol. LNEE 20. Springer, 3–14.

Huang, J., Seberry, J., and Susilo, W. 2006. On the internal structure of alpha-mac. In *VIETCRYPT 2006*, P. Nguyen, Ed. Vol. LNCS 4341. Springer, 271–285.

ISO9797-1. 1999. *Information technology - Security Techniques - Message Authentication Codes (MACs) - Part 1: Mechanisms using a block cipher*. ISO.

ISO/IEC10118-2. 2000. *Information technology - Security techniques - Hash-functions - Part 2: Hash-functions using an n-bit block cipher algorithm*. ISO.

Karlof, C., Sastry, N., and Wagner, D. 2004. Tinysec: A link layer security architecture for wireless sensor networks. In *SenSys'04: Proceedings of the 2nd international conference on Embedded networked sensor systems*. ACM Press, 162–175.

Knudsen, L., Mendel, F., Rechberger, C., and Thomsen, S. 2009. Cryptanalysis of mdc-2. In *Advances in Cryptology - EUROCRYPT 2009*, A. Joux, Ed. Vol. LNCS 5479. Springer, 106–120.

Kuryloski, P., Giani, A., Giannantonio, R., Gilani, K., Gravina, R., Seppa, V.-P., Seto, E., Shia, V., Wang, C., Yan, P., Yang, A. Y., Hyttinen, J., Sastry, S., Wicker, S., and Bajcsy, R. 2009. Dexternet: An open platform for heterogeneous body sensor networks and its applications. In *BSN '09: Proceedings of the 2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*. IEEE Computer Society, Washington, DC, USA, 92–97.

Lai, X. and Massey, J. 1993. Hash functions based on block ciphers. In *Advances in Cryptology - EUROCRYPT 1992*, R. Rueppel, Ed. Vol. LNCS 658. Springer, 474–494.

Li, T., Wu, H., Wang, X., and Bao, F. 2005. Sensec design. Tech. rep., I^2R Sensor Network Flagship Project (SNFP: security part): Technical Report-TR v1.0. February.

Luk, M., Mezzour, G., Perrig, A., and Gligor, V. 2007. Minisec: A secure sensor network communication architecture. In *Proceedings of IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. ACM Press, 479–488.

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

NESSIE. 2003. *Performance of optimized implementations of the NESSIE primitives*, v2.0 ed. The NESSIE Consortium.

NIST. May 2005. *Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*, Special Publication 800-38B ed.

Özen, O., Varici, K., Tezcan, C., and Ç. Kocair. 2009. Lightweight block ciphers revisited: Cryptanalysis of reduced round present and hight. In *ACISP 2009*, C. Boyd and J. Nieto, Eds. Vol. LNCS 5594. Springer, 90–107.

Paar, C., Poschmann, A., and Robshaw, M. 2008. New designs in lightweight symmetric encryption. In *RFID Security: Techniques, Protocols and System-on-Chip Design*, P. Kitsos and Y. Zhang, Eds. Springer, 349–371.

Perrig, A., Szewczyk, R., Wen, V., Culler, D., and Tygar, J. 2001. Spins: security protocols for sensor networks. In *Proceedings of the 7th annual international conference on Mobile computing and networking - MOBICOM 2001*. ACM Press, 189–199.

Rogaway, P., Bellare, M., and Black, J. 2003. Ocb: A block-cipher mode of operation for efficient authenticated encryption. *ACM Transactions on Information and System Security (TISSEC) volume 6, issue 3*, 365–403.

Wang, M. 2008. Differential cryptanalysis of reduced-round present. In *Progress in Cryptology - AFRICACRYPT 2008*, S. Vaudenay, Ed. Vol. LNCS 5023. Springer, 40–49.

Wang, W., Wang, X., and Xu, G. 2009. Impossible differential cryptanalysis of pelican, mt-mac-aes and pc-mac-aes. Preprint. Available at http://eprint.iacr.org/2009/005.

Wood, A., Virone, G., Doan, T., Cao, Q., Selavo, L., Wu, Y., Fang, L., He, Z., Lin, S., and Stankovic, J. 2006. Alarm-net: Wireless sensor networks for assisted-living and residential monitoring. Tech. rep., Department of Computer Science, University of Virginia,.

Yang, G. Z. 2003. *Body Sensor Network*. Springer London.

APPENDIX

For correctness examination, here we provide the test vectors of TuLP and TuLP-128 in hexadecimal notation. Let IV = IV_1 = `0123456789ABCDEF` and IV_2 = `FEDCBA9876543210`.

Key	Message	Tag (64 bits)
0000 0000 0000 0000 0000	FFFF FFFF FFFF FFFF	5C35 7515 9F31 9269
FFFF FFFF FFFF FFFF FFFF	0000 0000 0000 0000	503C 691F EDA0 C99E
1234 5678 90AB CDEF FFFF	FFFF FFFF FFFF FFFF	1205 8DE6 FAAE B3A3
0000 0000 0000 0000 0000	1234 5678 90AB CDEF	752D EE6C C7E7 78B7

Table V. Test vectors for TuLP.

Key	Message	Tag (128 bits)
k_l =0000 0000 0000 0000 0000 k_r =FFFF FFFF FFFF FFFF FFFF	FFFF FFFF FFFF FFFF	B91F 9B27 23EC 5886 26AC CD6F 22C7 85B7
k_l =FFFF FFFF FFFF FFFF FFFF k_r =0000 0000 0000 0000 0000	0000 0000 0000 0000	D3FE 5CF2 741C 7370 9C14 A62E D92F 034D
k_l =1234 5678 90AB CDEF FFFF k_r =0000 0000 0000 0000 0000	FFFF FFFF FFFF FFFF	0F4E 2B7D 7DE2 20A9 4C41 9A79 5DD3 2DBA
k_l =0000 0000 0000 0000 0000 k_r =1234 5678 90AB CDEF FFFF	1234 5678 90AB CDEF	67BB 918E 44E7 E816 5B33 0693 DAA4 B68B

Table VI. Test vectors for TuLP-128.