A Study on the Applicability of the Lesamnta-LW Lightweight Hash Function to TPMS*

Yuhei Watanabe[1,3], Hideki Yamamoto[1,2], Hirotaka Yoshida[1,3]

[1] SEI-AIST Cyber Security Cooperative Research Laboratory
[2] Sumitomo Electric Industries, Ltd. (SEI)
[3] National Institute of Advanced Industrial Science and Technology (AIST)

Abstract. The Tire Pressure Monitoring System (TPMS) is used to monitor the pressure of the tires and to inform the driver of it. This equipment is mandatory for vehicles in US and EU. To ensure the security of TPMS, it is important to reduce the cost of the cryptographic mechanisms implemented in resourced-constrained devices. To address this problem, previous work has proposed countermeasures employing lightweight block ciphers such as PRESENT, SPECK, or KATAN. However, it is not clear to us that any of these works have addressed the issues of software optimization that considers TPMS-packet protection as well as session key updates for architectures consisting of the vehicle TPMS ECU and four low-cost TPM sensors equipped with the tires. In this paper, we propose to application of the ISO/IEC 29192-5 lightweight hash function Lesamnta-LW to address this issue. Our approach is to apply the known method of converting Lesamnta-LW to multiple independent pseudo-random functions (PRFs) in TPMS. In our case, we generate five PRFs this way and then use one PRF for MAC-generation and four for key derivation. Although we follow the NIST SP 800-108 framework of converting PRFs to key derivation functions, we confirm the significant advantage of Lesamnta-LW-based PRFs over HMAC-SHA-256 by evaluating the performance on AVR 8-bit micro-controllers, on which we consider simulating TPMS sensors. We expect that our method to achieve multiple-purposes with a single cryptographic primitive will help to reduce the total implementation cost required for TPMS security.

Keywords: TPMS, 8-bit micro-controllers, FELICS, Lesamnta-LW, PRF, KDF

1 Introduction

The Tire Pressure Monitoring System (TPMS) is employed in modern vehicles to monitor the status of tire pressures and establish communication between the vehicles and the sensors equipped within the tires. It is typical that in TPMS the sensors spend more than 90% of their time in power-down mode [2]. Currently, it is mandatory for many new vehicles to adopt the TPMS in the US and EU.

In 2010, eavesdropping and spoofing attacks on TPMS were reported [3]. Therefore, the security of TPMS has become increasingly important, although there are harsh requirements in the automotive industry, saying that implementation costs are severely constrained. Costs are evaluated from many metrics including power consumption, energy consumption, required time, and circuit-size. Therefore, it is not always clear whether power consumption or energy consumption requirements can be met or not.

There have been a number of studies on cryptographic protocols for ensuring the confidentiality and/or integrity of communication packets as well as cryptographic or

*This paper has been presented at escarAisa 2018 [1] without transferring copyright.

non-cryptographic key management protocols for TPMS. For instance, a packet protection protocol using the hardware-oriented block cipher KATAN32 [4] has been proposed, and the FPGA implementation of this protocol has been evaluated in terms of delay and power [5]. The authors proposed a session-key update mechanism but did not apply a cryptographic mechanism. In [2], a lightweight protocol based on lightweight hardware-oriented block ciphers such as SPECK [6] and PRESENT [7], was proposed, and this achieved energy consumption some two orders of magnitude lower than that reported in [5].

According to the authors this is because the dedicated TPMS sensors form the deployment platform as well as the lightweight implementation properties of SPECK. The applicability of their protocol on Infineon SP 37 sensors implementated on an 8-bit micro-controller. In [8], the authors propose a packet protection protocol using PRESENT and show that their implementations are suitable for the NXP FXTH87 TPMS sensor in terms of RAM, ROM, and computation time.

We argue that the important problem is that, it appears to us that none of the above works [5, 2, 8] explores optimized embedded software solutions for session key generation (or key derivation), as well as communication packet protection, that are suitable for the TPMS architectures. These architectures consist of the vehicle TPMS electronic control unit (ECU) and four TPMS sensors that are severely constrained, especially in terms of RAM for cryptographic implementations.

Our Contribution In this paper, we propose the application of the ISO/IEC 29192-5 embedded software-oriented hash function Lesamnta-LW [9] to TPMS-communication protection and key derivation for the purpose of minimizing the RAM size of cryptographic implementations. Considering the long life-cycle of the vehicles, our preference is given to the 128-bit security that Lesamnta-LW is expected to offer over 80-bit security that many lightweight cryptographic primitives offer to meet very severe implementation requirements. We propose to use the known multiple pseudo-random functions (PRFs) based on Lesamnta-LW [10] for MAC-generation as well as key derivation in the NIST SP800-108-specified key derivation function (KDF) framework [11]. We first clarify the implementation specification for TPMS use and then confirm the suitability of the proposed methods for the TPMS architecture by evaluating its performance on an AVR 8-bit micro-controller which we consider a vehicle implementation environment. Our proposed KDF property that the session key sequences for the four tires are independent of each other.

Expected Impact on TPMS development In recent years, 8-bit micro-controllers such as Infineon SP 37 have received increasing interest from the automotive industry. From the viewpoint of a cryptographic protocol for TPMS, the implementation cost constraints are severe; in particular, the RAM cost on an 8-bit micro-controller for vehicle ECUs is considered critical. We expect that our achievement of 128-bit security on Lesamnta-LW-based PRFs will contribute to the development of TPMS in terms of cost-efficiency and the long life-cycle of vehicles.

We consider the problem of the key being compromised to be a serious issue. The key separation property that our KDF offers, helps to minimize the negative impact of this event.

Organization In Section 2, we discuss some preliminary work related to this paper. In Section 3, we identify the problems with previous protocols and then the proposed approach. In Section 4, we evaluate the performance of the Lesamnta-LW-based PRFs. Section 5 explores the limitations of our work. Section 6, presents our conclusion.

2 Preliminary

2.1 Overview of the Target TPMS

Referring to [3], we here give a brief introduction to TPMS. In TPMS, TPMS data transmissions typically use the 433/315 MHz bands. The tire pressure sensor contains an identifier (ID) that determines the origin of the packet and filters out packets sent by other vehicles. The TPMS sensors broadcast the pressure and temperature measurements associated with their identifiers in a periodical manner. The TPMS ECU/receiver receives the packets, filters out packets, and performs temperature compensation.

The above communication between the TPMS ECU and TPMS sensor is summarized in Fig. 1.

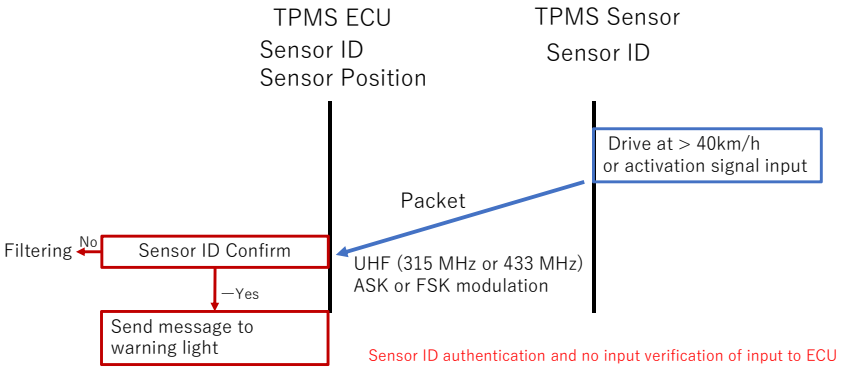

Figure 1: Overview of TPMS

2.2 Applications of Lesamnta-LW Lightweight Hash Function

2.2.1 Notations and Definitions

Following [10], we introduce notations and definitions for explaining specifications of cryptographic mechanisms. Let $\Sigma = \{0,1\}$. For any non-negative integer l, Σ^l is identified with the set of all Σ-sequences of length l. Σ^0 is the set of the empty sequence ε.

For x, the length of x is denoted by $|x|$. The concatenation of x_1 and x_2 is denoted by $x_1 \| x_2$.

2.2.2 The Hashing Mode of Lesamnta-LW and Its Variant with MDP

Lesamnta-LW is a Merkle-Damgård iterated hash function [9]. It is the plain Merkle-Damgård iteration of a block cipher E taking as input $n/2$-bit key and n-bit plaintext, where n is a positive even integer. The input of E from the top is its key input. $IV_0 \| IV_1 \in \Sigma^n$ is an initialization vector, where $|IV_0| = |IV_1| = n/2$. $M_1, M_2, \ldots, M_m$ are message blocks, where $M_i \in \Sigma^{n/2}$ for $i = 1, 2, \ldots, m$.

The variant of the hashing mode of Lesamnta-LW with the MDP domain extension [12] is introduced as follows. The MDP variant with a permutation π on $\Sigma^{n/2}$ is the function $J^{E,\pi}$, which is defined as follows: For $X_1, X_2, \ldots, X_x \in \Sigma^{n/2}$ and $Y_0 \in \Sigma^n$,

$$J^{E,\pi}(Y_0, X_1 \| X_2 \| \cdots \| X_x) = Y_x$$

such that

$$Y_i \leftarrow \begin{cases} E_{Y_{i-1,0}}(X_i \| Y_{i-1,1}) & (1 \le i \le x-1) \\ E_{Y_{i-1,0}}(X_i \| \pi(Y_{i-1,1})) & (i = x) \end{cases},$$

where $Y_j = Y_{j,0} \| Y_{j,1} \in \Sigma^n$ and $|Y_{j,0}| = n/2$ for $0 \leq j \leq x$. Note that π need not be a cryptographic primitive. Thus, the computational overhead of π can be very small.

2.2.3 Multiple PRFs based on Lesamnta-LW

Let $J_{IV}^{E,\pi} : \Sigma^{n/2} \times (\Sigma^w)^+ \to \Sigma^n$ be a keyed function such that $J_{IV}^{E,\pi}(K, X) = J^{E,\pi}(K \| IV, X)$, where $K \in \Sigma^{n/2}$, $IV \in \Sigma^{n-n/2}$ and $X \in (\Sigma^w)^+$. K is a secret key and IV is an initialization vector.

Suppose that $\Pi \cup \{id\}$ is pairwise everywhere distinct and that $\pi(IV) \neq \pi'(IV')$ for any $\pi, \pi' \in \Pi \cup \{id\}$ and $IV, IV' \in \mathcal{V}$ such that $(\pi, IV) \neq (\pi', IV')$.

Theorem 1 [9] states that $J_{IV}^{E,\pi}(K, X)$ and $J_{IV'}^{E,\pi'}(K, X)$ are two independent PRFs with a single key if E is a PRP. It is depicted in Fig. 2. In this way, one cryptographic primitive can be converted to produce multiple cryptographic functions with a small additional cost.

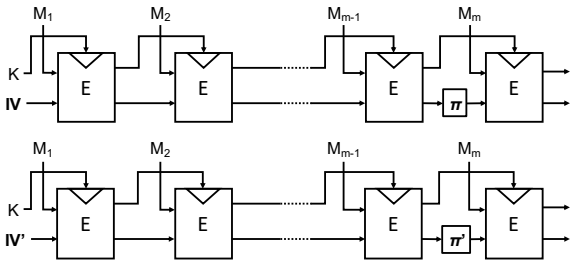

Figure 2: The multiple PRFs based on Lesamnta-LW [9]

In [9], π functions used in Lesamnta-LW-based PRFs are not specified. Hence, when applied to real-world systems, one has to specify them.

2.3 HMAC-SHA-256: Hash Function-based PRF

HMAC-SHA-256 [13] is a widely known PRF based on the SHA-256 [14] hash function for a general purpose. The overview of HMAC-SHA-256 is given in Fig. 3 where M is a message input, K is a secret key, H is SHA-256, `ipad` = `0x3636...36`, and `opad` = `0x5c5c...5c`.

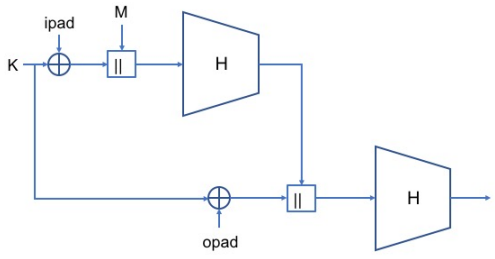

Figure 3: Overview of HMAC-SHA-256 PRF

3 The Problems and Our Approaches

3.1 The Problems

This section identifies several problems with previous protocols.

3.1.1 Security Properties of TPMS

The following aspects of cryptographic application to TPMS require clarification.

From the explanations given in the previous works, it is not systematically selected which data are protected for confidentially and which data are protected for integrity. It is necessary to protect IDs with regards to confidentiality and this is the case for ensuring location privacy.

The lightweight application of cryptographic primitives is derived by considering what is really needed to counter the targeted threats. More specifically, which data should be protected for which security perspective, namely confidentiality or integrity, is considered in a more optimized manner.

In the protocol proposed in [5], KATAN32 and CBC-MAC are used as the underlying primitive and mode of operation. However, KATAN32 only accepts small key and plaintext input lengths; hence, it is not clear to us whether the security strength of the proposed protocol is appropriate for **long-term security** over the lifecycles that vehicles typically (should) achieve.

PRESENT and SPECK also take small plaintext sizes because they have small block sizes. Therefore, it is not clear to us whether the security strength of the proposed protocol in [2] and [8] is appropriate for **long-term security** over the lifecycles that vehicles typically (should) achieve.

3.1.2 Implementation Cost for the TPMS Sensors

The cryptographic protocol for TPMS presented in [5] assumes that both the tire pressure sensors and the vehicle ECU have the hardware components for cryptography. Although hardware implementations of cryptography are effective in some cases, under the current circumstance of a tire pressure sensor, it is not reasonable to assume that a hardware-based cryptographic solution is the immediate choice.

TPMS faces severe cost constraints. For the vehicle ECU, it is expected that the cryptographic mechanisms are typically implemented in software on an 8-bit low-cost micro-controller. The relevant **computation cost** metrics are the amount of RAM consumption, which could be critical, the execution time, and the code size. As for applications to the TPMS sensors, implementations of cryptographic mechanisms must require a small amount of RAM.

In [5], the proposed protocol assumes that the cryptographic primitive and random number generator are employed in the TPMS sensors. Although the implementation of this protocol is evaluated, it is not clear to us whether the above **assumption is realistic**.

To reduce the implementation cost of TPMS-PRF using SHA-256, we investigate the applicability of lightweight cryptographic primitives. The known performance metrics of lightweight cryptographic primitives include the circuit size, power consumption, and latency for hardware, and the memory size and latency for software. Lightweight cryptographic primitives are typically designed for the target constraint device and are likely to be either hardware-oriented or software-oriented.

Regarding TPMS-PRF, we point out the software implementation requiring small amounts of RAM on the vehicle ECU is the critical factor to consider. Note that Regarding TPMS, the computation time is typically on the order of milliseconds (ms); therefore, this is not expected to be critical for cryptography.

In [2], a lightweight protocol has been proposed using lightweight block ciphers such as SPECK and PRESENT, and CBC-MAC to be used as the underlying ciphers and mode of operation respectively. The applicability of this protocol was studied using SP 37 sensors. The proposed protocol is similar to the protocol proposed in [5] and processing 96-bit data frames. However, they showed that implementation evaluation where the environment is SP 37 8-bit micro-controllers. The lightweight metrics of energy consumption and power

consumption for the TPMS sensors is the main focus of the evaluation experiments. Table 1 shows the performances of previous cryptographic protocols for TPMS.

Table 1: Key length and Performance of Previous Methods

Mechanism	Key Length (Bits)	RAM (Bytes)	ROM (Bytes)	Comp. Time (ms)	Evaluation Environment	Ref.
KATAN32-based Protocol*	80	N/A	N/A	36.6	Arduino Uno +RFM22	[5]
PRESENT-based Protocol	80	27	846	303.3	SP37	[2]
SPECK-based Protocol	128	20	865	29.1	SP37	[2]
PRESENT-based Enc-then-MAC	80 or 128 (Unspecified)	502	7,013	19.4	RL78	[8]

*: The authors' KATAN32-based implementation requires $1231uJ$ w.r.t power [5].

3.1.3 Key Management Required for Secure TPMS Communication

One of the key characteristics of the TPMS is that, unlike many IT systems, the sender (ECU) has to communicate with multiple (four) receivers in the form of the TPMS sensors.

The protocol in [5] assumed that the master key is installed in a secure environment, such as the car dealer stores. This suggests that the protocol assumes that individuals do not replace their own tires. The authors refer to the TESLA protocol [15], but claim that this protocol is **not** suitable for TPMS because there are only four receivers, even if the broadcast protocol is applied. The authors propose the session key establishment protocol and evaluate the FPGA implementations using KATAN block ciphers in terms of power and delay.

In [2], the assumption that the master key is installed at secure locations is removed, allowing tires to be changed at places other than secure garage. This protocol recommends using the standard technique specified in [16] for key establishment. However, this standard was developed for generic purposes rather than specific wireless systems such as TPMS.

In [8], the authors propose a protocol using PRESENT as underlying lightweight cipher and CMAC or Enc-then-MAC mode as the mode of operation. This protocol was evaluated by simulating four sensors. Consequently, the authors showed that their implementations are suitable for the NXP FXTH87 TPMS sensor. However, it is not clear to us how their protocol solves the key management issue.

The important key management problem is that none of the above works [5, 2, 8] appear to explore the solutions for session key generation or key derivation that are suitable for the TPMS architecture. That is, the functionalities of encryption/integrity have to be performed in software on the TPMS devices that are severely constrained in terms of RAM, all while addressing the issue of preventing the key from being compromised.

3.2 Our Approaches

This section explains how we propose to deal with the problems identified in the previous subsection.

3.2.1 Applying Lesamnta-LW Hash Function to TPMS Key Derivation

It would be a challenge that the key management problem is solved with a low cost. More specifically, we obtain different session key sequences from one master key without having many cryptographic primitives installed in the receiver. Different session key sequences, depending on the receiver, namely, TPMS sensors, would be better that same session key sequences from security perspectives.

We propose to apply to multiple PRFs based on Lesamnta-LW to the NIST framework of NIST SP 800-108 on key derivation using PRFs [11] that describes how to generate or **derivate the session keys** required for confidentially and/or integrity protection, which could be performed by means of KDF, as shown in Fig. 4. Here, for the 16-byte encryption key K_{enc} and the 16-byte authenticated K_{auth}, $K_{enc}||K_{auth}$ is generated as the output of the 32-byte output of each call to Lesamnta-LW-PRF/π_i. In this way, we achieve **key separation**, meaning that the compromise of some keys must not degrade the security of any of the other keys that are obtained from the output of the same KDF execution.

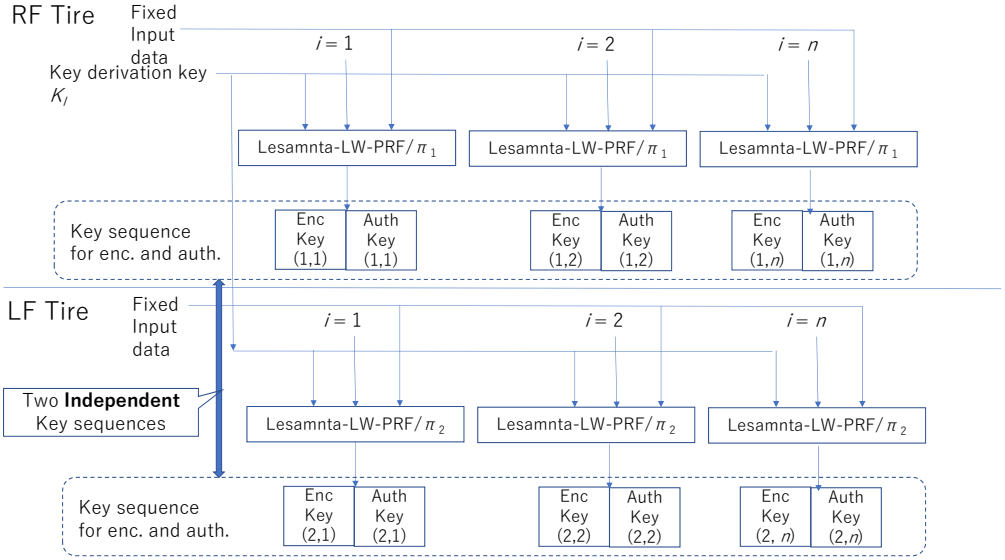

Figure 4: Application of Lesamnta-LW based PRFs to NIST SP 800-108 KDF in counter mode for TPMS

We show how to manage the session keys for the encryption and authentication in four tires. We assume that one master key is installed in the ECU and each TPMS sensor. We use PRFs based on Lesamnta-LW which can construct independent PRFs by changing the permutation π to obtain four independent session key sequences from a master key. Let K_{LF}, K_{RF}, K_{LB}, and K_{RB} be session keys for each tire. Let C_{LF}, C_{RF}, C_{LB}, and C_{RB} be fixed input data for each tire. The ECU send each fixed input data to each sensor. Let KDF_{LF}, KDF_{RF}, KDF_{LB}, and KDF_{RB} be KDFs constructed by independent PRFs for each tire. The session keys are given by $K_{LF} = KDF_{LF}(K_m, C_{LF}, i)$, $K_{RF} = KDF_{RF}(K_m, C_{RF}, i)$, $K_{LB} = KDF_{LB}(K_m, C_{LB}, i)$, and $K_{RB} = KDF_{RB}(K_m, C_{RB}, i)$. When a sensor update the session key, a sensor update counter i and calculate value for the session key by PRF. The ECU can obtain the session keys by using same process. We show the overview of above process in Fig. 5.

We discuss about the scenario of the session key compromise. By achieving **key separation**, even if one session key is compromised, there are no degradation of security in other three session keys. Since a session key is derived from the master key, fixed input value, and counter by using Lesamnta-LW-based PRFs, we consider that it is difficult for

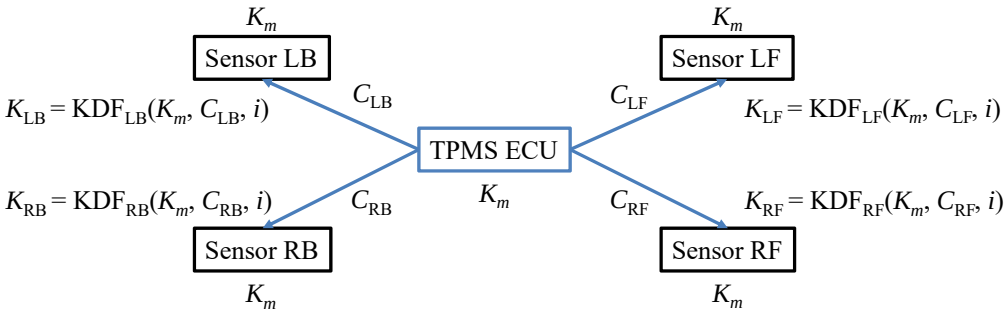

Figure 5: Scenario of Session Key Management

attackers to derive past session keys and future session keys from the compromised session key. Therefore, if a past TPMS message uses different session keys, our approach can prevent attack with the compromised session key.

Considering the 10-year life cycle of vehicles, we choose cryptographic primitives with a key length of 128 bits for our evaluation. For a real-world use, we focus on standardized cryptographic primitives.

In ISO/IEC 29192-5 [17], there are three hash functions, namely, PHOTON [18], SPONGENT [19], and Lesamnta-LW [9]. We chose Lesamnta-LW for our consideration because this standard characterizes this function as the (embedded)-software-oriented hash function whereas PHOTON and SPONGENT are characterized as hardware-oriented hash functions.

3.2.2 Applying of Cryptographic Hash Functions for Authenticated Encryption

To achieve the confidentiality and integrity of TPMS packets, we apply the Enc-then-MAC mechanism as is proposed in [8]. To reduce the implementation cost, for integrity protection of the TPMS communication, we again propose to use PRFs based on Lesamnta-LW as MAC generation algorithms. Note that the π function is changed from those used in PRFs for KDF. For TPMS-packet confidentiality protection, following the use of a block cipher in counter mode proposed in [8], we apply the underlying block cipher used in Lesamnta-LW in counter mode. By reusing Lesamnta-LW as the cryptographic primitive, we expect to reduce the **total** cost for integrity protection as well as key derivation.

For each data field in the TPMS packet, Table 2 list which property is protected in terms of confidentiality (**C**) and/or integrity (**I**).

3.2.3 FELICS-enhanced Performance Evaluation

We choose FELICS [20] as the evaluation tool. FELICS is a free and open-source benchmarking tool designed for software implementations of lightweight cryptographic primitives for embedded devices. It evaluates the performance of 21 lightweight block ciphers and three lightweight stream ciphers on the target micro-controllers: 8-bit AVR ATmega 128. The metrics are the execution time, RAM consumption and binary code size. There are **32** general purpose registers. In its Harvard memory architecture, there is **128 KB of Flash and 4 KB of SRAM**.

FELICS is an implementation performance evaluation tool for block ciphers and stream ciphers on micro-controllers. However, hash functions are not considered in the FELICS evaluation.

We first explain what is investigated regarding the interface of hash functions. To evaluate hash functions on FELICS, we view them as the composition of a mode of operation

Table 2: Confidentiality (**C**) and integrity (**I**) protection for each data field in the TPMS packet

Mechanism	SensorID	Pressure	Temperature	Reference
KATAN32-based Protocol	**C**	**I**	**I**	[5]
PRESENT-based Protocol	**C, I**	**C, I**	**C, I**	[2]
SPECK-based Protocol	**C, I**	**C, I**	**C, I**	[2]
PRESENT-based Enc-then-MAC	**C, I**	**C, I**	**C, I**	[8]
Lesamnta-LW -based PRF	**C, I**	**C, I**	**C, I**	This paper

and the underlying cryptographic primitive such as a block cipher. For example, Lesamnta-LW employs the AES-based block cipher as its underlying cryptographic primitive, whereas SHA-256 employs the SHACAL-2 block cipher. Therefore, we evaluate hash functions on FELICS using the following processes.

- Implement the internal process of hash functions as the block cipher
- Implement an iterative process that depends on the message length as the evaluation scenario in FELICS

We also implement hash function-based modes of operation such as PRF and HMAC as evaluation scenarios in FELICS.

It is important to evaluate the performance of the PRFs taking input messages shorter than 16 bytes. This is because we study how small the communication cost can be.

As for *data units*, the minimum data unit for all of our implementations is the `uint8_t` adopted by the FELICS encryption process evaluation adopts.

4 Our Evaluation Results on PRF employing a Lesamnta-LW Hash Function

For application to TPMS, we clarify the specification of the Lesamnta-LW-based PRFs introduced in Sect. 2 by specifying the π functions. Moreover, we evaluate their performance in the extended FELICS as explained in Sect. 3.2.3. We also evaluate the performance of HMAC employing the SHA-256 hash function to show the effectiveness of the Lesamnta-LW-based PRFs compared with those based on a general purpose hash function.

4.1 Specification of π function

We consider that PRF is used for generating session keys, namely K_{enc} and K_{auth} and the MACs in the vehicle ECU and four tires. As we use distinct values for the session keys in tires, different tires need different π functions. Therefore, our Enc-then-MAC mechanism uses five π functions. We now specify the π functions. The conditions on them are given as follows:

1. any pair of distinct permutations $\pi, \pi' \in \Pi$, $\pi(x) \neq \pi'(x)$ for every $x \in \mathcal{D}$

2. $\pi(IV) \neq \pi'(IV')$ for any $\pi, \pi' \in \Pi \cup \{id\}$ and $IV, IV' \in \mathcal{V}$ such that $(\pi, IV) \neq (\pi', IV')$.

In [10], the consideration of π function is given as follows,

Remark 1. [10] Let t_v and t_p be integers such that $t_v + t_p = n/2$. Let $V = \{IV_1, IV_2, \ldots, IV_a\}$ and $\Pi = \{\pi_1, \pi_2, \ldots, \pi_d\}$. Let $v_1, v_2, \ldots, v_a$ be distinct constants in Σ^{t_v}. Let $c_1, c_2, \ldots, c_d$ be distinct nonzero constants in Σ^{t_p}. Suppose that $IV_i = v_i || 0^{t_p}$ for $1 \leq i \leq a$ and that $\pi_j(x) \oplus (0^{t_v} || c_j)$ for $1 \leq j \leq d$. Then,

- $\Pi \cup \{id\}$ is pairwaise everywhere distinct, and
- since $\pi_j(IV_i) = v_i || c_j$, $\pi_j(IV_i) \neq \pi_{j'}(IV_{i'})$ if $(i, j) \neq (i', j')$.

The proofs of security of PRFs including the situation of Remark 1 are given in [10]. Therefore, we determine five π functions as follows: $\pi_0(y) = id$, $\pi_1(y) = y \oplus 0\text{x}1$, ..., $\pi_4(y) = y \oplus 0\text{x}4$. These functions fulfill condition 1. We use IVs that have the same value in four bits from the least significant bit (LSB) to fulfill condition 2. Since our π functions fulfill condition 1, they derive different values from the IVs.

4.2 Our Optimized Implementations of Lesamnta-LW and SHA-256

There is a reference implementation of Lesamnta-LW written by Kuwakado [21]. In his implementation of the underlying AES-based block cipher, a key scheduling process is performed until all round keys are generated. This implementation requires RAM to store all of the round keys. Based on this, we develop RAM-optimized implementations of cryptographic primitives in Lesamnta-LW.

Our implementation of Lesamnta-LW sequentially performs an encryption process and a key scheduling process in the round function. This implementation reduces the RAM required for data by decreasing that required for round keys.

In our implementations for both Lesamnta-LW and SHA-256, we mainly use uint32_t integers. The interfaces of input value in FELICS use uint8_t integers. Referring to the code of Trivium [22] in FELICS, we optimize the process in changing uint8_t to uint32_t. When we define a uint32_t array for storing uint8_t input values, we use a pointer as follows:

$$\text{uint32\_t } *\text{ex32} = (\text{uint32\_t } *)\text{ex},$$

where **ex** and **ex32** are uint8_t and uint32_t arrays, respectively. In this case, we modify the byte order of **ex32** to match that of **ex**. Since **ex32** shares RAM with **ex**, we can reduce the RAM consumption.

In our implementation of S-box and the round constants of both Lesamnta-LW and SHA-256, we use table-lookup implementations and store constants of S-box in ROM.

4.3 Our Results on Lesamnta-LW-based PRFs for KDFs and MACs

Table 3 presents our implementation results on PRF employing Lesamnta-LW, namely Lesamnta-LW-PRF/π_i for $0 \leq i \leq 4$ and HMAC employing SHA-256, namely HMAC-SHA-256, on an AVR ATmega 128 platform. Note that only implementations in C are considered here, and we use an 8-byte message.

According to Table 3, PRF employing a Lesamnta-LW hash function shows a better performance than HMAC-SHA-256 in terms of each metrics. Interestingly, our results show that Lesamnta-LW-PRF/π_i is about ten times faster than HMAC-SHA-256, and the amount of RAM on Lesamnta-LW-PRF/π_i is about one-tenth of the amount of RAM on HMAC-SHA-256. Since execution time is obtained from the number of cycles consumed in the cryptographic process, the fewer cycles the cryptographic process requires, the faster

the cryptographic process is. Our results show that Lesamnta-LW-PRF/π_i can have an advantage over HMAC-SHA-256, in terms of RAM and Time for the sensors for TPMS.

Table 3: Implementation Comparison on AVR ATmega 128 Platform

Name	Code [byte]	RAM (Data) [byte]	RAM (Stack) [byte]	# cycles @16MHz	Time [ms]	Compiler option
Lesamnta-LW-PRF/π_0*	1722	64	57	75925	4.75	-O1
Lesamnta-LW-PRF/π_0*	1780	64	52	68361	4.27	-O2
Lesamnta-LW-PRF/π_0*	2300	64	42	53956	3.37	-O3
Lesamnta-LW-PRF/π_0*	1702	64	59	77281	4.83	-Os
Lesamnta-LW-PRF/π_i (for $1 \leq i \leq 4$)	1732	64	57	75932	4.75	-O1
Lesamnta-LW-PRF/π_i (for $1 \leq i \leq 4$)	1790	64	52	68368	4.27	-O2
Lesamnta-LW-PRF/π_i (for $1 \leq i \leq 4$)	2310	64	42	53963	3.37	-O3
Lesamnta-LW-PRF/π_i (for $1 \leq i \leq 4$)	1712	64	59	77288	4.83	-Os
HMAC-SHA-256	2432	608	97	659469	41.22	-O1
HMAC-SHA-256	2384	608	99	659552	41.22	-O2
HMAC-SHA-256	2782	608	107	674392	42.15	-O3
HMAC-SHA-256	2468	608	107	703960	44.00	-Os

*: We consider that the process of xoring 0x0 is skipped by optimization of the compiler.

4.4 Our Results on Lesamnta-LW-based Authenticated Encryption

In Table 4, we present our results on Lesamnta-LW based authentication encryption. We estimate that the cost of Lesamnta-LW-based Enc-then-MAC is upper bounded by the cost of two calls to Lesamnta-LW PRF.

Table 4: Estimated Performance of Lesamnta-LW-based Authenticated Encryption

Mechanism	Key Length (Bits)	RAM (Bytes)	ROM (Bytes)	Comp. Time (ms)	Evaluation Environment	Ref.
Lesamnta-LW-based Enc-then-MAC	128	106	2,310	6.7	AVR ATmega 128	This paper

5 Limitation

Our experimental results on 8-bit AVR micro-controllers are at the simulation level, whereas the real implementation environment for TPMS, such as SP 37, might be more severe regarding the implementation resources available for cryptographic mechanisms. Therefore, further investigations would be needed.

All the codes we developed and evaluated here are written in C language. Since the optimization level is depended on the compiler, the optimization for performance is limited.

Our key management method is available for the management of session keys. The method of the management of the master key is important but is not provided in our paper.

6 Conclusion

In TPMS, the implementation cost constraints are severe, particularly the RAM cost on an 8-bit micro-controllers for vehicle ECUs. In this paper, we proposed the application of the known multiple PRFs based on lightweight hash function Lesamnta-LW to TPMS to ensure the confidentiality/integrity of data packet as well as to offer reasonable solutions for the key derivation in line with the NIST SP 800-108-specified KDF framework.

We confirmed the significant advantages of these proposed methods over the standard stack, namely, HMAC-SHA-256, by conducting experiments and evaluating the performance on the AVR 8-bit micro-controllers, on which we consider simulating TPMS sensors.

We expect that our results on Lesamnta-LW-based PRFs achieving 128-bit security contribute to the TPMS development from perspectives of its cost-efficiency and of the long life-cycle of the vehicle.

Furthermore, our proposed KDF shows an interesting property that the session key sequences for the four tires are independent of each other. We consider key compromise to be a serious issue. Having the key separation property that our KDF offers, the negative impact when this event happens would be minimized.

In future work, binding TESLA to our stack should help TPMS developers in terms of their cryptographic applications and implementations.

Acknowledgement

We would like to thank the anonymous reviewers for their valuable comments and Editage (www.editage.jp) for English language editing.

References

[1] Yuhei Watanabe, Hideki Yamamoto, and Hirotaka Yoshida. A study on the applicability of the lesamnta-lw lightweight hash function to tpms. 5th Embedded Security in Cars Conference Asia (escar Asia 2018), 2018.

[2] Cristina Solomon and Bogdan Groza. Limon - lightweight authentication for tire pressure monitoring sensors. In Security of Industrial Control Systems and Cyber Physical Systems - First Workshop, CyberICS 2015 and First Workshop, WOS-CPS 2015, Revised Selected Papers, volume 9588 of Lecture Notes in Computer Science, pages 95–111. Springer, 2015.

[3] Ishtiaq Rouf, Robert D. Miller, Hossen A. Mustafa, Travis Taylor, Sangho Oh, Wenyuan Xu, Marco Gruteser, Wade Trappe, and Ivan Seskar. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. In 19th USENIX Security Symposium, 2010, Proceedings, pages 323–338. USENIX Association, 2010.

[4] Christophe De Cannière, Orr Dunkelman, and Miroslav Knezevic. KATAN and KTANTAN - A family of small and efficient hardware-oriented block ciphers. In Cryptographic Hardware and Embedded Systems - CHES 2009, 11th International Workshop, Proceedings, volume 5747 of Lecture Notes in Computer Science, pages 272–288. Springer, 2009.

[5] Miao Xu, Wenyuan Xu, Jesse Walker, and Benjamin Moore. Lightweight secure communication protocols for in-vehicle sensor networks. In CyCAR'13, Proceedings of the 2013 ACM Workshop on Security, Privacy and Dependability for CyberVehicles, Co-located with CCS 2013, pages 19–30. ACM, 2013.

[6] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. SIMON and SPECK: block ciphers for the internet of things. IACR Cryptology ePrint Archive, 2015:585, 2015.

[7] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: an ultra-lightweight block cipher. In Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, volume 4727 of Lecture Notes in Computer Science, pages 450–466. Springer, 2007.

[8] Keita Emura, Takuya Hayashi, and Shiho Moriai. Toward securing tire pressure monitoring systems: A case of present-based implementation. In 2016 International Symposium on Information Theory and Its Applications, ISITA 2016, pages 403–407. IEEE, 2016.

[9] Shoichi Hirose, Kota Ideguchi, Hidenori Kuwakado, Toru Owada, Bart Preneel, and Hirotaka Yoshida. An AES based 256-bit hash function for lightweight applications: Lesamnta-lw. IEICE Transactions, 95-A(1):89–99, 2012.

[10] Shoichi Hirose, Hidenori Kuwakado, and Hirotaka Yoshida. A pseudorandom-function mode based on lesamnta-lw and the MDP domain extension and its applications. IEICE Transactions, 101-A(1):110–118, 2018.

[11] NIST. Special publication 800-108 recommendation for key derivation using pseudo-random functions (revised), October, 2009.

[12] Shoichi Hirose, Je Hong Park, and Aaram Yun. A simple variant of the merkle-damgård scheme with a permutation. J. Cryptology, 25(2):271–309, 2012.

[13] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference Proceedings, volume 1109 of Lecture Notes in Computer Science, pages 1–15. Springer, 1996.

[14] FIPS PUB 180-4. Secure hash standard (shs), August, 2015.

[15] Adrian Perrig, Ran Canetti, J. D. Tygar, and Dawn Song. The TESLA broadcast authentication protocol. RSA CryptoBytes, 5(2):2–13, Summar/Fall 2002.

[16] ISO/IEC 9798-2:2008 information technology – security techniques – entity authentication – part 2: Mechanisms using symmetric encipherment algorithms.

[17] ISO/IEC 29192-5:2016 information technology – security techniques – lightweight cryptography – part 5: Hash-functions.

[18] Jian Guo, Thomas Peyrin, and Axel Poschmann. The PHOTON family of lightweight hash functions. In Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, volume 6841 of Lecture Notes in Computer Science, pages 222–239. Springer, 2011.

[19] Andrey Bogdanov, Miroslav Knezevic, Gregor Leander, Deniz Toz, Kerem Varici, and Ingrid Verbauwhede. spongent: A lightweight hash function. In Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, volume 6917 of Lecture Notes in Computer Science, pages 312–325. Springer, 2011.

[20] FELICS - fair evaluation of lightweight cryptographic systems. https://www.cryptolux.org/index.php/FELICS.

[21] Hidenori Kuwakado. Lesamnta-lw reference c99 implementation. https://github.com/kuwakado/Lesamnta-LW, 2015. Github.

[22] Christophe De Cannière. Trivium: A stream cipher construction inspired by block cipher design principles. In Information Security, 9th International Conference, ISC 2006, Proceedings, volume 4176 of Lecture Notes in Computer Science, pages 171–186. Springer, 2006.