Hindawi
Security and Communication Networks
Volume 2020, Article ID 5738215, 12 pages
https://doi.org/10.1155/2020/5738215

Research Article

Lightweight Crypto Stack for TPMS Using Lesamnta-LW

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

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

Correspondence should be addressed to Yuhei Watanabe; yuhei.watanabe@aist.go.jp

Received 8 March 2019; Revised 20 July 2020; Accepted 10 September 2020; Published 24 September 2020

Academic Editor: Leonardo Mostarda

Modern vehicles which have internal sensor networks are one of the examples of a cyberphysical system (CPS). The tire pressure monitoring system (TPMS) is used to monitor the pressure of the tires and to inform the driver of them. This system is mandatory for vehicles in the US and EU. To ensure the security of TPMS, it is important to reduce the cost of the cryptographic mechanisms implemented in resource-constrained devices. To address this problem, previous works have 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 TPMS sensors equipped with the tires. In this paper, we propose the application of ISO/IEC 29192-5 lightweight hash function Lesamnta-LW to address these issues. When we apply cryptographic mechanisms to a practical system, we consider the lightweight crypto stack which contains cryptographic mechanisms, specifications for the implementation, and performance evaluation. Our approach is to apply the known method of converting Lesamnta-LW to multiple independent pseudorandom 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. We use the internal AES-based block cipher of Lesamnta-LW for encryption. 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 microcontrollers, on which we consider simulating TPMS sensors. We expect that our method to achieve multiple purposes with a single cryptographic primitive will help us to reduce the total implementation cost required for TPMS security.

1. Introduction

1.1. Positioning of This Paper. This paper is an extended version of the paper that appeared in the proceedings of the escar Asia 2018 [1]. The main changes and increments are as follows:

(1) Addition of the description of the relationship between modern vehicles and cyberphysical systems
(2) Addition of the background of the adoption of TPMS
(3) Addition of the definition of the TPMS sensor's architecture
(4) Addition of descriptions of the lightweight crypto stack
(5) Addition of specifications of TPMS sensors such as SP37, SP40+, and FXTH87
(6) Revision of descriptions of applications of Lesamnta-LW
(7) Implementation results of the Lesamnta-LW-based authenticated encryption mechanism

Yoshida who is one of the coauthors of this paper mentioned the above known results as current tendency of the lightweight cryptography in his presentation at ASK 2018 [2].

1.2. Background and Our Contribution. Modern vehicles are one of the examples of a cyberphysical system (CPS) because they have many sensors and computer systems which are connected via internal networks. There have been security analyses for systems in the modern vehicle, and the demand for security has been shown by them [3–5].

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 [6]. Since there was the Ford–Firestone tire failure controversy in the US [7], TREAD Act was established and TPMSs were the wide deployment. Currently, it is mandatory for many new vehicles to adopt the TPMS in the US [8] and EU [9]. Since it is difficult to establish wire connection between the vehicles and the TPMS sensors, the wireless communication by a radio frequency (RF) is used in the TPMS. Then, it is considered that there are security risks by eavesdropping communication packets of TPMS [10].

In 2010, eavesdropping and spoofing attacks on TPMS were reported [10]. 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 non-cryptographic key management protocols for TPMS. For instance, a packet protection protocol using the hardware-oriented block cipher KATAN32 [11] has been proposed, and the FPGA implementation of this protocol has been evaluated in terms of delay and power [12]. The authors proposed a session key update mechanism but did not apply a cryptographic mechanism. In [6], a lightweight protocol based on lightweight hardware-oriented block ciphers such as SPECK [13] and PRESENT [14] was proposed, and this achieved energy consumption of some two orders of magnitude lower than that reported in [12].

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 was studied using Infineon SP37 sensors [15] implemented on an 8-bit microcontroller. In [16], the authors propose a packet protection protocol using PRESENT and show that their implementations are suitable for the NXP FXTH87 TPMS sensor [17] in terms of RAM, ROM, and computation time.

We argue the important problem is that it appears to us that none of the above works [6, 12, 16] 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.

1.2.1. TPMS Sensor's Architecture. A TPMS sensor consists of devices such as a pressure sensor, a microcontroller, an RF transmitter, and a low-frequency receiver. TPMS sensors such as SP37 and FXTH87 usually use an 8-bit microcontroller. The Infineon SP40+ sensor [18] which is a successor product to SP37 also employs an 8-bit microcontroller. Thus, an 8-bit microcontroller is widely used in the TPMS sensor. TPMS sensors have constrained resources of RAM. For instance, there are 256 bytes of RAM in SP37, 160 bytes of RAM in SP40+, and 512 bytes of RAM in FXTH87. In this paper, we assume that the architecture of the TPMS sensor employs an 8-bit microcontroller and a constrained RAM. In our performance evaluation, we use the tool supporting 8-bit devices.

1.2.2. Performance Evaluation Tool. There is a free and open-source benchmarking framework named FELICS [19] to evaluate lightweight symmetric primitives in the embedded devices such as 8-bit, 16-bit, and 32-bit microcontrollers. FELICS supports evaluations of lightweight block ciphers and lightweight stream ciphers. It also provides evaluation scenarios which assume the communication protocol and the challenge-handshake authentication protocol in the IoT context. Nowadays, there are studies extending FELICS to evaluate an authenticated encryption scheme [20, 21].

1.2.3. Perspectives from Lightweight Crypto Stack. The lightweight crypto stack is summarized with essential perspectives to apply a lightweight cryptographic mechanism to the CPS. When an application of lightweight cryptographic mechanisms is constructed, consideration of integrated multiple perspectives is important to reduce the cost and to ensure the practicality. Especially, since resources for cryptographic mechanisms are constrained, it is also important that multiple cryptographic applications should be obtained from few cryptographic primitives. For instance, there have been a number of studies that have realized multiple symmetric key ciphers such as the block cipher, stream cipher, MAC, and hash function from single cryptographic primitive [22–26]. The lightweight hash function Lesamnta-LW [27] specified in ISO/IEC 29192-5 is one of the primitives which has the above mentioned characteristic. It employs the internal AES-based block cipher. There is a pseudorandom function mode [28] and an authenticated encryption with associated data (AEAD) [29] based on Lesamnta-LW. We consider that we obtain multiple applications for a TPMS protocol by using one primitive. We propose our methods based on the above concept.

1.2.4. Our Contribution. In this paper, we propose the application of the ISO/IEC 29192-5 embedded software-oriented hash function Lesamnta-LW 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 pseudorandom functions (PRFs) based on Lesamnta-LW for MAC generation as well as key derivation in the NIST SP800-108-specified key derivation function (KDF) framework [30]. We also obtain the encryption method by using the internal AES-based block cipher of Lesamnta-LW. We present an authenticated encryption mechanism by using the MAC generation method and the encryption method based on Lesamnta-LW. 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 8-bit microcontroller which we consider a vehicle implementation environment. Our proposed KDF properties that the session key sequences for the four tires are independent of each other. As a result, we propose three applications based on Lesamnta-LW and clarify the implementation specification for the TPMS use. In our evaluation, we compare Lesamnta-LW-based PRFs and HMAC-SHA-256. We use FELICS for the performance evaluation to fairly evaluate different cryptographic mechanisms. FELICS supports the evaluation on the AVR 8-bit microcontroller such as AVR ATmega 128. In order to evaluate hash function-based mechanisms, we extend the evaluation scenario in FELICS. We evaluate the performance of each mechanism and confirm the advantage of Lesamnta-LW-based PRFs. Therefore, our approach is constructed along the lightweight crypto stack and to obtain three applications from the Lesamnta-LW-based primitive.

1.2.5. Expected Impact on TPMS Development. In recent years, 8-bit microcontrollers such as Infineon SP37 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 microcontroller 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 us to minimize the negative impact of this event.

1.2.6. 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 and the Lesamnta-LW-based authenticated encryption mechanism. Section 5 explores the limitations of our work. Section 6 presents our conclusion.

2. Preliminary

2.1. Overview of the Target TPMS. Referring to [10], 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 Figure 1.

2.1.1. Tire Pressure Monitoring Sensors SP37 and SP40+. SP37 is a battery-powered sensor for TPMS and a product of Infineon. In [15], features of SP37 are summarized. This sensor has an 8-bit low-power embedded microcontroller based on Intel 8051, and we can use the instruction set compatible to the standard 8051 processor. 6 kB of flash memory for application codes, 16 kB of ROM for ROM library functions, and 256 bytes of RAM are present. Customers can implement their applications to this sensor and store them to flash memory.

SP40+ is a successor product to SP37. In [18], features of SP40+ are summarized. 12 kB of flash memory for application codes, 40 kB of ROM for firmware, and 160 bytes of RAM are present.

2.1.2. Tire Pressure Monitoring Sensor FXTH87. FXTH87 is a sensor used for TPMS and a product of NXP. In [17], features of FXTH87 are summarized. This sensor has an 8-bit S08 microcontroller. 8 kB of flash memory for application codes and 512 bytes of RAM are present. Customers can implement their applications to this sensor and store them to flash memory.

2.2. Lightweight Crypto Stack for Embedded Systems. In the lightweight crypto stack, there are six components, e.g., the cryptographic primitive, mode of operation, protocol, standard, specification, and implementation. We describe them by using the stack form as shown in Figure 2. The cryptographic primitive is described by a primitive. In a mode of operation, we consider how to use primitives to protect confidentiality and/or integrity. Cryptographic components in a protocol consist of primitives and modes of operation. When we use cryptographic techniques in a practical system, we choose standardized one to use more reliable techniques. We clarify detailed specifications of applications in the system to evaluate performances and costs. We implement them to evaluate performances on the microcontroller and clarify effectiveness and feasibility in the practical environment. When we propose applications of cryptographic mechanisms for the practical system, we consider them along the lightweight crypto stack.

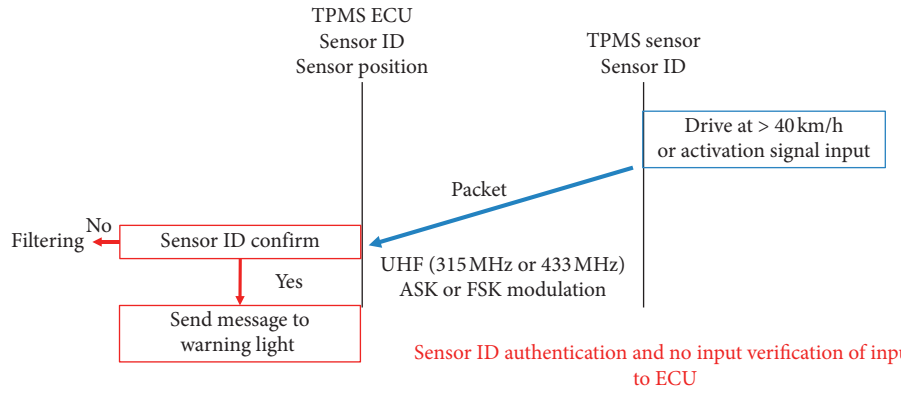

Figure 1: Overview of TPMS.

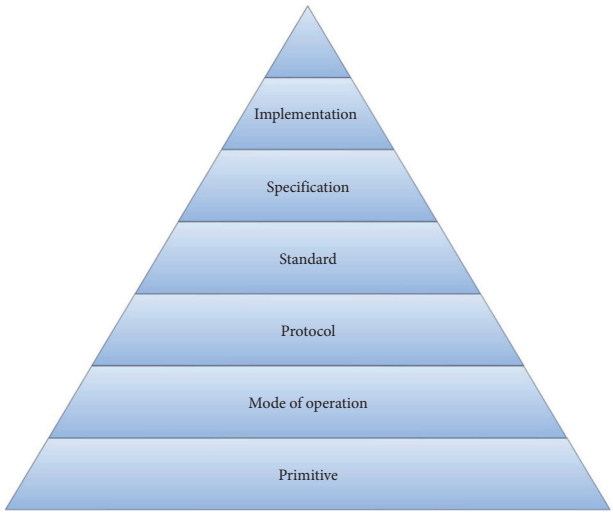

Figure 2: Lightweight crypto stack.

In the embedded system, implementation resources are severely constrained. From perspectives between the primitive and the protocol, we consider that multiple applications are obtained from few primitives to reduce costs for the cryptographic mechanisms in a system.

2.3. Applications of Lesamnta-LW Lightweight Hash Function

2.3.1. Notations and Definitions. Following [28], we introduce notations and definitions for explaining specifications of cryptographic mechanisms. Let $\Sigma = \{0, 1\}$. For any nonnegative 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. Let $(\Sigma^l)^+ = \cup_{i\geq 1}(\Sigma^l)^i$.

For $x \in (\Sigma^l)^+$, the length of x is denoted by $|x|$. The concatenation of x_1 and x_2 in $(\Sigma^l)^+$ is denoted by $x_1\|x_2$.

For any natural number n, let D be Σ^n. Let $\mathbf{P}(D)$ denote the set of all permutations on D. id represents an identity permutation. Let $\mathbf{C}(\kappa, n)$ be the set of all block ciphers with key size κ and block size n. A block cipher in $\mathbf{C}(\kappa, n)$ is called a (κ, n) block cipher.

Let $\Pi \subset \mathbf{P}(D)$. We say that Π is pairwise distinct everywhere if, for any pair of distinct permutations π, $\pi' \in \Pi$, $\pi(x) \neq \pi'(x)$ for every $x \in D$.

2.3.2. Hashing Mode of Lesamnta-LW and Its Variant with MDP. Lesamnta-LW is a Merkle–Damgård iterated hash function [27]. It is the plain Merkle–Damgård iteration of a block cipher E taking input as $n/2$-bit key and n-bit plaintext in $\mathbf{C}((n/2), n)$, 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 the 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 [31] is introduced as follows. Hereafter, it is assumed that the underlying block cipher E is in $\mathbf{C}(w, n)$, where $w < n$. The MDP variant with a permutation π on Σ^{n-w} is the function $J^{E,\pi}$: $\Sigma^n \times (\Sigma^w)^+ \longrightarrow \Sigma^n$, which is defined as follows: for $X_1, X_2, \ldots, X_x \in \Sigma^w$ and $Y_0 \in \Sigma^n$,

$$J^{E,\pi}\left(Y_0, X_1\|X_2\|\cdots\|X_x\right) = Y_x, \tag{1}$$

such that

$$Y_i \longleftarrow \begin{cases} E_{Y_{i-1,0}}\left(X_i\|Y_{i-1,1}\right), & (1 \leq i \leq x-1), \\ E_{Y_{i-1,0}}\left(X_i\|\pi\left(Y_{i-1,1}\right)\right), & (i = x), \end{cases} \tag{2}$$

where $Y_j = Y_{j,0}\|Y_{j,1} \in \Sigma^n$ and $|Y_{j,0}| = w$ 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.

2.3.3. Multiple PRFs Based on Lesamnta-LW. Let $J_{IV}^{E,\pi}$: $\Sigma^{n/2} \times (\Sigma^w)^+ \longrightarrow \Sigma^n$ 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. $J_{IV}^{E,\pi}(K, \cdot)$ is also denoted by $J_{IV,K}^{E,\pi}(\cdot)$. For $\Pi \subset \mathbf{P}(\Sigma^{n-w})$ and $V \subset \Sigma^{n-w}$, let

$$J_V^{E,\Pi} = \left\{ J_{IV}^{E,\pi} \,\middle|\, IV \in V \wedge \pi \in \Pi \right\}. \tag{3}$$

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

Theorem 1 (see [27]). *This theorem 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 Figure 3. In this way, one cryptographic primitive can be converted to produce multiple cryptographic functions with a small additional cost.*

In [27], π functions used in Lesamnta-LW-based PRFs are not specified. Hence, when applied to real-world systems, one has to specify them.

2.4. HMAC-SHA-256: Hash Function-Based PRF. HMAC-SHA-256 [32] is a widely known PRF based on the SHA-256 [33] hash function for a general purpose. The overview of HMAC-SHA-256 is given in Figure 4, where M is a message input, K is a secret key, H is SHA-256, $\mathrm{ipad} = 0 \times 3636...36$, and $\mathrm{opad} = 0 \times 5c5c...5c$.

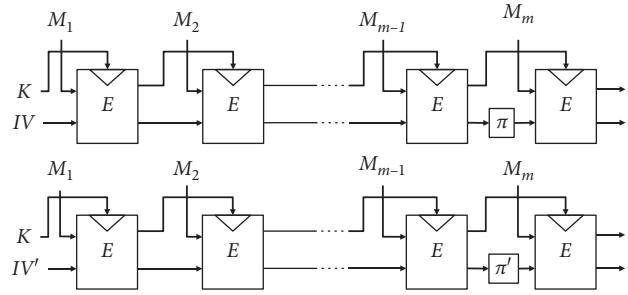

Figure 3: The multiple PRFs based on Lesamnta-LW [27].

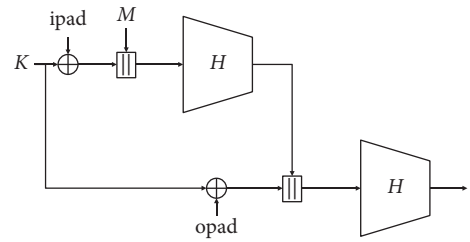

Figure 4: Overview of HMAC-SHA-256 PRF.

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 confidentiality 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 [12], 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 life cycles 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 [6, 16] is appropriate for **long-term security** over the life cycles that vehicles typically (should) achieve.

3.1.2. Implementation Cost for the TPMS Sensors. The cryptographic protocol for TPMS presented in [12] 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 microcontroller. 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 [12], 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 that 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 [6], 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 SP37 sensors. The proposed protocol is similar

to the protocol proposed in [12] and processing 96-bit data frames. However, they showed that implementation evaluation where the environment is SP37 8-bit microcontrollers. The lightweight metrics of energy consumption and power consumption for the TPMS sensors are the main focus of the evaluation experiments. Table 1 shows the performances of previous cryptographic protocols for TPMS.

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 [12] 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 [34] 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 [6], 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 [35] for key establishment. However, this standard was developed for generic purposes rather than specific wireless systems such as TPMS.

In [16], the authors propose a protocol using PRESENT as underlying lightweight cipher and CMAC mode as the mode of operation. They showed the simple Enc-then-MAC paradigm by PRESENT and CMAC. 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 [6, 12, 16] appears 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. We consider our approaches along the lightweight crypto stack as shown in Figure 2.

3.2.1. Applying Lesamnta-LW Hash Function to TPMS Key Derivation. It would be a challenge that the key management problem is solved at 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 than the 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 [30] that describes how to generate or **derivate the session keys** required for confidentiality and/or integrity protection, which could be performed by means of KDF, as shown in Figure 5. Figure 5 shows the example on the right front (RF) tire and the left front (LF) tire. Here, for the 16-byte encryption key K_{enc} and the 16-byte authenticated key K_{auth}, $K_{\text{enc}} \| K_{\text{auth}}$ 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.

We show how to manage the session keys for the encryption and authentication in four tires. Four tires are denoted by the RF tire, the LF tire, the right back (RB) tire, and the left back (LB) tire. We assume that one master key is installed in the ECU and each TPMS sensor. We use PRFs based on Lesamnta-LW to obtain four independent session key sequences from a master key. As mentioned in Section 2.3.3, PRFs based on Lesamnta-LW can construct independent PRFs by changing the permutation π. 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 sends 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 $K_{\text{LF}} = \text{KDF}_{\text{LF}}(K_m, C_{\text{LF}}, i)$, $K_{\text{RF}} = \text{KDF}_{\text{RF}}(K_m, C_{\text{RF}}, i)$, $K_{\text{LB}} = \text{KDF}_{\text{LB}}(K_m, C_{\text{LB}}, i)$, and $K_{\text{RB}} = \text{KDF}_{\text{RB}}(K_m, C_{\text{RB}}, i)$. When a sensor updates the session key, a sensor updates the counter i and calculates value for the session key by PRF. The ECU can obtain the session keys by using same process. We show the overview of the above process in Figure 6.

We discuss the scenario of the session key compromise. By achieving **key separation**, even if one session key is compromised, there is 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 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 the standardized cryptographic primitives.

In ISO/IEC 29192-5 [36], there are three hash functions, namely, PHOTON [37], SPONGENT [38], and Lesamnta-LW [27]. 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.

Table 1: Key length and performance of the previous methods.

Mechanism	Key length (bits)	RAM (bytes)	ROM (bytes)	Comp. time (ms)	Evaluation environment	Reference
KATAN32-based protocol*	80	N/A	N/A	36.6	Arduino Uno + RFM22	[12]
PRESENT-based protocol	80	27	846	303.3	SP37	[6]
SPECK-based protocol	128	20	865	29.1	SP37	[6]
PRESENT-based Enc-then-MAC	80 or 128 (unspecified)	502	7,013	19.4	RL78	[16]

*In this study, KATAN32-based implementation requires $1231uJ$ w.r.t power [12].

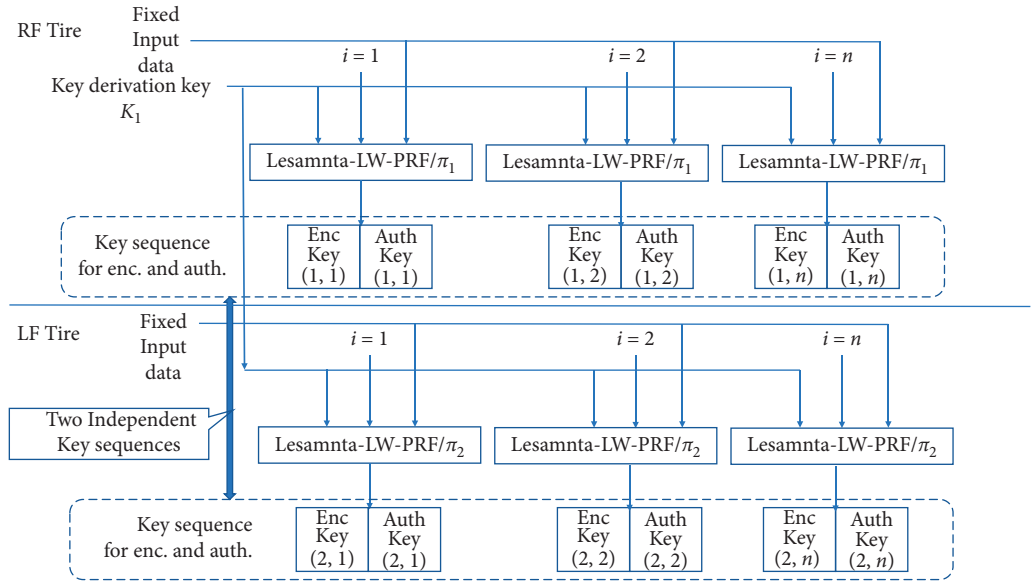

Figure 5: Application of Lesamnta-LW-based PRFs to NIST SP 800-108 KDF in the counter mode for TPMS.

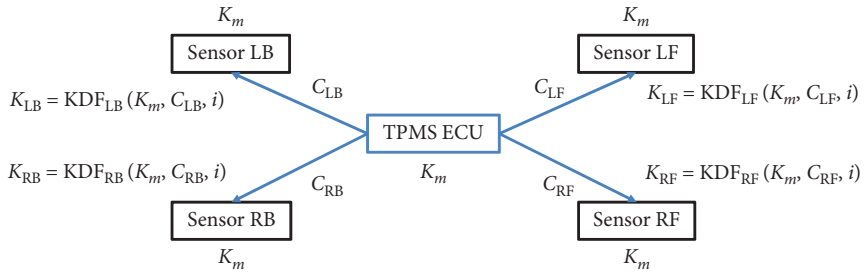

Figure 6: Scenario of session key management.

3.2.2. Applying 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 [16]. 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 [16], we apply the underlying block cipher used in Lesamnta-LW in the 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 lists which property is protected in terms of confidentiality (**C**) and/or integrity (**I**).

3.2.3. FELICS-Enhanced Performance Evaluation. We choose FELICS [19] 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 microcontrollers: 8-bit AVR ATmega 128. The metrics are the execution time, RAM consumption,

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

Mechanism	Sensor ID	Pressure	Temperature	Reference
KATAN32-based protocol	**C**	**I**	**I**	[12]
PRESENT-based protocol	**C, I**	**C, I**	**C, I**	[6]
SPECK-based protocol	**C, I**	**C, I**	**C, I**	[6]
PRESENT-based Enc-then-MAC	**C, I**	**C, I**	**C, I**	[16]
Lesamnta-LW-based PRF	**C, I**	**C, I**	**C, I**	This paper

and binary code size. **32** general purpose registers are present. In its Harvard memory architecture, there is **128 kB of Flash and 4 kB of SRAM**. This microcontroller has more implementation resources than those of SP37 as shown in Section 2.1.1. The limitations of our approach for implementations are mentioned in Section 5.

FELICS is an implementation performance evaluation tool for block ciphers and stream ciphers on microcontrollers. 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 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 [39]. Therefore, we evaluate hash functions on FELICS using the following processes:

(i) Implement the internal process of hash functions as the block cipher

(ii) 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, HMAC, and Enc-then-MAC as evaluation scenarios in FELICS. Our work of FELICS extension is summarized in Figure 7 where the files we added are indicated in italic.

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 Section 2 by specifying the π functions. Moreover, we evaluate their performance in the extended FELICS as explained in Section 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 $x \in D$

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

In [28], the consideration of π function is given as follows:

Remark 1 (see [28]). 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,

(i) $\Pi \cup \{\text{id}\}$ is pairwise distinct everywhere

(ii) 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 [28]. Therefore, we determine five π functions as follows: $\pi_0(y) = \text{id}$, $\pi_1(y) = y \oplus 0\text{x}1, \ldots, \pi_4(y) = y \oplus 0\text{x}4$. 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 [40]. 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

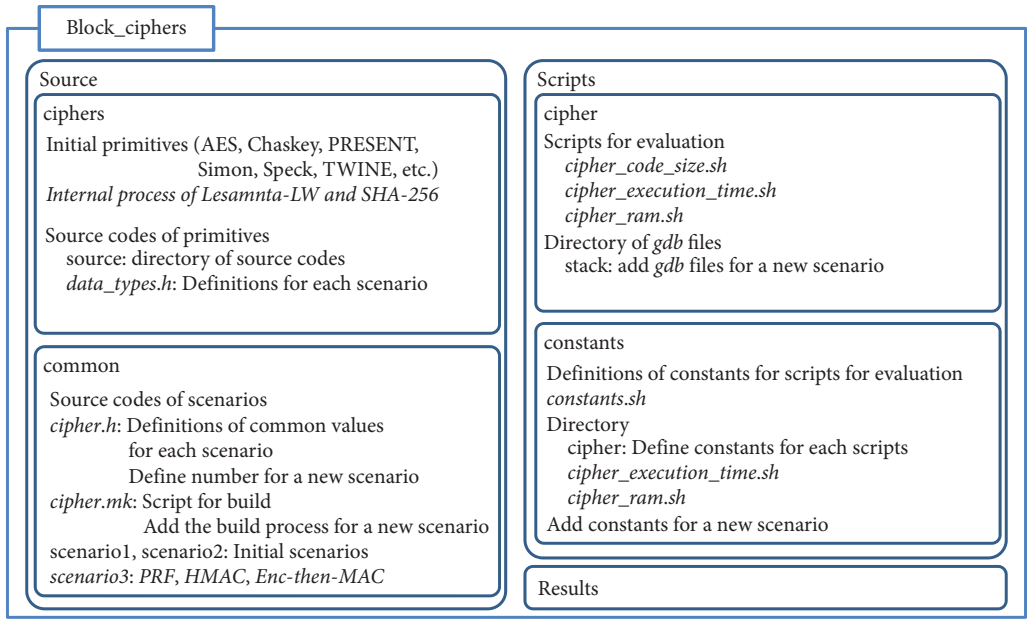

Figure 7: Extension of FELICS for hash function-based modes evaluation.

code of Trivium [41] 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:

$$\text{unit } 32\_t * \text{ex32} = (\text{unit } 32\_t *)\text{ex}, \tag{4}$$

where **ex** and **ex32** are the 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 $0 \le i \le 4$, 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 the cycles 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.

4.4. Our Results on Lesamnta-LW-Based Authenticated Encryption. In Table 4, we present our implementation results on Lesamnta-LW-based authenticated encryption. Note that only implementation in C is considered here, and we use an 8-byte message. We evaluate the performance of the Lesamnta-LW-based Enc-then-MAC mechanism. In our implementations, RAM (Data) contains data of the block, message, encryption key, authentication key, counter, and IV. Our approach requires 120 (= 32 + 8+16 + 16+32 + 16) RAM (Data) bytes. Remarkably, our mechanism only requires 183 (=120 + 63) RAM bytes.

5. Limitation

Our experimental results on 8-bit AVR microcontrollers are at the simulation level, whereas the real implementation environment for TPMS, such as SP37, 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 depends 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.

We show the authenticated encryption scheme based on Lesamnta-LW by using the Enc-then-MAC technique. The comparison between our scheme and the Lesamnta-LW-based AEAD scheme is not provided in this paper because our paper has been submitted before the presentation of the Lesamnta-LW-based AEAD scheme.

Table 3: Implementation comparison on AVR ATmega 128 platform.

Name	Code (byte)	RAM (data) (byte)	RAM (stack) (byte)	No of cycles @16 MHz	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 $1 \le i \le 4$)	1732	64	57	75932	4.75	−O1
Lesamnta-LW-PRF/π_i (for $1 \le i \le 4$)	1790	64	52	68368	4.27	−O2
Lesamnta-LW-PRF/π_i (for $1 \le i \le 4$)	2310	64	42	53963	3.37	−O3
Lesamnta-LW-PRF/π_i (for $1 \le i \le 4$)	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.

Table 4: Implementation results of Lesamnta-LW-based authenticated encryption on AVR ATmega 128 platform.

Mechanism	Code (byte)	RAM (data) (byte)	RAM (stack) (byte)	No. of cycles @16 MHz	Time (ms)	Compiler option
Lesamnta-LW-based Enc-then-MAC	1846	120	74	155652	9.73	−O1
	1962	120	68	144449	9.03	−O2
	2726	120	63	114963	7.19	−O3
	1836	120	81	157802	9.86	−Os

6. Conclusion

In TPMS, the implementation cost constraints are severe, particularly the RAM cost on 8-bit microcontrollers for vehicle ECUs. In this paper, we considered the lightweight crypto stack for TPMS and 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 microcontrollers, 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.

Data Availability

The data used to support the findings of this study have not been made available because of the contract of the collaboration research.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] Y. Watanabe, H. Yamamoto, and H. Yoshida, "A study on the applicability of the Lesamnta-LW lightweight hash function to tpms," in *Proceedings of the 5th Embedded Security in Cars Conference Asia (escar Asia 2018)*, Tokyo, Japan, 2018.

[2] H. Yoshida, "On the design and use of lightweight cryptography for cyber-physical systems," in *Proceedings of the 8th Asian Workshop on Symmetric Key Cryptography (ASK 2018)*, Kolkata, India, 2018, https://www.isical.ac.in/~ask2018/slides/hirotaka-yoshida.pdf.

[3] K. Koscher, A. Czeskis, F. Roesner et al., "Experimental security analysis of a modern automobile," in *Proceedings of the 31st IEEE Symposium on Security and Privacy (S&P 2010)*, pp. 447–462, Oakland, CA, USA, May 2010.

[4] S. Checkoway, D. McCoy, B. Kantor et al., "Comprehensive experimental analyses of automotive attack surfaces," in *Proceedings of the 2011 20th USENIX Security Symposium*, San Francisco, CA, USA, August 2011.

[5] J. Petit and S. E. Shladover, "Potential cyberattacks on automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 546–556, 2015.

[6] C. Solomon and B. Groza, "Limon-lightweight authentication for tire pressure monitoring sensors," in *Proceedings of the Security of Industrial Control Systems and Cyber Physical Systems-First Workshop, CyberICS 2015 and First Workshop, WOS-CPS 2015*, pp. 95–111, Vienna, Austria, 2015.

[7] S. Govindjee, *Firestone Tire Failure Analysis,* 2001, http://faculty.ce.berkeley.edu/sanjay/REPORT_Secure.PDF.

[8] National Highway Traffic Safety Administration, *Federal Motor Vehicle Safety Standards; Tire Pressure Monitoring Systems; Controls and Displays*, NHTSA, Washington, DC, USA, 2005, https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/fmvss/TPMSfinalrule_6.pdf.

[9] European Commission, "Cars safer from 1 November 2012," 2012, http://europa.eu/rapid/press-release_IP-12-1169_en.htm.

[10] I. Rouf, R. D. Miller, H. A. Mustafa et al., "Security and privacy vulnerabilities of in-car wireless networks: a tire pressure monitoring system case study," in *Proceedings of the 19th USENIX Security Symposium*, Washington, DC, USA, August 2010.

[11] C. De Cannière, O. Dunkelman, and M. Knezevic, "KATAN and KTANTAN - a family of small and efficient hardware-oriented block ciphers," in *Proceedings of the 11th International Workshop on Cryptographic Hardware and Embedded Systems-CHES 2009*, pp. 272–288, Lausanne, Switzerland, September 2009.

[12] M. Xu, W. Xu, J. Walker, and B. Moore, "Lightweight secure communication protocols for in-vehicle sensor networks," in *Proceedings of the 2013 ACM Workshop on Security, Privacy and Dependability for CyberVehicles, Co-located with CCS 2013 CyCAR'13*, Berlin, Germany, November 2013.

[13] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "SIMON and SPECK: block ciphers for the internet of things," *IACR Cryptology ePrint Archive*, vol. 2015, no. 585, 2015.

[14] A. Bogdanov, L. R. Knudsen, G. Leander et al., "PRESENT: an ultralightweight block cipher," in *Proceedings of the 9th International Workshop on Cryptographic Hardware and Embedded Systems-CHES 2007*, pp. 450–466, Vienna, Austria, September 2007.

[15] Infineon Technologies AG, *Sp37 High Integrated Single-Chip Tpms Sensor with a Low Power Embedded Micro-Controller and Wireless fsk/ask uhf Transmitter*, Infineon, Neubiberg, Germany, 2012.

[16] K. Emura, T. Hayashi, and S. Moriai, "Toward securing tire pressure monitoring systems: a case of present-based implementation," in *Proceedings of the 2016 International Symposium on Information Theory and its Applications (ISITA 2016)*, pp. 403–407, Monterey, CA, USA, April 2016.

[17] NXP Semiconductors, Fxth87 tire pressure monitor sensor family, https://www.nxp.com/products/sensors/pressure-sensors/tire-pressure-monitoring-sensors/fxth87-tire-pressure-monitor-sensor-family:fxth87.

[18] Infineon Technologies AG, *Sp40plus Tire Pressure Monitoring Sensor*, Infineon, Neubiberg, Germany, 2020.

[19] FELICS, Fair evaluation of lightweight cryptographic systems, https://www.cryptolux.org/index.php/FELICS.

[20] L. Cardoso dos Santos, J. Großschädl, and A. Biryukov, "FELICS-AEAD: benchmarking of lightweight authenticated encryption algorithms," in *Proceedings of the 2019 NIST Lightweight Cryptography Workshop*, Gaithersburg, MD, USA, 2019.

[21] K. Le Gouguec, "FELICS-AE: a framework to benchmark lightweight autheticated block ciphers," in *Proceedings of the 2019 NIST Lightweight Cryptography Workshop*, Gaithersburg, MD, USA, 2019.

[22] D. J. BernsteinS. Kölbl et al., "GIMLI: a cross-platform permutation," in *Proceedings of the 19th International Conference Cryptographic Hardware and Embedded Systems (CHES 2017)*, pp. 299–320, Taipei, Taiwan, 2017.

[23] R. AlTawy, R. Rohit, M. He, K. Mandal, G. Yang, and G. Gong, "SLISCP-light: towards hardware optimized sponge-specific cryptographic permutations," *ACM Transactions on Embedded Computing Systems*, vol. 17, no. 4, pp. 81:1–81:26, 2018.

[24] K. Stoffelen and J. Daemen, "Column parity mixers," *IACR Transactions on Symmetric Cryptology*, vol. 2018, no. 1, pp. 126–159, 2018.

[25] J. Daemen, S. Hoffert, G. Van Assche, and R. Van Keer, "The design of Xoodoo and Xoofff," *IACR Transactions on Symmetric Cryptology*, vol. 2018, no. 4, pp. 1–38, 2018.

[26] R. AlTawy, R. Rohit, M. He, K. Mandal, G. Yang, and G. Gong, "Towards a cryptographic minimal design: the sLISCP family of permutations," *IEEE Transactions on Computers*, vol. 67, no. 9, pp. 1341–1358, 2018.

[27] S. Hirose, K. Ideguchi, H. Kuwakado, T. Owada, B. Preneel, and H. Yoshida, "An AES based 256-bit hash function for lightweight applications: Lesamnta-LW," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E95-A, no. 1, pp. 89–99, 2012.

[28] S. Hirose, H. Kuwakado, and H. Yoshida, "A pseudorandom-function mode based on Lesamnta-LW and the MDP domain extension and its applications," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E101.A, no. 1, pp. 110–118, 2018.

[29] S. Hirose, H. Kuwakado, and H. Yoshida, "Authenticated encryption based on Lesamnta-LW hashing mode," in *Proceedings of the 22nd International Conference on Information Security and Cryptology (ICISC 2019)*, pp. 52–69, Seoul, South Korea, December 2019.

[30] NIST, *Special publication 800-108 recommendation for key derivation using pseudorandom functions (revised)*, NIST, Gaithersburg, MD, USA, 2009.

[31] S. Hirose, J. H. Park, and A. Yun, "A simple variant of the Merkle-Damgård scheme with a permutation," *Journal of Cryptology*, vol. 25, no. 2, pp. 271–309, 2012.

[32] M. Bellare, R. Canetti, and H. Krawczyk, "Keying hash functions for message authentication," in *Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology-CRYPTO '96*, pp. 1–15, Santa Barbara, CA, USA, August 1996.

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

[34] A. Perrig, R. Canetti, J. D. Tygar, and D. Song, "The TESLA broadcast authentication protocol," *RSA CryptoBytes*, vol. 5, no. 2, pp. 2–13, 2002.

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

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

[37] J. Guo, T. Peyrin, and A. Poschmann, "The PHOTON family of lightweight hash functions," in *Proceedings of the 31st Annual Cryptology Conference on Advances in Cryptology-CRYPTO 2011*, pp. 222–239, Santa Barbara, CA, USA, August 2011.

[38] A. Bogdanov, M. Knezevic, G. Leander, D. Toz, K. Varici, and I. Verbauwhede, "SPONGENT: a lightweight hash function," in *Proceedings of the 13th International Workshop*

Cryptographic Hardware and Embedded Systems (CHES 2011), pp. 312–325, Nara, Japan, September 2011.

[39] H. Handschuh and D. Naccache, "Shacal," in *Proceedings of the First open NESSIE Workshop*, Heverlee, Belgium, 2000.

[40] H. Kuwakado, "Lesamnta-LW reference c99 implementation," 2015, https://github.com/kuwakado/Lesamnta-LW.

[41] C. De Cannière, "Trivium: a stream cipher construction inspired by block cipher design principles," in *Proceedings of the 9th International Conference on Information Security (ISC 2006)*, pp. 171–186, Samos Island, Greece, August 2006.