Lightweight Authentication for Low-End Control Units with Hardware Based Individual Keys*

Sergei Bauer
KAI GmbH
Villach, Austria
sergei.bauer@k-ai.at

Martin Brunner
Infineon Technologies AG
Munich, Germany
martin.brunner2@infineon.com

Peter Schartner
Alpen-Adria-Universität Klagenfurt
Klagenfurt, Austria
Peter.schartner@aau.at

Abstract* — With increasing autonomous features of vehicles, key issues of robotic- and automotive engineering converge toward each other. Closing existing security gaps of device communication networks will be an enabling feature for connecting autonomously interacting systems in a more secure way. We introduce a novel approach for deriving a secret key using a lightweight cipher in the firmware of a low-end control unit. In this approach, we propose to use a non-standardized lightweight algorithm with unique hardware based parameters to prevent duplicate key generation. The randomness of the selected cipher was assessed by applying the NIST statistical test suite to produced key values. By evaluating the method on a typical low-end automotive platform, we could demonstrate the realistic applicability of the solution. The proposed method counteracts a known security issue in device communication between control units not only present in automotive solutions but also in the robotics domain. The security of the implemented solution has been compared to current automotive guidelines and recommendations for the security of resource constrained devices, also present in robotics. This approach allows low-end communication systems to be enhanced by message- and device authentication.

Keywords– Lightweight Cryptography, Message Authentication, Robotic Network Security, Automotive Device Authentication, ARX Block Cipher, Electrical Control Unit

I. INTRODUCTION

Currently vehicles are not equipped with security features to authenticate low-end control units. Devices attached to the communication bus might originate from stolen vehicles or could even be altered to manipulate the vehicles' behavior. Low-end automotive control units that carry out simple tasks are not equipped with any cryptographic security mechanisms and can easily be stolen or manipulated, as comprehensively summarized by Valasek et al. [30]. The following classes for electrical control units (ECUs) will be used in this publication:

Class 1: High-end automotive control units equipped with an automotive grade processor containing an integrated hardware security module (HSM) and a Common Criteria certified security component for cryptographic operations.

Class 2: Mid-range automotive processor units that only contain a HSM integrated into the processor core but no discrete security component.

Class 3: Low-end units that contain an automotive grade processor without any cryptographic hardware accelerators, used for lighting and add-on features.

The authors describe a novel approach for generating a secret key that is individual to each device and is based on a parameter introduced by the silicon vendor already available inside processors used for class-3 devices. This secret key can be used to authenticate the piece of hardware itself or any other cryptographic operation such as message authentication or encryption. Previously such operations could not be carried out by class-3 devices. The remaining paper is organized as follows: An overview of the problem and our contribution is presented in section II. Next, related work on key derivation schemes for automotive protocols and hardware based identifiers are summarized in section III. Current threats for constrained devices together with current security specifications for vehicular on-board networks are presented in section IV. Following this, the detailed solution is described in section V. After that, the evaluation and the results are presented in section VI. Finally, section VII provides a brief conclusion and an outlook on further research.

II. PROBLEM OVERVIEW AND CONTRIBUTIONS

Most ECUs in the automotive network are class-2 and class-3 units with very limited computational power. Fig. 1 graphically represents a modern automotive network topology.

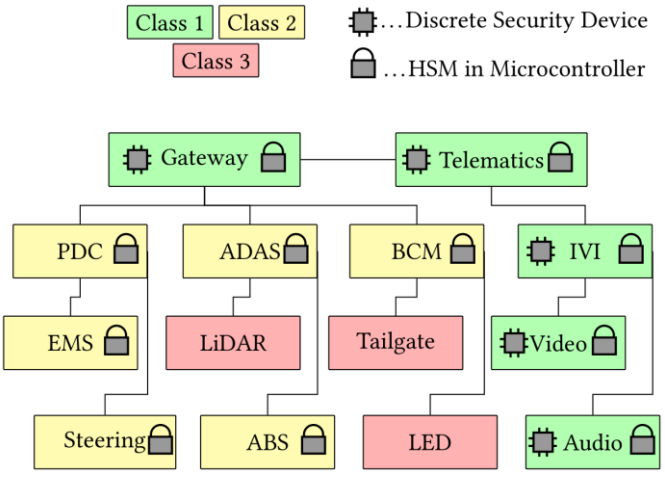

Figure 1. Hardware security features in selected control units arranged in a modern topology1

1 Abbreviations in Fig. 1: Power Domain Controller (PDC), Engine Management System (EMS), Advanced Driving Assistance System (ADAS), Body Control Module (BCM), and In-Vehicle Infotainment (IVI).

* *This resembles the full paper version of the demo paper submitted to the IEEE IRC 2019. This work was funded by the Austrian Research Promotion Agency (FFG, Project No. 862334)*

Even though class-3 devices may be limited in processing power they are often required to handle a large data throughput like transmitting data from LiDAR sensors or receiving animation patterns for LED-arrangements. In order to handle this data throughput conventional CAN frames with fixed payload sizes are no longer suitable and are being replaced by flexible data rate CAN (CAN FD) or automotive Ethernet.

Cryptographic hardware accelerators, which could be used to perform cryptographic operations for flexible payload sizes are too expensive for class-3 devices. Security solutions integrated into the platforms processor firmware are a feasible option that does not increase the hardware cost in terms of chip area or additional components. However, in a constrained processing environment classical cryptographic operations cannot be applied as they would introduce too much processing overhead. High processing loads must be prevented so that the device's actual application task is not impaired.

A. *Contributions*

This paper presents a firmware based key derivation scheme, which is used for device authentication of class-3 devices without requiring additional hardware. The authors propose a novel key derivation scheme for generating a class-3 device's secret key using unique device parameters only accessible during the primary firmware flashing procedure. The novel derivation scheme uses a peer reviewed cipher for the key derivation which is well suitable for firmware implementation of class-3 devices. The authors are not aware of any publications using device unique parameters with a lightweight cryptographic primitive to generate a secret key for class-3 control units in the way presented in section 5. This security feature increases the effort for adversaries to introduce stolen devices into vehicles without knowing the device unique secret. The key derivation algorithm has been assessed with the NIST statistical test suite to attest for the randomness of the generated key. An evaluation was carried out on an automotive grade processing core with a CAN FD interface to demonstrate its practical applicability. The contribution does not intend to propose yet another communication authentication protocol for class-3 devices. The solution was assessed according to existing threat scenarios and automotive requirements and will be discussed in the following section.

III. State-of-the-Art and Related Work

A. *Key Derivation Schemes in the Automotive Domain*

The authors reviewed available key derivation methods for CAN authentication approaches. This revealed that all reviewed publications *assume* the devices secret key as given, but none propose how to derive this key [12-14, 18-20].

The lightweight authentication proposal by Mundhenk et al. [20] provides a solution for applying authentication and covers the entire product life-cycle of an automotive device. However, when describing the setup phase they require a parameter, which must be unique to each device without stating how to generate it. In the broadcast authentication protocol LiBra-CAN, Groza et al. [12] explicitly state that for automotive security not another key exchange protocol is required, but a solution that can be integrated into real-world devices by a wide range of manufacturers. Our contribution utilizes a unique parameter that every silicon vendor currently already integrates into all processing units for class-3 devices.

Herrewege et al. [14] state that devices of the same type should contain individual keys to prevent one security breach from affecting a whole product group that uses the same secret key. We propose a method for each individual device to be equipped with a locally unique key counteracting this threat.

Other approaches not relying on unique keys exist and are suitable for class-1 and class-2 devices. However, processing constraints for class-3 devices do not allow resource intense operations such as the clock skew evaluation proposed by Cho et al. [5]. Cost constraints as well as security issues have made approaches with additional monitoring nodes unfavorable, like those proposed by Kurachi et al. [15].

Evaluations carried out by Dinu et al. [6] have shown that even the smallest software implementations of standardized AES are too expensive in terms of code size and RAM usage to be introduced to a class-3 device. Lightweight cryptography can be implemented in firmware and according to Sadhukan et al. [22] some algorithms such as the SPECK cipher are comparable to AES in terms of time-, space- and data complexities to mount an attack. This makes lightweight approaches attractive for addressing our problem.

B. *Standards on Lightweight Cryptography*

The ISO 29192 is a security standard that describes currently accepted lightweight cryptography algorithms for security features industrial applications. In part 6 of the ISO 29192, lightweight algorithms for message authentication codes (MACs) are described [25]. Currently three algorithms: are standardized for lightweight message authentication.

LightMAC has proven to be a feasible primitive for generating a MAC using the PRESENT algorithm in AES mode as Luykx et al. [26] demonstrated. *Tsudik's keymode* [27] relies on a hash function, which delivers a cryptographically suitable MAC, however the low data throughput was not acceptable for the application targeted by this solution. In 2015 Mouha proposed an update to the ARX based *Chaskey* algorithm [28] called *Chaskey-12*. ARX algorithms consist of modular addition, bit-field rotations and exclusive OR operations. Using an ARX algorithm rather than a hashing function for the generating MACs enable a higher data throughput. According to this reasoning, ARX based algorithms were preferred for the application in our solution.

While the *Chaskey-12* algorithm is standardized, other ARX-algorithms exist which have undergone more security evaluations than the *Chaskey-12* algorithm. Pfeiffer [19] suggests the SPECK cipher which is an ARX-based algorithm for implementing scalable CAN-bus security and demonstrates a holistic approach for authentication and encryption of messages on CANopen, SAE J1939 and CAN-FD bus communications. The scalability of the SPECK cipher results from the variation in plain and key sizes, which the SPECK cipher can be applied to. This allows an increase in security if

stronger hardware is available or the key length requires to be increased.

The SPECK algorithm offers options to increase the plaintext to cipher text diffusion, which is not available in *LightMAC* or *Chaskey-12*. According to Velichkov et al. [29], the exclusive OR operation provides diffusion between different words and linearity. On the other hand, modular addition provides nonlinearity and diffusion in single words. By increasing the diffusion with exclusive OR and modular addition operations the differential probability required for differential attacks can be positively influenced. In a recent amendment to the SPECK cipher Beaulieu et al. [3] suggested a variety of input parameters for the SPECK algorithm. We evaluated these parameter choices for their impact on the randomness of the cipher text output and in section VI.

The SPECK algorithm's code can be re-used, since the round key derivation function also relies on the cryptographic primitive that leads to a reduction in code size which is a limiting factor in our target hardware.

Dinu et al. [6] evaluated the 13 most relevant lightweight block ciphers on three embedded platforms (8, 16 and 32 bit) according to their code size, the RAM usage and their execution times. SPECK performed best due to the fact that its cryptographic primitive is used for the round ciphering function as well as the round-key derivation procedure.

The SPECK cipher is a round based ARX-based Feistel construction published by the NSA [2]. Its code size is smaller than other block ciphers because the primitive building block is used to perform the round key generation and the ciphering operation. Fig. 2 displays the SPECK cipher indicating the re-usability of the primitive building block. *L* and *R* are *n/2* bits of the left and right sections from the *n*-bit plain- and key text.

The best known attack on the SPECK-128 block cipher was published by Dinur [7], is a chosen plain text differential attack on 17 of 32 rounds of SPECK-128. This is comparable to the best impossible differential attack on 7 out of 10 rounds of AES [16] having similar complexities in data, space and time.

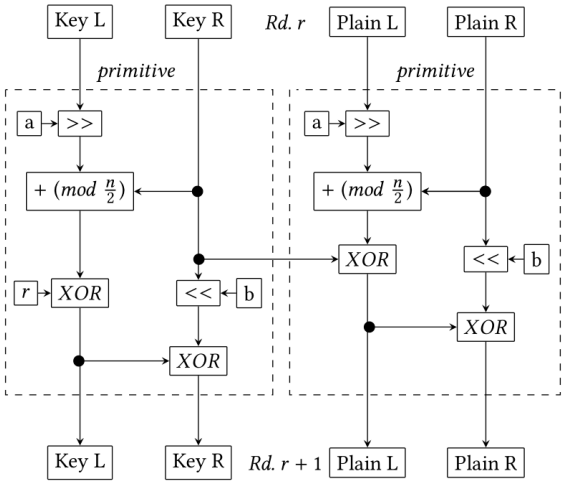

Figure 2. A single round r of the SPECK cipher for n-bit input, a=8 and b=3 in the original NSA publication [2]

IV. Security Requirements

Known threats to resource-constrained devices published by the Trusted Computing Group (TCG) [10] as well as the AUTOSAR guidelines for secure on-board communication [1] will be used to assess risk scenarios related to our solution.

A. Securing Resource Constrained Devices

As main threats to the security of constrained devices, which are also present in robotic applications, the TCG identifies: hardware tampering, algorithm subversion, access to concealed data, device impersonation and malware [10].

B. Guidelines for On-board Communication Security

In order to encourage compatibility of automotive devices the worldwide development partnership of AUTomotive Open System ARchitecture (AUTOSAR) has established standardized ECU software architectures.

In AUTOSARs latest specification on secure on-board communication (v4.3.1, [1]), the usage of symmetric authentication mechanisms using MACs to perform authentication of *entities* as well as of *data* is suggested. The specification proposes CMAC based on the AES cipher with an adequate key length as an example of how the data and component authentication should be achieved in devices with constrained processing capabilities.

C. Statistical Evaluation of Random Number Generators

When deriving a key, the generated output should be highly independent from the input data set it was generated from. This was assessed using the NIST statistical test suite. Explanations of each individual test were reported by Rukhin et al. [21]. While more sophisticated statistical test suites are available, the NIST test suite was originally applied to benchmark AES. Hence, it was used to evaluate if SPECK delvers random values comparable to results from the AES cipher.

The NIST test suite applies a hypothesis testing approach in which the H_0 hypothesis is that the Input bit sequence is random. During the evaluation, two errors can occur: Type I errors occur when the hypothesis is true but is rejected by the test. Type II errors occur when the hypothesis is false but is accepted.

In previous evaluations of the SPECK cipher Chew et al. [4] came to the conclusion that the SPECK cipher contains some nonrandom anomalies. They aimed to reduce Type I errors by setting a low significance level of $\alpha=0.001$. Sys et al [24] published an interpretation guide for statistical test suite results obtained from cryptographic functions. Sys et al. argue that by using a low confidence level for cryptographic functions is misleading since this causes more type II errors.

Type II errors are more severe to cryptographic functions since more generators are accepted with questionable randomness. Sys et al. recommend using higher confidence levels and analyzing failed cases with more data sets to receive results about its randomness. This is considered in the evaluation section VI.

V. SOLUTION AND IMPLEMENTATION

A. Secret Key Generation Schemes

The end-of-line flashing procedure where all ECUs are equipped with their device firmware is very time consuming. By introducing a symmetric technique a secret key must be provided to the communicating devices. To generate a secret key, a random number generator is required. The first choice for a random number generator would be a source of true randomness, which the vehicle manufacturer controls. This would require additional external hardware for the key generation scheme and would increase the duration of the end-of-line flashing procedure.

Block ciphers parametrized with random seed values are acceptable random number generators, e.g. CTR-mode [8]. By including a block cipher into the device's firmware, it can be used as a key generator and can be used to authenticate the device, verify message authentication codes or even perform encryption. An example of a message authentication scheme, using the SPECK block cipher was published by Pfeiffer [19].

By using the SPECK primitive for both the ciphering and the internal round key scheduling, the authors claim that this cipher is most suitable for deriving secret keys. The secret key generation process should not force the vehicle manufacturer to maintain additional hardware equipment for the end-of-line flashing or a secret key database facility. By performing the secret key generation in each class-3 device *internally* during the end-of-line flashing, we argue that a minimal time overhead is introduced in the key generation procedure and the key exchange between the class-2 devices that the target hardware communicates with.

B. Selecting Input Parameters for the Key Generation Schme

Other than the key derivation scheme proposed by Graunke [9], this solution does not require the device manufacturer to request an extra step in the secret key derivation and also does not require the semiconductor vendor to be directly involved in the key generation procedure. Since the detailed construction of the unique chip identifier (UCI) is part of the production process and thus may differ amongst chip manufacturers, we will define format that contains the parameters published by Graunke in order to explain our solution. Our UCI consists of three 16 bit values which are: the product identifier *PID*, a wafer number *W*, a serial number *S* and two 8 bit location-coordinates of the chip on the wafer[2] *X,Y* as follows:

$$UCI = [\, PID_{16b} \,||\, W_{16b} \,||\, S_{16b} \,||\, X_{8b} \,||\, Y_{8b}\,] \quad (2)$$

All automotive qualified silicon components contain this unique manufacturing information, which is only accessible to the processing core of newly manufactured components. The access to the UCI is physically disabled once the device manufacturer places the component in his device.

We propose to disable the UCI in a later stage, after the vehicle manufacturer has introduced his desired OTP parameters for the device in the final end-of-line flashing step. Other than the key derivation scheme presented by Graunke, our solution uses parts of this unique information as a NONCE input since no produced chip can contain this information twice. The product identifier is unusable for this, since it is constant for all devices.

The UCI contains less than the minimum key size of initial entropy making the UCI alone unsuitable to be used as an input to the block cipher to derive the secret key. In order to add additional entropy to the key derivation process, we propose to assign true NONCEs N^X_{8b} to half of the bits from the *PID*, *W* and S, as indicated below:

$$UCI=[PID_{8b} \,||\, N^P_{8b} \,||\, W_{8b} \,||\, N^W_{8b} \,||\, S_{8b} \,||\, N^S_{8b} \,||\, X_{8b} \,||Y_{8b}] \quad (1)$$

By adding this extra entropy, the unique nature of the UCI is not harmed and no additional chip area is required. A drawback of this approach is that the amount of nonarbitrary digits in *PID*, *W* and S are reduced. If the NONCEs are reset properly, neither the amount of possible UCIs, nor is the uniqueness of each UCI is reduced by this step.

At the vehicle manufacturing site where the devices are fitted to the vehicle, there is a high probability that vehicles produced after one another contain devices with chips from the same wafer. This correlation might be exploitable using the vehicle identification number (VIN), which is located at a spot that can easily be read from outside a locked vehicle. Decoding the VIN is a popular method amongst hardware thieves, to obtain information if the enclosed devices are compatible with the vehicles that their "clients" have. Hence, the only value from the UCI that contains a sufficient degree of uniqueness is the serial number *S* and the wafer location *X* and *Y*.

After functional testing and separation from the wafer the chips, the position information is no longer follows a deterministic pattern. A deductible relation between the wafer location information and the VIN, introduced at the vehicle manufacturing site, is highly unlikely. If the assumption can be made that the correlation of the wafer location information to the VIN is sufficiently low, then the wafer location information and the serial number can be used as a NONCE.

The key input for the used block cipher must be constant for all devices for a unique output. In symmetrical encryption a cipher text *c* is produced using an encryption function *ENC(.)* with plain text p and secret key *k* as arguments. If two different plain texts $p1 \neq p2$ are encrypted with the same key *k* it is an impossible event for two cipher texts to be identical:

$$ENC(p_1 , k) \neq ENC(p_2 , k). \quad (3)$$

With two different key values $k_1 \neq k_2$ and two different plain texts $p_1 \neq p_2$ a probability exists that the produced cipher texts are identical:

$$ENC(p_1 , k_1) = ENC(p_2 , k_2). \quad (4)$$

The NONCE enhanced product identifier $[PID_{8b} \,||\, N^P_{8b}]$ and the wafer number $[W_{8b} \,||\, N^W_{8b}]$ are *constant* for all components of a single wafer; however the serial number $[S_{8b} \,||\, N^S_{8b}]$ and location coordinates *X, Y* are individual and *vary* for each device on one wafer. We propose to use the *PID* and *W* to obtain a *constant* key input for the derivation scheme to be

[2] *A wafer with a diameter of 300 mm containing products with a die area of 6 mm² can yield up to ~10000 chips.*

bijective. *S*, *X* and *Y* will be used as *varying* plain inputs for the derivation scheme. Using these key and the plain text inputs, the values U_k and U_p are used to select an adequate amount of input bits from the OTP parameters introduced by the vehicle manufacturer to parametrize the class-3 device. This can be achieved without losing the local uniqueness of the key input data per device on the same wafer.

The plain text size S_{plain} is the input size required by the encryption function, reduced by the length of U_p. S_{plain} bits are selected from the least significant side of OTP memory. They are assigned to D_p. The remaining OTP Memory bits are assigned to D_k . The value U_p is concatenated with D_p to obtain the wafer unique plain input P_{OTP}[3]:

$$P_{OTP} = [U_P \ || \ D_P \ ]. \quad (5)$$

Next, the range in the OTP memory from where the key value K_{OTP} can be selected from, is calculated. This range is defined as the size of D_k reduced by the size of the secret key[4]:

$$R = sizeof(D_k) - S_{key}. \quad (6)$$

Finally, the offset O_k , required to extract the key input from D_k is determined. O_k resembles the bit-index from which bit onwards the key input is fetched from D_k. The bit index offset O_k is calculated using U_p and R as follows:

$$O_k = U_k \ \bmod R. \quad (7)$$

The modulo operation is applied in order to reduce the value of O_k without truncating the information content of U_k. At bit offset position O_k , a bit stream of length S_{key} is extracted from D_k , which is used as the key input K_{OTP}[5]. The device secret key K_S is then derived by applying an encryption function *ENC(.)* to P_{OTP} and K_{OTP} as follows:

$$K_s = ENC(P_{OTP}, K_{OTP}). \quad (8)$$

C. Selected Demonstration Hardware

As representative class-3 device an ASIC based on an *ARM Cortex M3* core, 32 kB of Program memory, 6 kB of RAM and proposed 1.5 kB one time programmable (OTP) OEM memory was chosen. To communicate with other devices on the same communication bus a discrete external CAN-FD transceiver was placed on the board of the class-3 device. The 1,5 kB OEM parameters are configurable during the end-of-line flashing procedure in the assembly line.

The selected device is not equipped with cryptographic hardware accelerators. Every additional feature in hardware would increase the chip area and increase the cost of the component. Furthermore, additional components would require automotive grade certification, which is not available for integrated cryptography blocks. An alternative solution to include security features to class-3 devices is to add firmware extensions. These firmware extensions must be kept as small as possible not to impair the actual application task of the ASIC. Due to these restrictions symmetrical techniques relying on a shared secret key seem to be the most suitable solution for class-3 devices.

The impact of all rotational constants and bit shift lengths published by Beaulieu et al. [3] could be analyzed for their effect on the randomness of the underlying SPECK primitive. Indirectly, the results in section VI can be interpreted as proof that larger rotational constants and shift lengths lead to more random output values. This might be caused by a positive influence of the differential probability pointed out by Velichkov et al. [29].

Fig. 4 displays the proposed key generation of a 128 bit key carried out by the hardware with the OEM parameter size stated above[6].

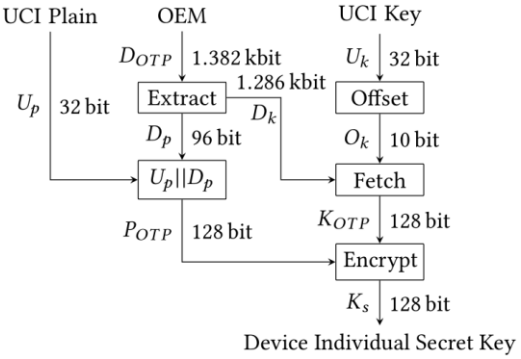

Figure 3. Applied example of the key derivation scheme

This approach allows the responsibility of the key derivation input information to be shared between silicon vendors and vehicle manufacturers, without requiring a direct exchange of information during the procedure.

D. Test Bench for Evaluating the Solution

The proposed solution was implemented on a class-3 device that is responsible for controlling a LED arrangement. The test bench setup consisted of a lab-PC that was attached to a USB to CAN FD converter for sending the communication bus commands and a demonstrator PCB, which received the commands. The external CAN FD transceiver on the PCB board forwards the information to the microcontroller. These commands are then forwarded to the LED arrangement on the demonstrator board as indicated in Fig. 4.

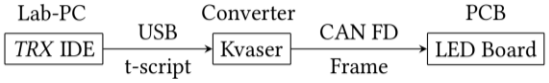

Figure 4. Test bench setup for the key derivation scheme

This setup was chosen since it allows an evaluation of the target hardware while connected to a communication bus, as it would occur in an automotive network. For the communication bus emulation a single channel CAN FD to USB interface was used (*Kvaser Leaf Pro)*. The application task of the board was

[3] *A unique value concatenated with an arbitrary value remains unique value.*

[4] *There must be sufficient data available to be extracted at the offset position.*

[5] *The fetch cannot exceed D_k since: $[sizeof(D_k) - S_{key}] \geq sizeof(O_k)$*

[6] *This is an example; the algorithm is applicable for arbitrary key sizes.*

to process incoming data in order to animate a LED arrangement. The firmware for the microcontroller was written in C-code in the IDE *µVision (V5.25.2.0)*.

E. Data Experiment Design

To compare the randomness of the cryptographic function to AES, Soto's [23] testing procedure was followed. The data sets required random initial values in order to be generated. These values were sourced from a Common Criteria EAL4+ certified true random number generator from a state-of-the-art smart card. A Java based query was designed to obtain 8 million 128 bit random values via a *Gemalto IDBridge* reader.

The nine data sets were produced using the proposed cipher implemented in *LabVIEW (V17.0f2)* using the generated initial values. The implementation on the µC and the LabVIEW implementation behave identically in terms of their input and output relations since the lab-pc was used to generate MAC messages emulating a class-2 communication partner which were all verified by the µC implementation.

The program code size and RAM usage of the proposed solution was measured using the GNU Development Tool: *arm-none-eabi-size (V7-2017-q4-major)* by measuring the firmware size and impact before and after the implementation.

For measuring the execution time of the proposed solution the IDE *µVision (V5.25.2.0)* was used in debugging mode. The cycles multiplied with the clock frequency equal the cycle time.

VI. EVALUATION

A. NIST Statistical Test Suite

For the statistical evaluation of the nine data sets the standard parameters of the improved test suite were used. Other than Dinu et al. [6] we agree with Sys et al. [24] and decided to test the data with $\alpha = 0{,}01$ to encounter less type II errors. The input data used for statistical testing is listed in Table I.

TABLE I. EVALUATED INPUT DATA SETS

	Data Set Sizes		
Unit	***SKA, SPA***	***PCC, CBC, RPRF***	***LDP, HDP, HDK***
MB	127	122	132
#	968	927	1007

The SPECK cipher uses the constants *(a, b)* for cyclic shifts. In a recent publication Beaulieu et al. [3] proposed 4 value pairs for these constants. They argue that the performance for 8 bit ASIC implementations decrease with larger bit distances between *a* and *b*. The same holds for constants that are far from integer multiples of eight.

So far these combinations have not been evaluated for their statistical impact on the randomness of the SPECK block cipher. We generated four cases of input data sets and analyzed the rotational constant pairs of (8,3), (7,2), (9,2) and (11,7).

The nine input data sets were generated for each of the cases using the test cases as described above. By performing the NIST statistical test suite on all four cases the randomness of each case was assessed by recording the pass rate in Table II.

TABLE II. STATISTICAL TEST SUITE RESULTS

Test Case	***Proportional Pass Rate in %***								
	SKA	***SPA***	***PCC***	***CBC***	***RPRF***	***LDP***	***HDP***	***LDK***	***HDK***
1	100	100	100	100	**98,9**	100	**99,5**	100	**99,5**
2	100	100	100	100	**98,9**	100	**99,5**	100	**99,5**
3	**99,5**	**99,5**	100	100	100	100	100	**99,5**	**98,9**
4	**99,5**	100	100	100	**99,5**	**99,5**	100	100	100
99±	**0,95**	**0,95**	**0,98**	**0,98**	**0,98**	**0,94**	**0,94**	**0,94**	**0,94**

Test case 4 has the largest value in rotation and shift length and produced the most random output when considering all nine data sets compared to the other test cases. All test cases passed the randomness test, however for the final implementation the constants from test case 4 were chosen. The results suggest that the SPECK cipher is random enough for generating secret keys with arbitrary input values.

B. Execution Time

The proposed solution requires 14625 processor cycles for the key derivation scheme. The time impact on the end-of-line flashing procedure for a class-3 device running at 40MHz would be 366µs per device. This penalty is tolerable, considering that the overhead of introducing a key from an external key derivation source would consume more time.

C. Security Assessment and Threat Coverage

Allowing the class-3 device to internally derive the secret key with OEM parameters is favorable, since no single party is in full control of all key derivation inputs.

If the final device firmware has been functionally tested and evaluated for customer release, then a logical attack to dump the memory during a cryptographic operation in order to extract the secret key should not be possible. A physical attack on the hardware would damage the device beyond usability and reduce its worth for an adversary. Adversaries would require a very large series of devices in order to extract a relation between the VID and the unique chip parameter. Stealing components with device individual keys becomes less attractive since the secret key is hard to guess. An adversary would be required to monitor and record all data communicated between the class-2 and class-3 device for a time span of several years in order to mount a guessing attack on the implemented SPECK-128 algorithm. This would only deliver the key of one single key, making this attack scenario unfeasible.

The proposed key derivation scheme can also be used to generate 256 bit keys that could be used for a symmetrical post-quantum computing 256 bit scheme with a suitable cipher.

This concludes the main threat analysis of the proposed solution. In the following paragraph the security of the solution will be evaluated according to specific security recommendations for resource constrained devices from the trusted computing group [10] as well as the AUTOSAR [1] Secure on-board recommendations.

1) Hardware Tampering

The evaluated class-3 device is designed in such a way that by opening the device enclosure, the LED arrangement is damaged reducing its economic value. The only noninvasive access that an adversary can have is by interfacing the device via its communication bus, which is beyond hardware tampering. Unfortunately class-3 devices located at exterior vehicle locations can be extracted and are subject to attacks with chosen hardware. For class-3 devices without protection, the value of the data stored in the devices is minimal. As soon as the secret key is introduced to class-3 devices the value of the data in the device increases since it can only be used if the secret key is known to communication partners.

2) Algorithm Subversion

The subversion of algorithms is identified as a threat that is achieved by brute force or replay attacks. The proposed solution supports authentication mechanisms using NONCEs by providing a secret key and a cryptographic function with which replay attacks can be prevented. An infeasible amount of data must be collected for a brute force attack to succeed. Generating device independent random numbers is possible since the UCI is individual for each device and even unique for devices from the same silicon wafer. This local uniqueness makes the chance of success for brute force attacks equal to the chance of guessing the correct key for a very large amount of consecutively produced devices.

3) Accessing Secret Data

The authors would like to remark that the UCI storage is equipped with a physical readout protection mechanism preventing the readout when placed inside the class-3 device. The processing engine for cryptographic operations class-3 devices cannot be isolated since additional circuitry would be necessary to accomplish this task. If the assumption can be made that the entire firmware was sufficiently tested for causing memory dumps then the stored secret key can be considered safe during operations in the core that needs to process the secret key. The key derivation uses the entire range of available OEM parameters for deriving the secret key. In the case that a memory dump can be caused, this would have to be achievable for all device functions in order for an adversary to obtain the full set of OEM parameters. This barrier increases the complexity of an attack for class-3 devices making such actions less attractive for adversaries.

4) Impersonating a Device

By concealing the devices secret during the storing action and when it is in use, the success rate of impersonating a device is reduced to the probability guessing the devices secret key. By introducing device individual keys, the impersonation of a device is made harder since if one key is disclosed it cannot be reused for other class-3 devices. The procedure in which the secret key is created could be carried out in a secure environment in order to prevent eavesdropping or manipulation

5) Malware

Secure boot procedures and firmware verification capable of mitigating malware cannot be provided with our solution.

6) Entity and Data Authentication

Through simple challenge response procedures the class-3 device is fully capable of authenticating itself to its communication partner. By means of *mutual authentication* the class-3 device can also verify if the class-2 device is allowed request authentication. When using the device unique key in combination with a *freshness-value* the communicated data can be authenticated by means of any accepted mode of operation.

7) Functional Specifications

The evaluated solution does not impair the application task of the device. The fact that the used cipher is not standardized, deviates from the AUTOSAR requirements. If in future such ciphers are included, class-3 devices could comply with the AUTOSAR secure on-board communication specification.

VII. Summary and Outlook

A. Key Derivation Scheme

The proposed key derivation scheme offers the capability of delivering device unique secret keys. By using a cryptographic function to derive the secret key, other operations such as data authentication and encryption are possible by reusing the firmware code. The key derivation scheme uses inputs both from the silicon device and OEM parameters. A downside of the proposed scheme is that it uses a non-standardized cipher.

The 128 bit SPECK cipher was chosen over the standardized ciphers, since the SPECK primitive and its round key derivation use the same primitive for the round key, making its overall code size smaller than standardized ciphers. Our solution is not limited to the SPECK cipher and can be applied by using any other comparable cipher with similar cryptographic properties and implementation size.

B. Statistical Evaluation

Through applying a statistical test suite to the underlying cryptographic function it could be empirically verified that the chosen ciphers output is comparable to AES. By analyzing recently proposed variations of the ciphering function, stronger diffusion and nonlinearity could be observed.

C. Performance Analysis

The implementation of the solution was assessed according to code size, RAM usage and execution time on an appropriate class-3 device. Our entire solution occupies less than 10% of the available resources. The numbers of processor cycles to perform the key derivation were measured and found to be acceptable compared to alternatives for deriving secret keys.

D. Security Considerations

By applying the proposed solution, the key is stored in a region of the memory that is separate from the program code. Since the derived key is individual per device, attacks on the device are expensive. By enforcing the inputs for the key derivation to obtained from the OEM as well as the silicon vendor, manipulations of the full input become very hard to achieve. One major deficiency of the evaluated solution is the lack of securely booting. Side channel and logical attacks

cannot be prevented by the proposed solution due to the lack of dedicated hardware. The firmware of the device must be carefully designed and well evaluated so that memory dumps can be excluded as an attack scenario. If adversaries can gain access to the area where the OEM parameters are stored the secret key can be stolen.

E. Outlook and Future Work

A Full round trip analysis of the implemented solution with a suitable authentication scheme and encryption capabilities is intended to assess the overhead introduced by the authentication procedure. In order to demonstrate the flexibility of the solution, a challenge response application with the target hardware will be set-up in future work. In this follow-up work the applicability of the SPECK cipher for device authentication and authenticated encryption with additional data will be demonstrated with the hardware used in this article. If a new device is being introduced to the vehicle and the OTP memory is already set, communication should be declined. With a mechanism to determine if the OTP memory has been set or not, broken devices can be replaced with genuine parts that can be flashed in a trusted environment.

F. Conclusion

In this paper we proposed a key derivation scheme that can generate device individual secret keys based on an internal hardware identifier. The procedure is applicable to low-end automotive and robotics devices that are not equipped with cryptographic hardware. The proposed solution relies on the randomness of a cryptographic function, which we statistically evaluated to demonstrate its capability of delivering random values. Our solution enables the generation of secret keys that are unique for every device from the same wafer. With our solution low-end control units are enabled to perform message authentication. Hardware theft of these components becomes unattractive since stolen device cannot be reintroduced to other vehicles or robots without unfeasible effort. The key derivation occurs inside the device itself by means of a procedure where neither the silicon vendor nor the device manufacturer is in full control of all input parameters. The applicability was evaluated on a low-end automotive device, in order to demonstrating its lightweight nature and the usability for resource constrained processing units. The solution was assessed not only according to AUTOSAR secure on-board requirements but also according to the TCG criteria for resource constrained devices to address automotive- and robotics requirements. If device manufacturers cannot provide hardware security features to their low-end devices due to economic limitations, our solution offers an innovative approach to enable sufficient security features that can be used for key derivation, hardware authentication and message authentication without limiting the execution capabilities of the device at hand.

REFERENCES

[1] AUTOSAR. 2017. Specification of Secure Onboard Communication. Technical Report.

[2] Ray Beaulieu et al . 2015. The SIMON and SPECK lightweight block ciphers. CM/EDAC/IEEE Design Automation Conference.

[3] Ray Beaulieu et al . 2018. Notes on the Design and Analysis of SIMON and SPECK. Cryptology ePrint Archive 2017/560 .

[4] Liyana Chew et al . 2015. Randomness Analysis on Speck Family of Lightweight Block Cipher. International Journal of Cryptology Research

[5] Kyong-Tak Cho et al. 2016. Fingerprinting Electronic Control Units for Vehicle Intrusion Detection. USENIX Security Symposium.

[6] Daniel Dinu et al . 2015. Triathlon of Lightweight Block Ciphers for the Internet of Things. NIST Workshop on Lightweight Cryptography.

[7] Itai Dinur. 2014. Improved Differential Cryptanalysis of Round-Reduced Speck. Cryptology ePrint Archive: Report 2014/320.

[8] Morris Dworkin. 2001. NIST Special Publication 800-38A. Technical Report. NIST.

[9] Gary Graunke. 2010. Apparatus and Method for distributing Private Keys to an Entity with Minimal Secret. Unique Information.

[10] "Trusted Computing Group". 2017. TCG Guidance for Securing Resource-Constrained Devices. Technical Report.

[11] "Trusted Computing Group". 2018. Hardware Requirements for a Device Identifier Composition Engine. Technical Report.

[12] Bogdan Groza et al . 2017. LiBrA-CAN: Lightweight Broadcast Authentication for Controller Area Networks. ACM Transactions on Embedded Computing Systems (TECS).

[13] Oliver Hartkopp and Roland Schilling. 2012. MaCAN - Message Authenticated CAN. ESCAR Conference.

[14] Anthony Herrewege et al . 2011. CANAuth - A Simple, Backward Compatible Broadcast Authentication Protocol for CAN bus. ECRYPT Workshop on Lightweight Cryptography.

[15] Ryo Kurachi, Yutaka Masubara, and Hiroaki Takada. 2014. CaCAN - Centralized Authentication System in CAN. ESCAR Europe.

[16] Hamid Mala et al. 2010. Improved Impossible Differential Cryptanalysis of 7-Round AES-128. INDOCRYPT.

[17] Dennis Mattoon. 2018. Establishing Unique Identity and Security for Cost-Sensitive Embedded Products. http://www.electronicdesign.com/embedded-revolution/establishing-unique-identity-and-security-cost-sensitive-embedded-products

[18] Stefan Nuernberger et al.. 2016. vatiCAN Vetted, Authenticated CAN BUS. Cryptographic Hardware and Embedded Systems.

[19] Olaf Pfeiffer. 2017. Implementing Scalable CAN Security with CANcrypt. Embedded Systems Academy.

[20] Artur Mrowca Philipp Mundhenk, AndrewPaverd. 2016. Security in Automotive Networks: Lightweight Authentication and Authorization. ACM Transactions on Design Automation of Electronic Systems.

[21] Andrew Rukhin et al . 2010. A Statistical Test Suite fo Random and Pseudorandom Number Generators for Cryptographic Applications. Technical Report. NIST.

[22] Rajat Sadhukan, Sikhar Patranabis, and Ashrujit Ghoshal. 2017. An Evaluation of Lightweight Block Ciphers for Resource-Constrained Applications: Area, Performance and Security. Journal of Hardware and Systems Security.

[23] Juan Soto. 1999. Randomness Testing of the Advanced Encryption Standard Candidate Algorithms. NIST.

[24] Marek Sys, Zdenek Riha, and Vashek Matyas. 2015. On the Interpretation of Results from the NIST statistical Test Suite. Romanian Journal of Information Science and Technology.

[25] ISO/IEC 2012. Information technology – Security techniques – Lightweight cryptography – Part 6: Message authentication codes (MACs). Industrial Standard.

[26] Atul Luykx et al. 2016. A MAC Mode for Lightweight Block Ciphers. FSE 2016: Fast Software Encryption. Springer.

[27] G. Tsudik. 1992. Message Authentication with One-Way Hash Functions. ACM SIGCOMM Computer Communication Review.

[28] N. Mouha. 2015. Chaskey: a MAC Algorithm for Microcontrollers – Status Update and Proposal of Chaskey-12 –. Cryptology ePrint Archive 2015/1182.

[29] V. Velichkov et al. 2011. The Additive Differential Probability of ARX. FSE: Fast Software Encryption. Springer.

[30] C. Valasek et al. 2014. Adventures in Automotive Networks and Control Units. White Paper. IOActive Comprehensive Information Security.