Enabling Secure Communication for Automotive Endpoint-ECUs through Lightweight-Cryptography

CANsec: Built-in Security for CAN XL

Friedrich Wiemer
friedrich.wiemer@de.bosch.com
Robert Bosch GmbH
Cross-Domain Computing Solutions
Advanced Network Solutions
Leonberg, Germany

Alexander Zeh
alexander.zeh@infineon.com
Infineon Technologies AG
Automotive Division
Research & Development
Munich, Germany

ABSTRACT

Enabling secure communication to and from endpoint-ECUs in automotive E/E architectures is crucial, as e.g. shown by recent attacks such as CAN injection. Cost-efficient and resource-saving in-vehicle solutions are currently missing. Emerging network technologies for upcoming zone-based architectures require bandwidths of 10 Mbit/s for nodes at the edge of the internal vehicle network.

The new security protocol *CANsec*, achieving Authenticated Encryption with Associated Data (AEAD) for CAN XL frames, aims to satisfy the new requirements. The industry encryption standard for AEAD is AES-GCM, the Advanced Encryption Standard used in the Galois Counter Mode. However, AES-GCM exhibits severe drawbacks when it comes to so-called nonce misuses. In this paper, we study an alternative cipher suite for automotive in-vehicle networks with a focus on two properties.

First, to allow applications in resource-constrained endpoint-ECUs in automotive networks to additionally execute CANsec, we propose an alternative solution to AES: the lightweight algorithm Ascon.

Second, the nonce misuse behaviour of Ascon in the particular application of CANsec should improve on the AES-GCM case. Here, we compare already known attacks and their implications for the different choices of cipher suites. In particular, we look at GCM decryption and forgery attacks, as well as at decryption and forgery attacks on generic sponge constructions. Besides these attacks, we also analyse the behaviour of AES-GCM-SIV and Ascon with respect to nonce misuses.

We conclude the study by suggesting Ascon as an additional, optional cipher suite for CANsec.

CCS CONCEPTS

• **Security and privacy** → **Security protocols**; *Symmetric cryptography and hash functions*; • **Networks** → *Link-layer protocols*; • **Computer systems organization** → Embedded systems.

CSCS '23, December 05, 2023, Darmstadt, Germany

ACM ISBN 979-8-4007-0454-3/23/12.
https://doi.org/10.1145/3631204.3631861

KEYWORDS

AES-GCM, AES-GCM-SIV, Ascon, CANsec, CAN XL, Lightweight Cryptography, Nonce Misuse Resistance, Secure In-Vehicle Communication

ACM Reference Format:
Friedrich Wiemer and Alexander Zeh. 2023. Enabling Secure Communication for Automotive Endpoint-ECUs through Lightweight-Cryptography: CANsec: Built-in Security for CAN XL. In *Computer Science in Cars Symposium (CSCS '23), December 05, 2023, Darmstadt, Germany*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3631204.3631861

1 INTRODUCTION

We are right in the middle of multiple revolutionizing developments in the automotive domain. Next-generation road vehicles will offer ***P**ersonalized* experiences to the passengers, ***A**utomated* driving features will have a huge impact on the way we operate our vehicles, enable higher ***C**onnectivity* to integrate personal devices such as smartphones, all combined with an ***E**lectrification* of the drive train. These PACE changes require as a foundation market-shaping trends like software-defined vehicles and zone-based E/E architectures within the vehicles. Besides the increased need for security due to the ongoing digitalization of vehicles, regulatory requirements for their security increase as well, e.g. by the UNECE regulation R155 [30] coming into effect.

Zone-based in-vehicle E/E architectures require increased bandwidth due to the above-mentioned PACE changes. For example, multi-gigabit Ethernet communication on the backbone between vehicle computers and zone controllers within the vehicle will be used on the roads in a few years. Furthermore, the in-vehicle communication bandwidth between sensors and zone controllers has to increase from one to two Mbit/s, which is achieved by legacy bus systems such as Classic CAN [7] or CAN FD [8].

Driven by the need for a cost-efficient solution to fill the bandwidth gap between Classic CAN / CAN FD and Automotive Ethernet, typically achieving 100 Mbit/s (100Base-T1) and more, a group of industry partners started to specify CAN XL under the CAN in Automation (CiA) umbrella. CAN XL builds on the advantages of Classic CAN and CAN FD and reaches bandwidths of 10 Mbit/s up to 20 Mbit/s. An alternative solution to CAN XL is Automotive Ethernet 10Base-T1S, achieving 10 Mbit/s as well.

Note that for Classic CAN and CAN FD a native security protocol was not considered. AUTOSAR's Secure Onboard Communication (SecOC, [2]) is widely used in the automotive domain. However, the

restrictions of Classic CAN for a security protocol are quite severe. Besides this, SecOC has its drawbacks, e.g., it does not support encrypted CAN frames and the specification leaves out important details such as the freshness handling or a mechanism to rotate / agree on new (session-) keys. These drawbacks result in different proprietary extensions by different OEMs, making it hard to, e.g., offer one single solution for SecOC.

One major advantage of SecOC on the other hand is its applicability for end-to-end (E2E) security scenarios. A typical implementation of SecOC-protected CAN frames in a vehicle's E/E architecture applies the frame authentication at the initial sender of the frame and verifies the authenticity of the received frame at the final receiver. This not only protects the message from modifications by malicious gateway nodes in between the sender and receiver. Much more important, E2E protection also enables the vehicle's architect to apply security on the functional level of the E/E architecture, without the need to worry about possible modifications at the decomposed technical level with possible intermediate nodes in the network added.

Different to Classic CAN and CAN FD, a security protocol for CAN XL, termed *CANsec*, is developed in parallel and right from the start, to avoid the SecOC drawbacks, while maintaining its advantages.

CANsec operates on Layer 2 of the ISO/OSI communication model and is thus comparable to MACsec [21], a Layer 2 security protocol for Ethernet. Due to its availability, the CANsec design is heavily orientated by MACsec. However – without the need for the big versatility that MACsec has to fulfil (for all possible different flavours of Ethernet) and the "legacy" of over 15 years of MACsec – CANsec jumps at the chance to also improve over its quasi-ancestor.

One particular possibility for improvement is the choice of cipher suites offered in the protocol. MACsec uses the standard AES-GCM with two different key sizes (128- and 256-bit) as well as two different freshness value sizes (32- and 64-bit). While AES-GCM, as the industry de-facto standard for Authenticated Encryption with Associated Data (AEAD), is definitely one good choice and should probably be available as a cipher suite, GCM does also have strong security requirements for its super-system (CANsec or MACsec):

> The "Number used only ONCE" (Nonce) input should be unique for every encryption call (under the same key).

This may sound trivial (in theory), but is a real problem (in practice) and has led to real-world vulnerabilities, see e.g. [9].

The fundamental problem of GCM in this so-called nonce misuse scenario is the catastrophic security reduction for repeated nonces. We recall the relevant attacks in Section 3.

However, the cryptographic community has realized this problem and focused on developing alternatives. Examples of these endeavours are the CAESAR and NIST LWC competitions, on which we give more details later. A very recent announcement in this regard by NIST[1] specifically announces a workshop on this topic and says

> The goal of the workshop is to discuss how NIST can best address the limitations of the block cipher modes of operation that are approved in the NIST Special Publication 800-38 series, and the possibility of standardizing a tweakable wide block encryption technique that could support a large range of input lengths.

Finally, one could argue that, instead of fixing the problem of nonce misuses by choosing a more resistant mode of operation or algorithm, one could also solve the problem by designing the system to make nonce misuses harder in the first place. The problem we see with this (the second approach) and why we argue to do *both* is, that from a standardization point of view, it is harder (but not impossible) to specify the nonce generation part. There can always be vulnerabilities in the design of the system that allow an attacker to enforce a nonce-misuse. For such a case, the security architecture should apply a *defence-in-depth* approach and not only rely on a single security measure, to be more robust in general. We thus suggest approaching the problem of nonce misuses from both sides: first, use a cipher suite that is less reliant on the nonce property and second, write the specification in such a way that nonce misuses will be as hard as possible to provoke.

Besides the aim of providing better security properties, CANsec also borrows from SecOC's E2E protection possibilities. To fully exploit these E2E possibilities, the necessary cryptographic operations need to be available on the sender and receiver side. While this does not pose a problem for the very performant central vehicle computers, the situation looks different for possibly very resource-constrained endpoint-ECUs, such as sensors or actuators.

Therefore, we also take into consideration another research area in the last years, i.e., *lightweight cryptography (LWC)*. The goal of LWC is to develop specifically optimized algorithms under specific performance criteria (e.g., hardware gate count, encryption latency, power consumption), to design domain-specific algorithms. Since the first LWC ciphers, such as Noekeon [14] and Present [25], several designs were published, many of them outperforming AES significantly, see the discussed performance figures in the next section. We aim to analyse and propose better candidates (more secure/resistant and more performant) than AES-GCM for CANsec.

Contribution. Our contribution in this paper is twofold. On the technical side, we analyse the nonce misuse behaviour of three different cipher suite candidates for CANsec:

- AES-GCM [28],
- AES-GCM-SIV [17], as well as
- Ascon [15].

For this, we specifically focus our analysis on the use case for CAN XL in CANsec. This focus allows us to recommend an optimized scheme without sacrificing security.

We combine our practical analysis of the theoretical security bounds with practical attacks on the three cipher suites. Our results show, that the lightweight cipher option Ascon provides the best security guarantees for misused nonces in the authentication-only scenario – which remains the most important one for automotive in-vehicle communication.

This result, in combination with Ascon's performance benefits, emphasizes the suitability of Ascon as the cipher suite for CANsec.

[1] See https://groups.google.com/a/list.nist.gov/g/lwc-forum/c/21vrX8fs3eY.

Our second contribution is educational: we make the cryptanalysis results and nonce misuse risks accessible to the automotive community. For this, we cover the relevant results in a self-contained way and favour understandability over brevity when it comes to the description of the cryptanalysis parts.

Organization. Section 2 covers the background for our work, namely the emerging bus technology CAN XL and the corresponding security protocol CANsec. To enable easier following the parts of our analysis, we decided to introduce the necessary background on the analysed cipher suites in the corresponding later sections.

In Section 3, we analyze the impact of nonce repetitions, i.e., misuses, for cipher suites which are relevant to the CANsec specification. The paper concludes in Section 4 by summarizing the impact of the previous attacks on CANsec.

2 BACKGROUND

CAN XL is developed to be backwards compatible with Classic CAN [7, 22] and CAN FD [8, 20] in the sense that it can be used on the same bus in mixed applications, i.e., some nodes on the bus speak CAN XL while some nodes still speak Classic CAN or CAN FD. We do not cover CAN XL in full detail here but concentrate on the important aspects of the remaining part of this work.

2.1 CANsec

CANsec is a working draft by the CiA [12]. It is inspired by MACsec for Ethernet [21] while aiming to improve over it when possible. Due to MACsec's need for versatility (Ethernet point-to-point and multidrop bus systems) and legacy parts, CANsec is a specifically tailored security protocol for CAN XL that can step ahead and provide advanced security properties.

In general, CANsec enables authenticated-only or authenticated-encrypted communication for a subset of CAN XL bus nodes. For this, the CAN XL bus is separated into Secure Zones (SZs). Any participant in an SZ authenticates itself by being in possession of the Secure Zone Key (SZK). The SZK acts as a long-term secret key, which is used to derive session keys through a control plane protocol. The nodes within an SZ communicate through unidirectional Secure Channels (SCs), which consist of up to two Secure Associations (SAs). Each SA manages a Secure Association Key (SAK; the session keys), and a Freshness Value (FV). As the nodes are communicating via a multidrop bus system, every node within an SZ uses one transmitting SC that all other nodes use as a receiving SC. Multiple SAs are used, to allow a seamless change between session keys, once the old session key expires.

The expiration of session keys depends on the bus load as well as the freshness value, i.e., a session key expires because the set of "fresh" freshness values empties. On a CAN XL bus, the load depends on the time a frame needs for transmission. Due to distinguishing between arbitration and data phase, the transmission time itself depends on

- the *arbitration phase* with a corresponding bit rate d_a,
- the (minimal) arbitration frame length ℓ_a, i.e., the sum of the frame field sizes that are transmitted in the arbitration phase,
- the *data phase* with a corresponding bit rate d_d, and
- the (minimal) data frame length ℓ_d, i.e., the sum of the frame field sizes that are transmitted in the data phase.

We give estimates of these values in the following section. Then, we assume the expiration time T of an SAK to be greater than:

$$T \geqslant T_n \stackrel{\text{def}}{=} 2^n \cdot \left(\frac{\ell_a}{d_a} + \frac{\ell_d}{d_d} \right), \tag{1}$$

where $\ell_a/d_a + \ell_d/d_d$ is the transmission time of a CAN XL frame of minimal length in seconds. For readiblity, we later scale T to years. To lower-bound the expiration time we assume minimal length CAN XL frames to be sent 2^n times,[2] where n is the bit length of the freshness value. Subsequently, we use this lower bound T_n, as defined in Eq. (1), to discuss the implications of different nonce misuse scenarios.

2.2 Estimating CAN XL Bus Utilization

Due to the backward compatibility of CAN XL to Classic CAN and CAN FD, the arbitration phase uses the same bit-rate, which can be up to $d_a = 1\,\text{Mbit/s}$.[3] The switch between the (slower) arbitration phase and the (faster) data phase happens after transmission of the "Arbitration to Data High" (ADH) field as part of the "Arbitration to Data Sequence" (ADS). The acknowledge field and End Of Frame (EOF) field in the CAN XL footer are eventually also sent as part of the arbitration phase. Details on the CAN XL frame format can be found in [13, Figure 2]. Figure 1 shows the frame format of a CANsec-protected CAN XL frame. The arbitration and data phases are colour-coded.

From the figure, we can determine a minimal length for the arbitration phase of $\ell_a = 31\,\text{bit}$. After the switch, during the data phase, information is sent as "fast bits", resulting in data bit-rates of 10 Mbit/s up to 20 Mbit/s (the exact bit-rate depends on the actual system configuration). We assume $d_d = 20\,\text{Mbit/s}$. Similar to the arbitration phase, we can lower-bound the length of the data phase, which is dynamic due to the LLC data frame field being zero up to 2048 byte long, see again Figure 1.

We thus end up with the length of 115 bit for the CAN XL frame, excluding the data field. The minimal length of the data field depends on the CANsec frame.

CANsec-protected CAN XL frames are indicated through the Single Extended Content (SEC) field, which is set in case additional functionalities are included in the data field. In case the SEC field is set, the data field starts with an AddOn Type (AOT) field, which specifies the particular additional functionality. An AOT value of $\texttt{010}_b$ identifies a CANsec frame. All CANsec fields can again be found in Figure 1.

These fields add up to 128 bit and are all included in the CAN XL data field. Note that different additional functionalities can be combined in one CAN XL frame, resulting in the necessity to have an additional SEC field in the CANsec PCI. The minimal length of the CANsec service data unit (the payload / data field of the CANsec frame) is eight bits. The overall resulting minimal length

[2] Note that CANsec uses a monotonic counter for the freshness value. In case random numbers would instead be used, a birthday-style bound on collisions should be taken into account, leading to another factor of two, i.e. 2^{n-1} instead.

[3] See https://www.can-cia.org/can-knowledge/can/can-xl/ for numbers on the arbitration and data phase rates.

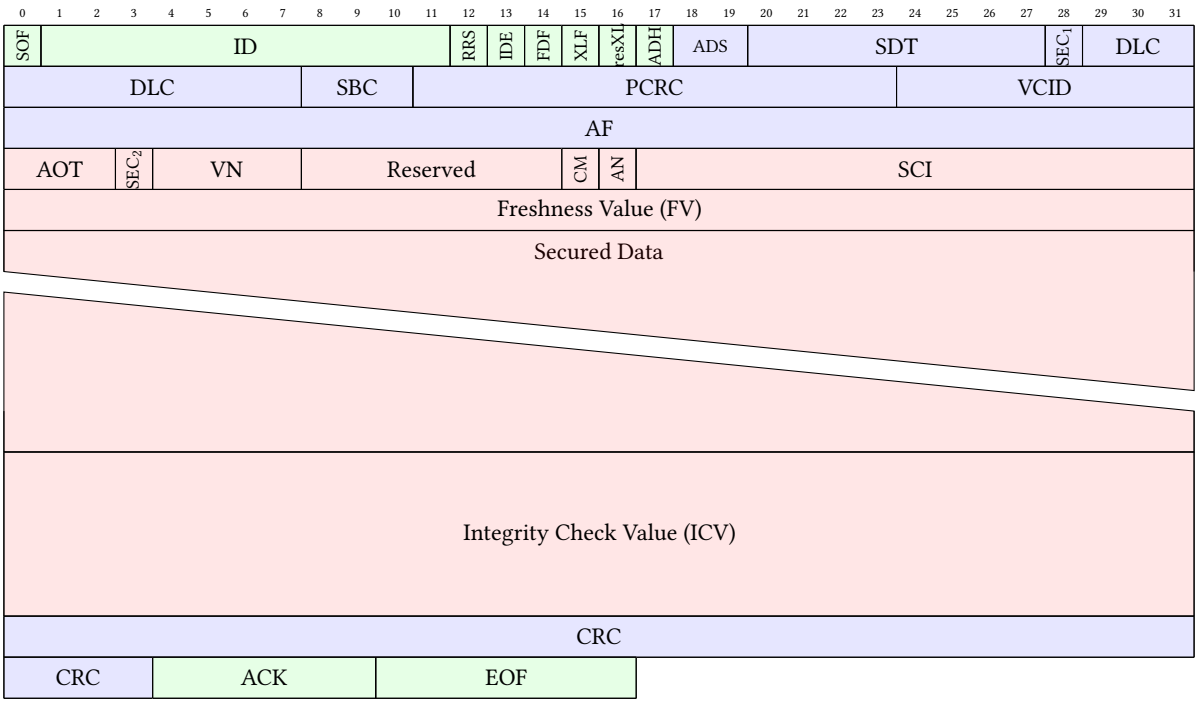

Figure 1: Fields of a CANsec-protected CAN XL frame. Colour coding indicates the sending mode for the CAN XL part (CAN XL part in arbitration phase , CAN XL part in data phase) as well as the CANsec part , which is sent in the data phase , too.

of the data phase for a CANsec-protected CAN XL frame is $\ell_d = 115 + 128 + 8 = 251$ bit.

In summary, we have

$$d_a = 1\,\text{Mbit/s}, \quad d_d = 20\,\text{Mbit/s}, \\ \ell_a = 31\,\text{bit}, \text{ and } \quad \ell_d = 251\,\text{bit}, \tag{2}$$

resulting in $T_n = 2^n \cdot 43.55\mu s$.

2.3 Algorithms under Test

For our analysis, we choose three different cipher suites: AES-GCM, AES-GCM-SIV and Ascon. The first, AES-GCM [28], is the de facto standard for AEAD modes in the automotive domain. This is mainly due to its predominant availability in hardware accelerators of automotive micro-controllers and system-on-chips. We choose the second, AES-GCM-SIV [17], as it is primarily designed to avoid the nonce misuse weakness of GCM and provides a provable nonce misuse resistance. However, unfortunately, AES-GCM-SIV cannot make use of the available AES hardware accelerators in today's automotive chips. We thus decided to take the chance and improve over the AES, by taking into account a lightweight algorithm. While there is a wide range of available lightweight designs, a natural choice is to take the recently (beginning of this year) announced winner of NIST's lightweight cryptography competition (LWC),[4] namely Ascon [15].

Ascon, offering mainly 128-bit keys, shows very good security properties with respect to nonce misuse scenarios, even if it does not achieve provable misuse resistance as AES-GCM-SIV does. We cover the details in Section 3.3. Furthermore, Ascon also exhibits a better performance than AES-GCM. For hardware implementations, see for example [1, Table 3] (ASIC benchmarks) and [27] (FPGA benchmarks). In particular, comparing implementations of AES-GCM and Ascon that achieve roughly the same throughput, [1, Table 3], i.e. implementations number 9 (Ascon, throughput of 2 523.4 Mbit/s) and 43 (AES-GCM, throughput of 2 455.3 Mbit/s), we can see that Ascon outperforms AES-GCM by a factor of more than 2.5 in area consumption. In other words, Ascon hardware implementations with the same throughput require only 40% of the resources of an AES-GCM hardware implementation. Similar rations can be found for FPGA implementations in [27]. For software implementations, NIST's status report on the second round of the LWC gives a good comparison for six different micro-controllers, see [29, Table 14]. Here again, the average speed gain we get when replacing AES-GCM with Ascon is roughly 40%.

In the following section, we look at different cipher suites that are relevant for CANsec and discuss their behaviour regarding nonce repetitions.

3 REPEATING THE UNREPEATABLE: NONCE MISUSES

The three ciphers suites we are studying are AES Galois Counter Mode (AES-GCM), AES-GCM-SIV [19], and Ascon [15]. Each exhibits a different behaviour regarding nonce misuses.

[4] See https://csrc.nist.gov/projects/lightweight-cryptography

3.1 Nonce Misuse Vulnerable: Galois Counter Mode

The Galois Counter Mode (GCM), instantiated with AES,[5] is the industry standard in the automotive domain when it comes to AEADs. Most of today's automotive-grade microcontrollers, that come with an embedded Hardware Security Module (HSM), also offer AES-GCM hardware accelerators. This ubiquitous availability of the GCM makes it a natural choice for use cases that require authenticated encryption functionalities.

Figure 2 shows the exemplary structure of the GCM. In GCM, the plaintext is encrypted following the same approach as in the Counter Mode (CTR), i.e., the block cipher is turned into a stream cipher. This stream cipher is then used to generate a key stream, based on the encryption of a counter value, and the plaintext is encrypted by XORing with the key stream. Additionally, the authentication tag is computed based on multiplications with a key-dependent polynomial, denoted $\texttt{MUL}_H$ in Figure 2, XORing with the ciphertext and a final encryption with a key stream block. In other words, GCM is a combination of the CTR mode in an encrypt-then-mac scheme, using a Wegmann-Carter construction for the MAC (this MAC is also referred to as GMAC in the case of GCM). The key-dependent polynomial H is also called *hash key* and is computed as

$$H = E_K(0^n)\,, \tag{3}$$

where E is the block cipher (in our case: AES), K is the secret key, and n is the block length of E. Thus, the hash key is the encryption of the all-zero input.

The GCM is provable secure, see [23], in the *nonce-respecting* scenario, i.e., when no nonce misuses happen. NIST requires to change the secret key after 2^{32} encryptions [28].

When we leave the realm of nonce-respecting adversaries, the situation however changes drastically. In 2006, Joux described in the so-called "forbidden attack" [24], what happens if an adversary can force a nonce repetition.

GCM decryption. A first attack compromises the confidentiality: Assume the attacker recorded a ciphertext C with corresponding nonce N (and authentication tag T). If the attacker now learns a second ciphertext C' under the same nonce N (and authentication tag T') with the belonging plaintext P', this pair can be used to compute back the key stream $ks = P' \oplus C'$. This key stream depends only on the secret key K and the nonce N. If K did not change, the attacker can simply decrypt $C \oplus ks = P$, as the nonce N was repeated.

This attack is already known from CTR mode and applies to basically any stream cipher. However, in AEAD modes, the authentication tag and its verification typically prevent the forging of valid new plaintext / ciphertext pairs.

GCM forgery. With the attack by Joux, we can overcome this shortcoming. Similar to the first attack, the adversary again records two ciphertexts C, C' under the same nonce, N with authentication tags T, T'. Computing $T \oplus T'$ cancels out the "encryption" of the authentication tags and results in $S \oplus S'$, where S, S' are the values after the last polynomial multiplication $\texttt{MUL}_H$ (see the bottom part of Figure 2). This resulting polynomial $S \oplus S'$ has H as a root, which can thus be computed as such. Note that, if the attacker succeeds in recovering H, he has learned the hash key, which is *independent of the nonce*!

While this attack is "forbidden" in the sense that it is required to not repeat the nonce, later works have shown this to be a very realistic scenario, see e.g. [9].

Let us sum this section up with a short discussion of the impact of this attack in the scenario of CAN XL and CANsec. An adversary with access to the CAN XL bus can

(1) easily wiretap messages and
(2) suppress CAN XL frames by disturbing the transmission on the bus.

In case a nonce repetition in CANsec with the GCM cipher suite occurs, the attacker can recover the hash key H. For any following CAN XL frame protected with CANsec, the adversary can now record the original frame and suppress its successful transmission. He can then manipulate the frame using the well-known XOR malleability of stream ciphers (to which CTR and GCM belong), generate a new valid ICV / authentication tag using the learned hash key H and send this manipulated frame on the bus.

Note. The "encryption" of the authentication tag, or final key blinding step, does not prevent this attack for fresh nonces! Once the attacker learns the nonce-independent hash key, this key can be used to compute the internal authentication state S for a given (recorded) message and nonce. With S and the authentication tag, the attacker can then compute the "encryption" / key blinding term as $S \oplus T$ and use it for a new forgery under the same nonce. When the attacker suppresses the initial (recorded) message with the same nonce, the receiver will also not realize that the nonce is reused.

3.2 Nonce Misuse Resistance: AES-GCM-SIV

To avoid this dangerous nonce repetition, the NIST requirement is that the probability of an initialization vector (IV) collision should not exceed 2^{-32}, which is translated in [16] by limiting the allowed number of encryptions with AES-GCM using a random IV to 2^{32}. To overcome the GCM nonce misuse weakness, AES-GCM-SIV [17] is designed as a nonce misuse-resistant AEAD scheme and allows certain re-use of nonces. Based on the concrete security bounds of Gueron et al., we analyze the advantages of AES-GCM-SIV when applied to the particular use case of CANsec, see also [17], [19] and [18].

The number of different nonces in encryption and decryption queries is denoted by Q; N_E^i is the number of messages encrypted per nonce. The maximum message length is $2^m - 1$. The parameters are chosen, such that the dominating terms of the bounds are smaller than 2^{-32} (see Table 1). Scenario S1 in Table 1 gives the parameters of AES-GCM in a nonce-respecting setting, i.e., the number of messages encrypted per nonce is $N_E^i = 1$. An example taken from [17] is Scenario S2. It provides an overall improvement of 2^7 compared to S1 for the overall length ($Q \cdot N_E^i \cdot 2^m$).

We assume that the length of a CAN XL LLC frame is smaller than 2048 byte, corresponding to 2^7 (AES) blocks of 2^4 byte length (see Scenario S3 in Table 1). Then, the maximum number of nonce-reuse repetitions N_E^i for a given $Q = 2^{32}$ is $N_E^i = 2^{25}$ in S3. The case of the smallest CANsec frames is Scenario S4, where we assume

[5] Of course, GCM can in principle be instantiated with any block cipher.

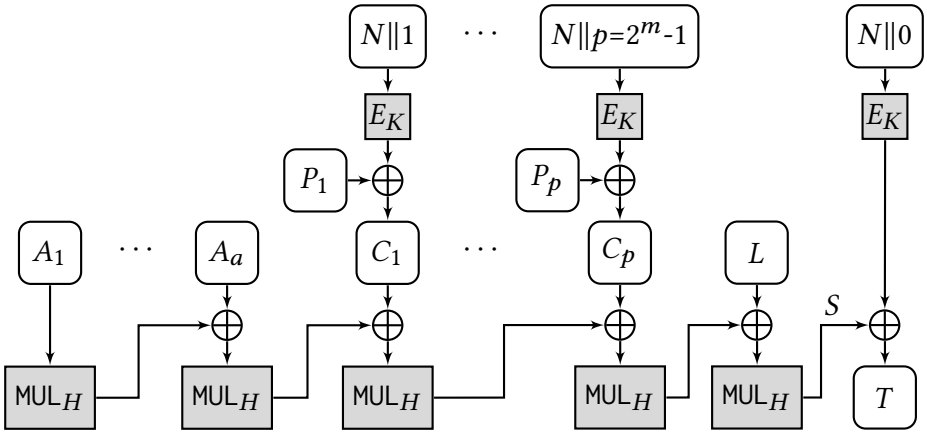

Figure 2: The structure of the AES-GCM encryption with secret key K, nonce N, the plaintext blocks $P_1, P_2, \ldots, P_p$ and the associated data blocks $A_1, A_2, \ldots, A_a$. Outputs are the ciphertext blocks $C_1, C_2, \ldots, C_p$ and the authentication tag T. S is the aggregated authentication value before the "encryption" / key blinding step to generate the actual authentication tag. The length L is the bit length of original authenticated data and plaintext and H is defined as in Eq. (3).

Table 1: Dominant terms of the security bounds of AES-GCM-SIV for four scenarios S1, ..., S4, where Q denotes the number of different nonces in encryption and decryption queries. N_E^i is the number of messages encrypted per nonce. S1: bounds from AES-GCM security proof a in nonce-respecting scenario. S2: exemplary bounds from AES-GCM-SIV paper [17]. S3: bounds for AES-GCM-SIV for CAN XL maximal length frame. S4: bounds for AES-GCM-SIV for CAN XL minimal length frame.

	Q	N_E^i	2^m	$Q \cdot B_{max}^2 / 2^{129}$	$\sum_{i=1}^{Q} (N_E^i)^2 / 2^{126-m}$
S1: AES-GCM [16]	2^{32}	1	2^{32}	–	–
S2: [17, Row 1, Table 1]	2^{45}	2^{10}	2^{16}	2^{-32}	2^{-45}
S3: CAN XL (a)	2^{32}	2^{25}	2^{7}	2^{-33}	2^{-37}
S4: CAN XL (b)	2^{32}	2^{30}	2^{2}	2^{-33}	2^{-32}

that at most 2^2 AES blocks are required to secure the whole CANsec frame. Then, a nonce can be repeated $N_E^i = 2^{30}$ times for a given $Q = 2^{32}$. Note, that in Scenario S4 the second term of Gueron et al.'s security bound is dominant.

Translating the values from Table 1 into expiration times T_n of a session key, see Eq. (1), results in the values given in Table 2. In Scenario S1 a 32-bit counter used as a freshness value for AES-GCM is exhausted in 2.16 days, while in S3 and S4 with AES-GCM-SIV the expiration time T_n of a session key is greater than 1000 years.

In summary, the *nonce misuse resistance* proven by Gueron et al. translates to an increased lifetime of a session key by a factor of 10^7 for applications in CANsec.

3.3 Nonce Misuse Resilience: Ascon

Besides providing full nonce misuse resistance as AES-GCM-SIV achieves as described above, we can also aim at a security level between the (catastrophic) nonce misuse weakness of AES-GCM and a full resistance (e.g. as of AES-GCM-SIV). One example of such an algorithm is Ascon [15], recently announced as the winner of the NIST lightweight cryptography competition (LWC).

NIST conducted a competition to find a new standard cryptographic algorithm for lightweight applications, i.e., use cases in very constrained environments. Such scenarios can require a very cheap implementation in either software or hardware, a low-latency design or low power consumption, among others. NIST started the LWC officially in 2018 with a call for algorithms and received 57 submissions of which 56 were accepted as Round 1 candidates. From these, 32 submissions made it to the second round and 10 were selected as finalists. NIST then announced to choose the Ascon submission on February, 7th, 2023.[6]

Besides NIST's LWC, Ascon was also selected as the first choice for the lightweight application use case of CAESAR (Competition for Authenticated Encryption: Security, Applicability, and Robustness), which was held between 2013 and 2019. CAESAR had a broader aim than NIST's LWC, namely to find new standard algorithms for AEAD "that offer advantages over AES-GCM and are suitable for widespread adoption".[7]

Ascon, winning two cryptographic competitions, received lots of scientific scrutiny and is thus a prime candidate for the choice of a modern cryptographic algorithm. In this section, we look at the nonce misuse behaviour of Ascon. Before getting into the details

[6] https://csrc.nist.gov/News/2023/lightweight-cryptography-nist-selects-ascon
[7] https://competitions.cr.yp.to/caesar-call.html

Table 2: Values of T_n as defined in Eq. (1) (scaled to years) for the four scenarios with CAN XL parameters as in Eq. (2). Scenarios S1, ..., S4 as in Table 1.

	n	T_n
S1: AES-GCM	32	$2.16 \cdot 1/365$
S2: [17, Row 1, Table 1]	55	$> 49.7 \cdot 10^3$
S3: CAN XL (a)	57	$> 199 \cdot 10^3$
S4: CAN XL (b)	62	$> 6.3 \cdot 10^6$

on this, we first recall the structure of Ascon and its inner workings up to the necessary details.

3.3.1 Sponge structure. Overall, Ascon is a *sponge-duplex* construction [6] and thus belongs to the field of *permutation-based* cryptography. In permutation-based cryptography, we build cryptographic algorithms (for encryption, hashing, etc.) using (un-keyed) permutations only. The sponge construction was mainly developed by the Keccak team, Bertoni et al., in the context of the SHA-3 competition and received a lot of attention also in its aftermath, as Keccak was chosen as SHA-3. The sponge construction enables nice security proofs, see e.g. [5] and is very versatile when it comes to its applications – that is, we can build hash-functions, MACs, AEADs, and more from it.

The term sponge was chosen, as the construction (especially for hash applications) can be split into two phases: an absorption phase and a squeezing phase. To stay in the hash-scenario: during the absorption phase, the message is "sponged" up into the internal state, while afterwards, the hash-digest is "squeezed" out of the internal state.

The AEAD scenario is shown in Fig. 3 for the particular case of Ascon. The overall encryption process is divided into four phases:

(1) Initialization: The constant, key and nonce are processed to build the first internal state.
(2) Associated Data: The associated data is consumed.
(3) Plaintext: The plaintext is consumed and the corresponding ciphertext is computed.
(4) Finalization: The authentication tag is computed.

Besides these four phases, the construction consists of the following important parts. The internal state is split into a *rate* and a *capacity* part, each of length r-bits and c-bits (for Ascon: $r = 64$, and $c = 256$). In each round of the construction, a permutation p is applied either a or b times often. The exact details of the permutation p are not relevant for the remainder of this work and are thus left out.

Let us now look at the nonce misuse behaviour of sponges in general and Ascon in particular. Note, however, that the designers of Ascon did not provide any security estimates in the nonce misuse scenario and we are thus outside of the claimed security properties of the cipher. The initial situation is thus the same as for GCM.

3.3.2 Nonce misuse behaviour of generic sponges. During the CAESAR competition, [31] identified two nonce misuse attacks, that apply generically to any sponge-duplex construction.

The first attack "CPA decryption: self-synchronizing streamciphers" works similarly to the nonce misuse CPA decryption attack on GCM: The attacker uses a known plaintext/ciphertext pair $(P_i, C_i)_i$ under a nonce N to compute the "keystream" (ks) of the encryption as $P_i \oplus C_i = ks_i$. Note that the keystream of a following block ks_{i+1} also depends on all previous plaintext blocks P_j, $j \leq i$. Now, for a repeated nonce N and new ciphertext $(C'_i)_i$, the attacker can partially decrypt C'_i. Let j denote the index, up to which the corresponding plaintext blocks P_i and P'_i are equal. In other words:

$$P_\ell = P'_\ell \qquad \forall \ell \in \{1, \ldots, j\} \ .$$

Then, the attacker can recover the j+1 blocks P'_ℓ, where $1 \leq \ell \leq j$+1 simply by adding the computed keystream from the first step:

$$P'_\ell = C'_\ell \oplus ks_\ell \qquad \forall \ell \in \{1, \ldots, j+1\} \ .$$

Due to the above-mentioned note that the keystream for one block depends on all previous plaintext blocks, this attack works only for the first $j + 1$ blocks. Once, a new plaintext block differs from the previous plaintext under the re-used nonce, the keystream for the new plaintext will also change and cannot be re-used.

This attack breaks the confidentiality under nonce misuses.

The second attack "Semi-universal Forgery on Sponges" finds for a given message P and authenticated data A two fresh nonces N and N', such that $E_K(N, A, P) = (C, T)$ is a valid ciphertext, as well as the forged one $N', A, (C, T)$. The attack finds these two nonces N and N' with a nonce reusing collision attack on the internal capacity of the sponge. Its complexity is thus bound by the birthday paradox and the capacities size c as $2^{c/2}$, which is 2^{128} for Ascon – and thus not better than brute-forcing the secret key.

Besides the unpractical complexity, we see no useful application for this generic attack in the particular case of CANsec.

3.3.3 Nonce misuse behaviour of Ascon in particular. When it comes to the nonce misuse behaviour of Ascon, we have to discuss two lines of work. The first line, see [3], [4], [11], [10], describes state-recovery attacks in a nonce misuse scenario. The attacks used for the state-recovery are so-called conditional cube attacks, which are a kind of symmetric cryptanalysis that exploits a low algebraic degree of the cipher. The state that is recovered by these attacks is the internal sponge state. The conditional cube attack in [3, 4] requires 2^{40} chosen encryptions, all under a fixed-key, fixed/reused nonce, fixed associated data input. Knowing this internal state allows us to decrypt arbitrary ciphertexts under the same input. However, an adversary that recovered the internal state *cannot*

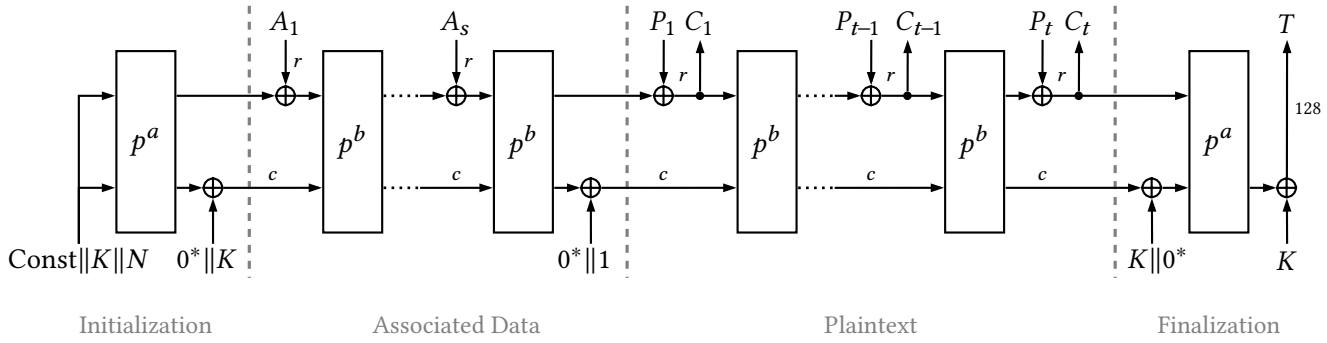

Figure 3: Ascon's sponge structure and mode-of-operation for encryption. The structure consists of four different stages: 1) During *initialization* the internal state is computed based on the secret key K and a nonce N. 2) Then the *associated data* blocks A_i is "absorbed into the sponge". 3) Next, the *plaintext* P_i is encrypted block-wise to the ciphertext C_i. 4) After that, the encryption is *finalized* and the authentication tag T is output. Each stage is domain-separated: initialization and finalization by a key-whitening step, associated data and plaintext by a fixed constant.

Table 3: Values of the dominating terms of the security bound of [26, Thm. 2] for Ascon in the authentication-only nonce misuse case for the four scenarios S1, ..., S4, where $k = 128$, $c = 256$, $Q_p = 2^{96}$ and $M = Q \cdot N_E^i \cdot 2^m \cdot 2$. Scenarios S1, ..., S4 as in Table 1.

	Q	N_E^i	2^m	$Q/2^k$	$Q_p/2^k$	$Q_p \cdot M/2^c$
S1: AES-GCM [16]	2^{32}	1	2^{32}	2^{-96}	2^{-32}	2^{-95}
S2: [17, Row 1, Table 1]	2^{45}	2^{10}	2^{16}	2^{-83}	2^{-32}	2^{-88}
S3: CAN XL (a)	2^{32}	2^{25}	2^7	2^{-96}	2^{-32}	2^{-95}
S4: CAN XL (b)	2^{32}	2^{30}	2^2	2^{-96}	2^{-32}	2^{-95}

- recover the secret key K, due to Ascon's key blinding step at the initialization, nor
- forge authentication tags, due to the key blinding step at the finalization.

Actually, and this is the second line of work, very recently [26] proved that Ascon achieves authenticity under state-recovery, exactly due to the above-mentioned key blinding steps.[8] We consider the high-level expression, see [26, Thm. 2], for the authenticity-only nonce misuse setting where M represents the total number of encryption and decryption blocks queried, Q_p is the number of primitive queries (denoted N by Mennink and Lefevre), and k denotes the key length in bits. For $m \geqslant 128$, $\mu = 1$ (single-user setting) and $n = 320$, we obtain that the advantage of an adversary is

$$O\left(\frac{Q + Q_p}{2^k} + \frac{Q_p \cdot M}{2^c}\right). \tag{4}$$

For comparison, we set the "offline complexity" to $Q_p \leqslant 2^{96}$ in Table 3. We show the corresponding values for $k = 128$, $c = 256$. Note, in the case of AES-GCM-SIV (see Table 1) we considered blocks of length 2^4 Bytes (AES input length), while in Ascon M blocks of length 2^3 Bytes are assumed. Therefore, we have $M = Q \cdot N_E^i \cdot 2^m \cdot 2$ (using the notation of Table 1). We can summarize that the capacity of $c = 256$ bits of Ascon allows high re-use of the nonce, while the offline complexity Q_p dominates the attacker's advantage. This is also the reason, why it makes no sense to compute an Ascon version of Table 1.

Authentication-only scenario. Note that, while Ascon specifies no dedicated authentication-only mode, we can simply use the AEAD mode, see [15, Algorithm 1]. Here, the plaintext $P \in \{0,1\}^*$ is allowed to be the empty string ε. Thus, choosing $P = \varepsilon$ while keeping the to-be-authenticated data A_i, enables the authentication-only scenario.

4 CONCLUSION

We have analyzed six nonce misuse attacks for AES-GCM, AES-GCM-SIV and Ascon. The attacks on AES-GCM allow to break the confidentiality for all messages sent under the same misused nonce as well as forge authentication tags for newly encrypted messages – with some restrictions *even under new nonces*.

For AES-GCM-SIV, the advantage and complexity of any attack under nonce misuses can be bound as in Table 1, resulting in basically 2^{25} to 2^{30} acceptable nonce repetitions for the CAN XL communication scenario.

For Ascon we discussed two generic attacks (valid for any sponge construction) and one specific attack on the Ascon scheme. The generic attack on the decryption breaks the confidentiality *partly*

[8] Confidentiality cannot be achieved under state-recovery, as an adversary can easily compute "backwards" and "forwards" from any internal state, as only the initialization and finalization steps are key-dependent.

for any message sent under the same misused nonce. Partly here means that confidentiality can be broken until and including the first diverging block, which differs from the known message. This includes the associated data – thus in case a difference occurs already in the associated data, the message cannot be decrypted. For the generic forgery attack, we do not see an application in the CANsec scenario. Finally, the conditional cube attack is also hardly applicable in the CANsec scenario: it requires a huge amount of misused nonces, where the associated data is fixed. Furthermore, once the attack recovers the internal state, this can be used for decrypting messages. However, forging tags requires to additionally break the authenticity. Following the result from [26], this requires additional computational effort of 2^{96} operations – which is infeasible (see Table 3).

4.1 Cipher Suites

While AES-GCM, as an industry standard for AEADs, probably cannot be ignored for a new protocol specification, we suggest including an alternative cipher suite. This alternative should provide better nonce misuse behaviour, to overcome AES-GCM's biggest weakness.

In-vehicle communication in the automotive domain always takes place in restricted internal networks, where confidentiality of messages is less relevant than their authenticity, see e.g., also [30, Annex 5, Mitigation M 10]. For this authentication-only scenario, we find Ascon to be much more resistant to nonce misuses than AES-GCM-SIV, due to its authentication-under-state-recovery property.

Taking also the lightweight aspects of Ascon into account, i.e. the costs for implementing it in hardware or software, but also the (smaller) effort required for a side-channel resistant implementation, we conclude that Ascon is a well-suited choice as a cipher suite for CANsec and recommend to include it in the specification of CANsec.

Besides the benefits mentioned so far (better security, more performance), our suggestion might introduce compatibility problems. In case an endpoint-ECU does only include an Ascon implementation (in software or hardware) and no AES-GCM implementation, this ECU can only communicate using CANsec-protected CAN XL frames to other participants on the bus, that also implement Ascon. The compatibility has to be ensured for all necessary communication streams by the system architect.

We accept this disadvantage, as we see our suggestion as an enabler for security in endpoint-ECUs. In other words, if the CANsec standard does not include a lightweight cryptography option, we expect the hurdles to introduce secure communication to such (very resource-constrained) endpoint-ECUs to be too high. Then (if they do not include any cryptographic support), these endpoint-ECUs cannot communicate via CANsec-protected CAN XL frames to other nodes on the bus, either.

4.2 Signalling Which Cipher Suite Is Used

The current CANsec proposal has no possibility of identifying the used cipher suite for a transmitted frame from the information contained in the CANsec frame fields. Note that this not only applies to the cipher suite but also to the used key length. In other words, by recording a CANsec-protected CAN XL frame, it is not possible to tell if AES-GCM-128 or AES-GCM-256 is used (or maybe another cipher-suite).

Thus, if our suggestion is adopted and Ascon is included in the CANsec specification, it also has to include a mechanism to process this information. Two generic solutions are possible: either treat the cipher suite choice as a system parameter, i.e., statically configure which CAN XL frames are protected with AES-GCM and which are protected with Ascon. Or, the cipher suite can be identified via additional information in the CANsec header, e.g., by introducing a new field in the reserved bits, see also Figure 1. Depending on how many cipher suites should be supported, one or more bits have to be used. As the current approach for the key length is following the first option, this might also be a good choice for another cipher suite. In both cases, the system architect has to ensure the compatibility of all nodes.

4.3 Required Security Level

Our initial claim that we aim to suggest a cipher suite with better security raises the question, of what the required security level for CANsec is. While AES offers a choice of three key lengths (of which two are considered for CANsec), namely 128-bit, 192-bit and 256-bit, Ascon does at most provide 128-bit security. We argue that a 128-bit level security is sufficient for securing in-vehicle communication, especially for endpoint-ECUs, whose communication is in many cases not secured at all currently. Furthermore, CAN XL is a real-time bus system, which implies that the biggest part of frames sent on the bus will be relevant for only a very short time frame. If data with higher long-term security requirements is sent via the CAN XL bus, it is recommended to add a second "long-term" security layer on top of CANsec.

Another argument against providing only a 128-bit cipher[9] might be quantum threats. Folklore evaluations of symmetric cryptanalysis solely judge the symmetric security level on the theoretical runtime of Grover's algorithm, resulting in a square root speed up and a (theoretical) security level of 64-bit for Ascon.[10] However, NIST does not follow this argumentation in their submission criteria for their Post-Quantum-Cryptography (PQC) competition:[11]

> [Level 1] Any attack that breaks the relevant security definition must require computational resources comparable to or greater than those required for key search on a block cipher with a 128-bit key (e.g. AES-128).

We thus rate Ascon to provide the "right" security from the perspective of classical and quantum security levels (as the AES does). The facet of security where Ascon fulfils our initial claim of providing *better security* than AES-GCM is the case of nonce misuse scenarios, as detailed in this paper.

ACKNOWLEDGMENTS

We thank Shay Gueron, Maria Eichlseder, Florian Mendel, Charlotte Lefevre and Bart Mennink for helpful discussions.

[9] Note that we suggest to include Ascon as an *additional*, optional cipher suite in the standard and *not to replace* AES-GCM.
[10] There is also an 80-bit quantum-security variant of Ascon.
[11] See https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/evaluation-criteria/security-(evaluation-criteria).

REFERENCES

[1] Mark D. Aagaard and Nusa Zidaric. 2021. ASIC Benchmarking of Round 2 Candidates in the NIST Lightweight Cryptography Standardization Process. Cryptology ePrint Archive, Paper 2021/049. https://eprint.iacr.org/2021/049 https://eprint.iacr.org/2021/049.

[2] AUTOSAR. 2020. Specification of Secure Onboard Communication Protocol. https://doi.org/fileadmin/standards/R20-11/FO/AUTOSAR_PRS_SecOcProtocol.pdf

[3] Jules Baudrin, Anne Canteaut, and Léo Perrin. 2022. Practical Cube-Attack against Nonce-Misused Ascon. NIST LWC Workshop. https://csrc.nist.gov/Presentations/2022/practical-cube-attack-against-nonce-misused-ascon

[4] Jules Baudrin, Anne Canteaut, and Léo Perrin. 2022. Practical Cube Attack against Nonce-Misused Ascon. *IACR Transactions on Symmetric Cryptology* 2022, 4 (Dec. 2022), 120––144. https://doi.org/10.46586/tosc.v2022.i4.120-144

[5] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. 2008. On the Indifferentiability of the Sponge Construction. In *EUROCRYPT (LNCS, Vol. 4965)*. Springer, 181–197.

[6] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. 2011. Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. In *Selected Areas in Cryptography (Lecture Notes in Computer Science, Vol. 7118)*. Springer, 320–337.

[7] BOSCH. 1991. CAN Specification 2.0 (1991, 1997). *Robert Bosch GmbH* (1991). https://web.archive.org/web/20221010170747/http://esd.cs.ucr.edu/webres/can20.pdf

[8] BOSCH. 2012. Bosch CAN FD Specification Version 1.0 (2012). *Robert Bosch GmbH* (2012). https://web.archive.org/web/20151211125301/http://www.bosch-semiconductors.de/media/ubk_semiconductors/pdf_1/canliteratur/can_fd_spec.pdf

[9] Hanno Böck, Aaron Zauner, Sean Devlin, Juraj Somorovsky, and Philipp Jovanovic. 2016. Nonce-Disrespecting Adversaries: Practical Forgery Attacks on GCM in TLS. Cryptology ePrint Archive, Paper 2016/475. https://eprint.iacr.org/2016/475 Published at Woot'16.

[10] Donghoon Chang, Deukjo Hong, Jinkeon Kang, and Meltem Sönmez Turan. 2023. Resistance of Ascon Family Against Conditional Cube Attacks in Nonce-Misuse Setting. *IEEE Access* 11 (2023), 4501–4516. https://doi.org/10.1109/ACCESS.2022.3223991

[11] Donghoon Chang, Jinkeon Kang, and Meltem Sönmez Turan. 2022. A New Conditional Cube Attack on Reduced-Round Ascon-128a in a Nonce-misuse Setting. NIST LWC Workshop. https://csrc.nist.gov/Presentations/2022/a-new-conditional-cube-attack-on-reduced-round-asc

[12] CiA. 2022. *CAN XL add-on services - Part 2: Security.* Technical Report CiA 613-2 Version 0.0.7. CAN in Automation.

[13] CiA. 2022. *CAN XL specifications and test plans.* Technical Report CiA 610-1 Version 1.0.0. CAN in Automation.

[14] Joan Daemen, Gilles Van Assche, Michael Peeters, and Vincent Rijmen. 2000. The Noekeon. In *First open NESSIE Workshop.*

[15] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. 2021. Ascon v1.2: Lightweight Authenticated Encryption and Hashing. *J. Cryptol.* 34, 3 (2021), 33.

[16] Morris Dworkin. 2007. *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC.* Technical Report NIST Special Publication (SP) 800-38D. https://doi.org/10.6028/NIST.SP.800-38D

[17] Shay Gueron, Adam Langley, and Yehuda Lindell. 2017. AES-GCM-SIV: Specification and Analysis. Cryptology ePrint Archive, Paper 2017/168. https://eprint.iacr.org/2017/168

[18] Shay Gueron, Adam Langley, and Yehuda Lindell. 2019. AES-GCM-SIV: Nonce Misuse-Resistant Authenticated Encryption. *RFC* 8452 (2019), 1–42.

[19] Shay Gueron and Yehuda Lindell. 2017. *Better Bounds for Block Cipher Modes of Operation via Nonce-Based Key Derivation.* Technical Report 702. https://eprint.iacr.org/2017/702

[20] Florian Hartwich et al. 2012. CAN with Flexible Data-Rate. 1–9.

[21] IEEE. 2018. IEEE Standard for Local and metropolitan area networks-Media Access Control (MAC) Security. *IEEE Std 802.1AE-2018* (2018), 1–239. https://doi.org/10.1109/IEEESTD.2018.8585421

[22] ISO. 1993. ISO 11898: Road Vehicles : Interchange of Digital Information : Controller Area Network (CAN) for High-speed Communication.

[23] Tetsu Iwata, Keisuke Ohashi, and Kazuhiko Minematsu. 2012. Breaking and Repairing GCM Security Proofs. In *CRYPTO (Lecture Notes in Computer Science, Vol. 7417)*. Springer, 31–49.

[24] Antoine Joux. 2006. Authentication failures in NIST version of GCM. NIST Comment. https://csrc.nist.gov/csrc/media/projects/block-cipher-techniques/documents/bcm/comments/800-38-series-drafts/gcm/joux_comments.pdf

[25] Gregor Leander, Christof Paar, Axel Poschmann, and Kai Schramm. 2007. New Lightweight DES Variants. In *FSE (Lecture Notes in Computer Science, Vol. 4593)*. Springer, 196–210.

[26] Bart Mennink and Charlotte Lefevre. 2023. Generic Security of the Ascon Mode: On the Power of Key Blinding. https://eprint.iacr.org/2023/796 Report Number: 796.

[27] Kamyar Mohajerani, Richard Haeussler, Rishub Nagpal, Farnoud Farahmand, Abubakr Abdulgadir, Jens-Peter Kaps, and Kris Gaj. 2020. FPGA Benchmarking of Round 2 Candidates in the NIST Lightweight Cryptography Standardization Process: Methodology, Metrics, Tools, and Results. Cryptology ePrint Archive, Paper 2020/1207. https://eprint.iacr.org/2020/1207 https://eprint.iacr.org/2020/1207.

[28] NIST. 2007. *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC.* Technical Report NIST Special Publication (SP) 800-38d. National Institute of Standards and Technology, Gaithersburg, MD. https://doi.org/10.6028/NIST.SP.800-38D

[29] Meltem Sonmez Turan, Kerry McKay, Donghoon Chang, Cagdas Calik, Lawrence E. Bassham, Jinkeon Kang, and John M. Kelsey. 2021. Status Report on the Second Round of the NIST Lightweight Cryptography Standardization Process. https://doi.org/10.6028/NIST.IR.8369

[30] United Nations. 2021. UN Regulation No 155 – Uniform provisions concerning the approval of vehicles with regards to cybersecurity and cybersecurity management system [2021/387]. , 30-59 pages.

[31] Serge Vaudenay and Damian Vizár. 2018. Can Caesar Beat Galois? - Robustness of CAESAR Candidates Against Nonce Reusing and High Data Complexity Attacks. In *ACNS (LNCS, Vol. 10892)*. Springer, 476–494.