A Periodic Authentication Scheme for Safety/Security Co-Design on CAN Buses

Mingqing Zhang, Dharshan Krishna Murthy, Philip Parsch, Henry Hoffmann, Philipp H. Kindt, and Alejandro Masrur

***Abstract*—With an increasing number of interconnected vehicles, it has become easier for an attacker to gain remote access to in-vehicle buses like CAN and, hence, security measures become necessary. However, most existing attempts to secure CAN either modify the hardware, which leads to increased costs, or apply encryption/authentication, which negatively impacts the timing behavior on the bus. In particular, when using authentication, new data (i.e., an authentication tag) is added, resulting in larger payloads. As a consequence, multiple CAN frames need to be sent (instead of only one) for each message, introducing extra delays and potentially leading to deadline misses. To address this problem, we propose a safety/security co-design approach based on combining a technique we call periodically authenticated encryption (PAE) with plausibility checks. The main idea is to not authenticate each and every message transmitted on CAN, but only with a certain configurable frequency. This way, less authentication tags are sent, which reduces the overhead. Plausibility checks are then used to determine whether non-authenticated messages sent in between two authenticated ones have been altered by an attack, e.g., whether the transmitted values exceed a given range or do not match previous ones. We then illustrate the benefits of the proposed approach on two realistic case studies consisting of emergency braking and adaptive cruise control.**

***Index Terms*—AES, authentication, automotive systems, CAN, cybersecurity, encryption, plausibility checks, safety, timing behavior.**

I. Introduction

VEHICLES are increasingly communicating with each other as well as with roadside and back-end infrastructure. As connectivity continues to grow in the automotive domain and, in general, in embedded systems, cybersecurity is becoming a greater concern [1], [2], [3]. In particular, an attacker can disturb in-vehicle buses, endangering the safety of passengers and possibly nearby road participants, especially, during critical maneuvers such as emergency braking, overtaking, etc.

Manuscript received 2 March 2023; revised 3 July 2023, 16 September 2023, and 17 October 2023; accepted 21 October 2023. Date of publication 17 November 2023; date of current version 22 April 2024. The review of this article was coordinated by Dr. Gedare Bloom. *(Corresponding author: Mingqing Zhang.)*

Mingqing Zhang, Dharshan Krishna Murthy, Philipp H. Kindt, and Alejandro Masrur are with the Department of Computer Science, TU Chemnitz, 09111 Chemnitz, Germany (e-mail: mingqing.zhang@informatik.tu-chemnitz.de; dharshan.krishna-murthy@s2013.tu-chemnitz.de; philipp.kindt@informatik.tu-chemnitz.de; a.masrur@cs.tu-chemnitz.de).

Philip Parsch and Henry Hoffmann are with the Department of Computer Science, The University of Chicago, Chicago, IL 60637 USA (e-mail: philip.parsch@googlemail.com; hankhoffmann@cs.uchicago.edu).

Digital Object Identifier 10.1109/TVT.2023.3333843

As a result, there is a need for increased (cyber-)security on in-vehicle buses. However, CAN, one of the most popular buses in the automotive domain, was not designed for this kind of scenarios. With limited payload and restricted transmission rates, CAN does not facilitate encryption and authentication, which are commonly used for securing communication. Particularly, encrypting and authenticating increase the size of messages, i.e, more frames need to be sent leading to longer delays, which may further jeopardize safety.

Existing approaches to securing CAN either provide an inadequate level of security, i.e., lack of encryption [4], [5], [6], [7], [8], [9] or authentication [10], [11], or require modifying the CAN protocol and/or its controllers [4], [10], [11], [12], [13], which significantly increases costs compromising competitiveness. This also holds true for more sophisticated networks such as CAN-FD, FlexRay and Automotive Ethernet.

Contributions: To overcome this issue, we propose an approach for safety/security co-design on CAN buses. Our approach consists in combining a technique called Periodically Authenticated Encryption (PAE) with plausibility checks. Basically, not every individual message needs to be authenticated, but authentication is carried out with a given configurable frequency instead, allowing us to reduce the associated overhead. Messages sent in between two authentications are validated by *plausibility checks* running on the different nodes.

In principle, CAN messages contain single data values representing physical quantities, e.g., the vehicle's velocity, etc. Such quantities cannot change arbitrarily fast over time, which is then used by a plausibility check to detect potential attacks. A plausibility check verifies whether the values contained in the messages are within an acceptable, i.e., safe, range with respect to previously transmitted (and authenticated) ones. Clearly, the higher the authentication frequency, the more overhead there will be on the bus. However, a low authentication frequency can lead to malfunction too, since plausibility checks will discard unauthenticated messages whose transmitted values are out of range, potentially compromising safety. As a result, the authentication frequency becomes an important design parameter, which we investigate in this paper.

We present two case studies consisting of emergency braking and adaptive cruise control (ACC). In particular, depending on the case study, we evaluate how the authentication frequency influences the resulting stopping distances and relative distance between vehicles and analyze the brake and ACC controller's stability when being under attack. Further, we show results

obtained from experiments performed on real hardware (viz., Arduino + CAN-BUS Shield) and from an OMNeT++ simulation. Based on the latter, we study the impact of different authentication frequencies on CAN's communication delay.

Our results indicate that the proposed technique considerably reduces communication delay on secure CAN buses (as compared to the case of authenticating each individual message). This allows us to effectively protect the bus against cyberattacks as detailed next, while still meeting typical automotive deadlines and, thereby, guaranteeing safety without increasing costs.

Structure of the paper: The paper is structured as follows: Section II discusses background knowledge necessary in this paper, whereas Section III presents related work. In Section IV, we discuss emergency braking and ACC as case studies, which we later use to explain the proposed approach and further for validation purposes. The proposed approach for safety/security co-design on CAN buses is then introduced in Section V. Further, Section VI presents our evaluation results, while Section VII concludes the paper.

II. Background

This section briefly discusses the necessary background knowledge for the remainder of the paper.

A. Controller Area Network

CAN is widely adopted in the automotive domain due to its robustness and, in particular, its low cost. It is a multi-master bus designed to simplify the wiring harness between ECUs (electrical control units) of a vehicle.

There are two different CAN versions, which basically differ in the number of bits used for the identifier (ID), which determines a frame's priority. An 11-bit ID is used in standard frames (i.e., 2048 different IDs are possible) and a 29-bit ID is used in extended frames [14]. It should be noted that standard and extended frames can coexist on the same bus and that the proposed technique in this paper is not affected by whether standard or extended frames are used. However, for ease of exposition, in this paper, we consider only standard (data) frames.

One of CAN's characteristics is that it implements a bitwise arbitration to resolve simultaneous transmissions ensuring real-time behavior based on a non-preemptive, fixed-priority scheme [14]. That is, during the transmission of the arbitration field, if the bus is idle and two or more nodes start transmitting simultaneously, the node sending a dominant bit at a given point in time (associated with a logical 0) wins arbitration over those sending recessive bits (associated with a logical 1). A dominant bit drives the state of the bus, i.e., the bus will be in a dominant state independently of what other nodes may be sending. If a node sends a recessive bit, but the bus is in a dominant state, the node immediately stops transmitting. That is, there is another node sending a higher-priority frame. In other words, the lower the frame's ID, the higher its priority will be. Nodes/frames losing arbitration will have to compete for the bus again after it becomes idle. However, under normal circumstances, the ongoing transmission of a message that has already won arbitration cannot be preempted despite its priority.

There are four types of frames in CAN: remote frame, data frame, overload frame, and error frame [14]. Remote frames are used to request data from another ECU/node, data frames are used to transmit data, overload frames are sent by a node to indicate that it cannot keep up with the rate at which frames are being sent, and error frames are transmitted when errors are detected.

Finally, CAN implements a number of error-signaling mechanisms that allow guaranteeing that either all nodes correctly receive the transmitted data or none of them does. This property eases the implementation of encryption methods, in particular, AES in counter mode, as explained later.

B. Encryption and Authentication

Encryption refers to the technique of converting plaintext (i.e., data) into ciphertext (i.e., unreadable/encrypted data), for which a *key*, i.e., a code, is typically used. Encryption techniques are classified into either symmetric or asymmetric encryption algorithms [15].

Symmetric encryption algorithms use the same key for both encryption and decryption. In principle, there are two kinds of such algorithms: stream cipher and block cipher. Stream cipher uses a keystream to encrypt or decrypt plaintext bit by bit. Steam cipher is mostly very fast, however, it requires a sufficiently long and random keystream, which is difficult to attain [15].

A block cipher takes an input block, i.e., a bit sequence with fixed length/size, and encrypts or decrypts it yielding an output block of the same size. Even though block ciphers are more complex and slower than stream ciphers, they are generally more secure [16]. In contrast to stream ciphers, if the plaintext is shorter than the block size, padding needs to be used, which increases the relative size of the ciphertext over the plaintext.

On the other hand, an asymmetric encryption algorithm uses a pair of keys: a public key known to all participants and a private key known only to the key owner. A message that has been encrypted using a public key can only be decrypted using the corresponding private key. Asymmetric encryption algorithms are mostly used in open networks, as they do not require sharing private keys, where no secure channel is available. The lengths of the keys are usually much longer in comparison to symmetric keys and, as a result, the computational complexity is increased [16].

Advanced Encryption Standard: AES or Rijndael is a block cipher encryption algorithm established by the National Institute of Standards and Technology (NIST) [17].

Since a block cipher such as AES defines only encryption and decryption processes for a single block, different modes of operation need to be used in order to establish how to repeatedly apply the same algorithm on multiple blocks. In this paper, we focus on the Counter (CTR) Mode and on the Galois/Counter Mode (GCM), as briefly explained next.

CTR is actually part of GCM, which uses an initialization vector (IV) combined with a counter to generate a counter block, which is then encrypted and applied to the plaintext (using an XOR) to generate the ciphertext. Clearly, the counter needs to be increased by some amount after each encryption to avoid

patterns being formed. However, the same counter value is required for both encryption and decryption and, thus, needs to be synchronized across all participants at all times. In our case, this can be easily done thanks to CAN's robustness (i.e., either every node/ECU gets a frame or none of them does), so nodes can increase this counter (by a fixed amount) with every successful transmission/reception.

The main advantage of CTR is that it does not increase the relative size (i.e., the number of bits) of the ciphertext with respect to the plaintext. However, CTR does not provide any data authentication, which leads us to GCM. GCM is an authenticated encryption (AE) cipher using CTR to generate the ciphertext and GHASH to obtain an authentication tag through an XOR operation with the encrypted and hashed IV. The authentication tag is then transmitted together with the ciphertext to ensure both data integrity and authenticity.

C. Attack Model

We assume that an attacker has no physical access to the CAN bus, but gains control over a non-critical node through wireless connection. This is the most probable attack scenario in the automotive domain, where physical access to the vehicle is normally not easy to achieve. This allows the attacker, however, to monitor the bus, replay CAN messages and impersonate other nodes to send malicious messages with their IDs. The attacker may perform one or more of the following actions as detailed next.

- Sniffing attacks (eavesdropping): The attacker can passively monitor the CAN bus to intercept messages being sent between different ECUs/nodes. This gives the attacker access to sensitive information, such as diagnostic codes, which they could use to compromise the network.
- Spoofing attacks (message injection): The attacker could actively inject messages onto the CAN bus, posing as a legitimate ECU, for example, to manipulate sensor readings. This could cause other ECUs to perform unexpected actions, such as disabling safety features or triggering actuators.
- Replay attacks: The attacker could capture messages from the CAN bus and replay them at a later time. This could cause the system to behave in unexpected ways or allow the attacker to gain access to sensitive information.

Note that desynchronization attacks like CANCloak [18] require physical access to the network and are, hence, out of the scope of this paper. Similarly, we disregard DoS-based attacks like Bus-off [19] and WeepingCAN [20], which can be dealt with by message filtering and throttling or by monitoring the bus traffic to identify and isolate malicious nodes [19], [21], [22].

We further assume that the attacker does not actually compromise any of the *critical* ECUs[1] and, thus, has no access to cryptographic keys, which are typically not accessible in a vehicle via wireless connection. As a result, only the above attacks are possible.

In the next section, we discuss existing approaches from the literature that aim to secure CAN as well as their advantages and disadvantages.

III. Related Work

There are a number of previously known approaches for securing CAN against different cyberattacks. Some of them use techniques to legitimize messages without encrypting their payload, which results in less overhead at nodes. Approaches based on sending authentication tags/codes without encryption are presented in [4] and [7]. More specifically, [4] proposes an *out-of-band*, i.e., separate, channel to send a 14-byte hashed authentication code, which poses the problem that CAN needs to be modified incurring high costs.

An even longer hashed authentication code that does not require modifying CAN was proposed in [7]. However, for each standard message, three additional frames are required to be transmitted. Although this approach provides a very reliable authentication, sending four frames per message leads to a significant delay, which makes it unsuitable for most automotive applications.

In [6], an approach is proposed that calculates a considerably smaller 1-byte authentication tag, which is appended to the original data and sent in the same frame. However, this is not as secure as with longer authentication tags and restricts the remaining payload data to at most 7 bytes.

Further, another non-encryption approach proposed in [8] uses a shuffling algorithm to change bit positions in the payload of CAN messages, which can only be correctly received by nodes knowing the shuffling function. This approach, however, cannot prevent replay/spoofing attacks. Another protocol named LeiA authenticates messages using a shared key initialized by the sender and receiver ECUs with a secure key agreement protocol [9]. The sender ECU computes a hashed message authentication code (HMAC) (using the shared secret), which the receiver ECU verifies upon reception. This approach also requires sending multiple frames, which impacts the timing behavior on CAN. A detailed analysis of schedulability/timing behavior considering different authentication schemes on CAN is presented in [23]. The main advantage of the proposed approach over the mentioned works is that it allows for trade-offs between authentication and timing requirements as discussed later in detail.

Further, an approach against spoofing attacks is proposed by Matsumoto et al. [11], where each node monitors whether messages with IDs assigned to it are being sent by other nodes in an unauthorized manner. If this is the case, the corresponding node sends error frames to override the unauthorized messages. In addition, secure protocols for CAN, called MaCAN and LiBrA-CAN, were proposed in [12] and [13], respectively. However, similar to [11], these protocols also require modifying CAN significantly. In contrast, our approach does not require modifying CAN and, hence, it can be implemented on existing systems without increasing costs.

[1]Critical ECUs are those holding the cryptographic keys. Once one of those ECUs are compromised, the attacker will get ahold of the corresponding encryption keys and there is nothing we can do to protect the system anymore.

In [5], a similar approach as in [8] is proposed, which uses asymmetric encryption to establish a session key between ECUs before attaching an authentication code to the original message. Without encrypting the payload, unauthorized nodes can eavesdrop on potentially sensitive data. To prevent this, a concept based on a stream cipher was proposed in [10], where the keys need to be generated dynamically and synchronized across all nodes. However, such a synchronization can be difficult to achieve without modifying the CAN controller and, in addition, a stream cipher is known to be less secure than AES, upon which our proposed solution is built. Asymmetric encryption is also used by a protocol named vatiCAN, where each node maintains digital certificates of trusted nodes [24]. Certificates are exchanged with neighboring nodes and verified against a set of trust anchors. Another protocol named vulCAN combines vatiCAN and LeiA; however, this requires changes to legacy ECUs to support hardware extensions [25], [26].

In [27], a Lightweight Encryption and Authentication Protocol (LEAP) based on the RC4 stream cipher [28] is proposed. Although LEAP introduces encryption and some basic authentication, it is considerably less secure than approaches based on block ciphers, as the one used in this paper. In particular, we aim for the best possible trade-off between (cyber-)security and safety on a CAN bus.

On the other hand, plausibility checks have been mainly used for detecting errors in different contexts, e.g., in sensor readings, etc. In [29] and [30], plausibility checks are used to monitor speed measurements. The use of plausibility checks for security purposes has also been considered. For example, in [31], multiple sensor inputs are combined without encryption to detect anomalies. Similarly, [32] proposes using plausibility checks or authenticity and integrity checks to reject cyberattacks based different models. Our work is in line with the latter. However, in addition to plausibility checks, we propose periodically sending authentication tags/codes and, thereby, achieve a higher level of security, as explained next.

IV. Considered Case Studies

In this section, we introduce two case studies consisting of emergency braking and ACC, which we use to explain our proposed approach in the next section.

A. Emergency Braking

Let us consider the example of a two-axle vehicle initially cruising at a typical highway speed of around 100 km/h, which undergoes emergency braking. To this end, the *brake ECU* is responsible for decelerating the vehicle at its maximum capability so as to minimize the stopping distance.

Fig. 1 shows the forces acting on the vehicle during braking. As explained in [33], forces in Newtons (N) generated by the vehicle's brakes are acting on the front and rear axles, and are denoted by F_{bf} and F_{br}, respectively. The aerodynamic force R_a aids in braking and is acting at a height h_a (in m) from the road surface. Similarly, the rolling resistances at the front and rear axles, denoted by respectively R_{rf} and R_{rr}, also aid in braking. The weights (in N) acting at the front and rear axles, denoted

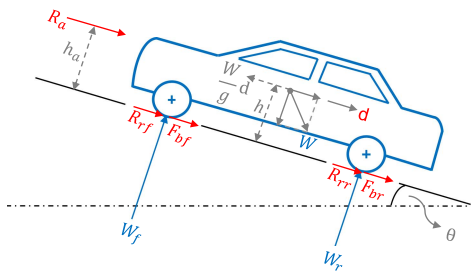

Fig. 1. Brake forces during braking [33].

by W_f and W_r respectively, constitute the total vehicle weight W. Finally, θ represents the angle (in degrees) of the road with respect to the horizontal.

All these forces produce a deceleration d (in m/s^2) at the vehicle's center of gravity, which is at a height h (in m) from the road surface. The deceleration can be computed as:

$$\frac{F_b + f_r W\cos(\theta) + R_a \pm W\sin(\theta)}{W} = \frac{d}{\mathrm{g}}, \tag{1}$$

where the brake forces at the front and rear axles (F_{bf} and F_{br} respectively) are combined into one resultant force F_b. The rolling resistances at the front and rear axles (R_{rf} and R_{rr} respectively) are also combined into one resultant force $f_r W\cos(\theta)$, where f_r is the coefficient of rolling resistance. The grade resistance $W\sin(\theta)$ has a $\pm$ sign indicating that it aids braking in an uphill ($+$ sign) and opposes it ($-$ sign) in a downhill. The acceleration due to gravity (in m/s^2) is denoted by g.

The force generated by the vehicle's brakes and the corresponding distribution to the front and rear axles determine the magnitude of the achieved deceleration. Only when the distribution is in proportion to the corresponding weights on the axles, the vehicle can decelerate at the maximum rate. However, vehicles have a fixed brake-force distribution, and hence, depending on the weights on the axles, the maximum deceleration that can be achieved varies even for the same vehicle.

Even under an optimal brake-force distribution, i.e., in exact proportions as per the weights on the axles, the achieved deceleration when normalized by g cannot exceed the coefficient of road adhesion μ. On dry asphalt surfaces, $\mu = 0.85$ and, hence, the maximum 'zachievable deceleration is 0.85 g. On snowy surfaces, this value reduces to around 0.2 g.

Attack scenario: Let us assume that an attacker injects spoofed/replayed CAN messages of the accelerometer such that they contain greater deceleration values than those achieved by the vehicle. In this case, the brake ECU, unaware of the cyberattack, relies on the modified accelerometer readings and applies a lower brake force than the required one. This, in turn, leads to the vehicle decelerating at a rate below its maximum capability. As a consequence, the stopping distance increases potentially leading to a collision with the traffic ahead and endangering passengers and other road participants.

B. Adaptive Cruise Control

Let us now consider our second example consisting of ACC on a highway, where the goal is either to keep a safe relative

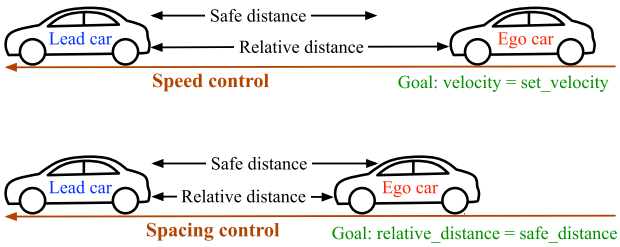

Fig. 2. Adaptive cruise control (ACC).

distance to the vehicle ahead (i.e., the lead car) or to maintain a driver-set speed when the lane is free. ACC uses radar sensors to detect the lead car and adjusts the speed accordingly [34].

As shown in Fig. 2, the relative distance as well as the safe distance between the two cars are calculated based on the position of the lead car and the current speed of the ego car. If the relative distance is larger than the safe distance, speed control is enabled to adjust the ego car's speed to the driver-set speed. Spacing control will be enabled once the relative distance between the two vehicles is smaller than the safe distance, in order to slow down the ego car and keep a safe distance to the lead car.

The distance traveled from a position p with an acceleration a in (in m/s^2) and current velocity v (in m/s) within the time t (in s) can be computed as $d = p + v \cdot t + \frac{1}{2} a \cdot t^2$.

Based on this, we can obtain the relative distance over time $d_{rel}(t)$ between the lead and ego car as follows:

$$d_{rel}(t) = \Delta p + \Delta v \cdot t + \frac{1}{2} \Delta a \cdot t^2, \tag{2}$$

where Δp, Δv and Δa are the differences in position (i.e., $p_{lead} - p_{ego}$), speed (i.e., $v_{lead} - v_{ego}$) and acceleration (i.e., $a_{lead} - a_{ego}$ between the two cars.

Attack scenario: Let us consider that the ego car, under spacing control, follows the lead car with a constant but lower speed than that set by the driver, i.e., the relative distance equals the safe distance. If now the lead car starts accelerating, ACC will switch to speed control and will also start accelerating the ego car to reach the driver-set speed. Instead of the lead car accelerating, let us assume that an attacker injects spoofed/replayed CAN messages of the relative distance such that they contain greater distance values than the actual ones. In this case, ACC speed control, unaware of the cyberattack, relies on the modified distance readings. It accelerates the ego car to match the driver-set speed, reducing the actual separation between the cars to a non-safe distance. As a consequence, a collision can happen endangering passengers and other road participants.

V. Safety/Security Co-Design

In order to deal with the cyberattacks according to Section II and the attacks in the scenarios described in Section IV, we propose to use a combination of encryption and authentication. By applying encryption, sniffing attacks can be effectively prevented. In addition, the use of counters as in AES-CTR greatly weakens spoofing and replay attacks, which then become random and cannot be controlled by the attacker. Moreover, introducing authentication completely prevents spoofing and replay attacks altogether, however, considerably increasing the communication overhead.

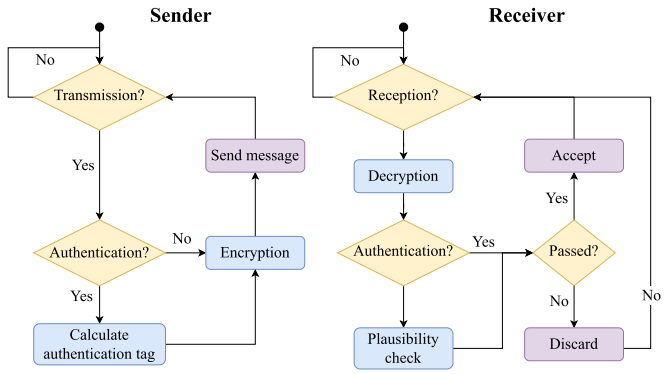

Fig. 3. Workflow of the sender and receiver node.

To overcome this predicament, we propose using *Periodically Authenticated Encryption (PAE)* combined with plausibility checks, which allows for safety/security co-design on CAN buses. More specifically, only a fraction of the messages on CAN are authenticated. Message authentication is carried out periodically with a given configurable frequency, which allows us to reduce the associated overhead. Unauthenticated messages sent in between two authenticated ones are still protected by encryption (i.e., AES-CTR) and verified by plausibility checks running on the different nodes.

PAE's workflow in combination with plausibility checks on both sender and receiver nodes is shown in Fig. 3.

A. Periodically Authenticated Encryption

We consider that CAN messages m_i from the accelerometer or from the radar sensor to the corresponding ECU, depending on the case study, are transmitted with a given repetition period p_i. Under PAE, not every instance of m_i is authenticated, but with a given frequency. To this end, we introduce the concept of *authentication frequency*, which we denote by $\frac{1}{\alpha_i}$, where α_i indicates a certain number of consecutive transmissions of m_i. In particular, a frequency of 1 over 1 (short $\frac{1}{1}$) implies that every instance of m_i is authenticated. Similarly, a 1 over 10 (short $\frac{1}{10}$) frequency indicates that only one out of ten m_i messages is authenticated, i.e., there are nine unauthenticated m_i messages between the two authenticated CAN messages.

Clearly, a higher authentication frequency will raise the level of security (since the origin of more messages will be reliably identified). However, this will also increase the overhead introduced and, therefore, will affect the timing on the bus. In the worst case, the ECU does not receive reliable updates on m_i for $\alpha_i - 1$ instances, which may compromise safety. For example, if sensor readings are available every 20 ms — which is a typical sampling period in the automotive domain, this results in a time interval of around 180 ms (with variations due to arbitration on the bus), during which messages are not authenticated. This means that the authentication frequency should also take safety requirements into account, which yields the aforementioned safety/security co-design.

Encryption: For PAE, we propose using AES in both CTR and GCM modes to encrypt and/or authenticate data. Since this is done at each individual node before sending a frame, it does not require changes in the CAN protocol/hardware and, hence, PAE can be used on standard networks without (significantly) increasing costs.

We consider that data is 8 bytes long, i.e., the full CAN payload is used.[2] AES-CTR is used to encrypt data without increasing its size, i.e., the ciphertext has exactly the same number of bits as the plaintext. Since AES-CTR encrypts data with a counter that is increased with each message sent, it prevents sniffing attacks and hinders replay and spoofing as well. That is, stored messages have outdated counter values, which can be easily detected.

As mentioned before, in addition to AES-CTR, AES-GCM generates 8-byte authentication tags that are sent periodically. This makes performing replay and spoofing attacks harder in comparison to using only AES-CTR. On the other hand, each authenticated message requires transmitting two data frames, resulting in an increased overhead with respect to AES-CTR.

Clearly, to implement AES in CTR and GCM mode, not only the key, but also a set of IVs (initialization vectors) and current counter values have to be stored on the local persistent memory of each node/ECU. Since a counter is incremented at the sender and at every receiver node each time the corresponding message is sent/received, different IVs and the current state of different counters need to be maintained for different message IDs. This implies that nodes can only *understand* the messages they are supposed to send/receive, increasing the level of security in the unlikely case that one of the nodes gets compromised.

B. Plausibility Checks

Plausibility checks monitor one or more variables of interest providing protection against alterations on unauthenticated messages. We identify basically two types of plausibility checks: absolute and differential plausibility checks. An absolute plausibility check is defined as follows:

$$T_{\min}(\cdot) \leq |\beta(t)| \leq T_{\max}(\cdot), \tag{3}$$

where $T_{\min}(\cdot)$ and $T_{\max}(\cdot)$ are thresholds or bounds on a time-varying parameter $\beta(t)$. $T_{\min}(\cdot)$ and $T_{\max}(\cdot)$ themselves may vary with the time or mode of operation. If $\beta(t)$ exceeds these bounds at any time t, the plausibility check is set to fail. On the other hand, a differential plausibility check rather monitors $\beta(t)$'s rate of change and is defined as follows:

$$|\beta(t_1) - \beta(t_2)| \leq T_{\text{dif}}(\cdot), \tag{4}$$

where $t_1 < t_2$ are points in time and $T_{\text{dif}}(\cdot)$ is a threshold/bound on $\beta(t)$'s rate of change within $t_2 - t_1$ time units. $T_{\text{dif}}(\cdot)$ may also vary with the time or operation mode.

[2]Even though this is not a necessary/compulsory condition for PAE, a full payload increases the level of security, i.e., it becomes harder to crack by brute force. In most cases, one may opt to add padding bits without considerably affecting timing. Padding bits may be all zeros, i.e., the actual value of the message is not changed and padding bits can be easily removed at the receiver to restore the original message.

Clearly, plausibility checks, in particular, the thresholds/bounds $T_{\min}(\cdot)$, $T_{\max}(\cdot)$ and $T_{\text{dif}}(\cdot)$, are very much application-dependent. As a result, we next discuss the plausibility checks in the context of the case studies introduced before.

Emergency braking: Let us consider the attack scenario described in Section IV-A, where altered deceleration values are injected by an attacker. However, the brake ECU can estimate of the maximum achievable deceleration based on the vehicle's brake-force distribution and loading conditions [33]. This can be used as an absolute plausibility check. That is, deceleration values that exceed the vehicle's maximum deceleration capability can be discarded by the brake ECU.

In addition, vehicle dynamics can be used for a further plausibility check. Once emergency braking is initiated, the commanded maximum deceleration cannot be instantaneously attained. The ECU initially exhibits a *transient* behavior during which the applied brake force increases in magnitude and the vehicle's deceleration gradually reaches its (desired) value. After this, the ECU performs minor or negligible modifications to the applied brake force such that the attained (maximum) deceleration is maintained until standstill.

The duration of the transient behavior depends on the performance of the underlying controller running on the brake ECU. For example, if the controller is designed using the proportional-integral-derivative (PID) technique, the chosen PID gains impact the duration of the transient behavior/phase.

For our case study, we assume that this phase lasts for 400 ms. In other words, once emergency braking is initiated (e.g., at time 0), the brake ECU continuously receives samples from the accelerometer through CAN messages every 20 ms.[3] Based on this, it increases the applied brake force reaching the maximum deceleration magnitude at 400 ms. From then onward, the ECU enters and stays in its second phase of operation, termed *steady state*, where the attained maximum deceleration magnitude is maintained until the vehicle reaches standstill.

In both these phases of operation, if the brake ECU is aware of the possible changes in deceleration magnitude from one sample to the next (20 ms in this case), it can use this information for an additional differential plausibility check. Note that these possible changes lie within two different ranges of deceleration, one for each of the two operation phases (transient and steady state). Hence, deceleration values injected during a cyberattack can be discarded, if the change in magnitude between two subsequent samples exceeds the corresponding range.

An example: Let us consider that our two-axle vehicle is cruising at an initial velocity of 30 m/s (i.e., 108 km/h) on a flat road (i.e., $\theta = 0$). Due to its loading conditions and fixed brake-force distribution, it can achieve a maximum deceleration magnitude of 7.28 m/s^2 (i.e., 0.74 g), which constitutes a first (absolute) plausibility check. That is, if an unauthenticated message is received with a deceleration value that is greater than the maximum achievable deceleration, it will be automatically discarded.

[3]A sampling frequency of 20 ms guarantees that all changes in acceleration can be captured without the vehicle having covered more than 1 m at a speed of around 100 km/h. Clearly, a higher speed requires a higher sampling rate.

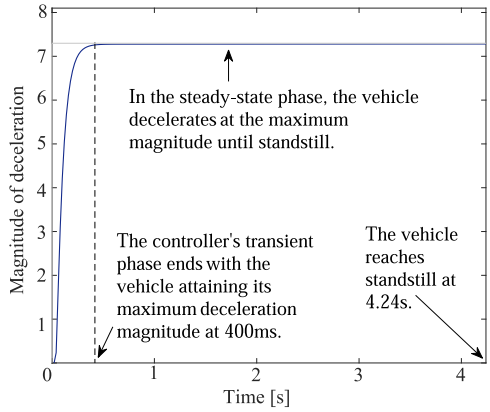

Fig. 4. Brake ECU deceleration tracking.

During emergency braking, as discussed before, the brake ECU has two operation phases, i.e., transient and steady state, as illustrated in Fig. 4. In the transient phase, the possible change in deceleration magnitude from one sample to the next lies in the range of 0.07 m/s^2 to 1.46 m/s^2. As a consequence, if an unauthenticated message is received with a deceleration value that is increased by a greater amount with respect to the previously received (and authenticated/validated) value, it will also be discarded, constituting a further differential plausibility check.

Similarly, in the steady state, this range is between 0.07 m/s^2 and 0.95 m/s^2, which makes up our third and last plausibility check for this case study — also differential. Again, if an unauthenticated CAN message is received at steady state with a deceleration value that exceeds the maximum possible increase with respect to the previous (valid) value, it will also be discarded.

Adaptive Cruise Control: Let us now analyze the attack scenario described in Section IV-B, where the relative distance between lead and ego car is altered by an attacker.

In the worst case, the attacker will spoof relative distance values such that they change as quickly as possible, since this will force the ego car to accelerate at its maximum rate to reach the driver-set speed in the shortest possible time. In reality, however, any increase in the relative distance, from a time t_1 to a time t_2, follows the expression:

$$\Delta d_{rel} = d_{rel}(t_2) - d_{rel}(t_1), \tag{5}$$

with $t_2 > t_1$ and $d_{rel}(t)$ given by (2). This constitutes a differential plausibility check, as detailed next. In contrast to emergency braking, the bound in (5) is a time-varying expression, which depends particularly on the cars' current speeds and accelerations.

An example: Let us consider a Porsche 911 Turbo as a lead car, with one of the highest maximum accelerations among all road vehicles of 10.5 m/s^2 [35]. Further, the ego car's acceleration is in the order of 2.5 m/s^2, as per the corresponding ACC standards [36]. As a result, the maximum acceleration difference (Δa) cannot exceed 8 m/s^2 — see again (2). If we now consider that the radar sensor measures the relative distance every 100 ms — as per ACC standards [36] — and that the two car had an equal initial speed — as ACC was operating in spacing control, an increase in the relative distance cannot be more than 0.04 m during the first 100 ms. That is, an unauthenticated sensor value received over CAN exceeding this increase in relative separation will be discarded. Note again that the maximum possible increase in relative distance changes over time according to (5), due to cars having different accelerations and speeds over time.

C. Authentication Frequency

As mentioned above, PAE's authentication frequency $\frac{1}{\alpha_i}$ is a configurable parameter. The more messages are authenticated, the higher the security level, however, also the more communication overhead is induced. This worsens CAN's timing behavior, potentially endangering safety.

Lower bound: We obtain a lower bound on α_i based on CAN's schedulability analysis [37]. Basically, a CAN message m_i is associated with a transmission period p_i, a (maximum) transmission time $c_{\max}$ and a deadline d_i, within which it needs to be sent successfully. Given an authentication frequency of $\frac{1}{\alpha_i}$, we know that the worst-case transmission time of m_i's k-th transmission is given by $r'_{i,k}$:

$$r'_{i,k} = \hat{b}_i + k \cdot c_{max} + \left\lceil \frac{k}{\alpha_i} \right\rceil c_{max} + \sum_{j=1}^{i-1} \left\lceil \frac{r'_{i,k}}{p_j} \right\rceil c_{max} + \sum_{j=1}^{i-1} \left\lceil \frac{r'_{i,k}}{\alpha_j \cdot p_j} \right\rceil c_{max}, \tag{6}$$

where $\hat{b}_i$ is m_i's blocking time by lower priority messages. Furhter, the term $\lceil \frac{k}{\alpha_i} \rceil c_{max}$ considers the additional overhead by authenticated m_i messages, whereas $\lceil \frac{r'_{i,k}}{\alpha_j \cdot p_j} \rceil c_{max}$ accounts for the authentication overhead by higher-priority m_j messages with $1 \leq j \leq i-1$.

In order to compute what value of α_i allows every transmission of m_i to meet its deadline, let us first find an upper bound on $r'_{i,k}$, for which we remove the ceiling functions in (6) to obtain:

$$r'_{i,k} \leq \hat{b}_i + k \cdot c_{max} + \left(\frac{k}{\alpha_i} + 1 \right) c_{max} + \sum_{j=1}^{i-1} \left(\frac{r'_{i,k}}{p_j} + 1 \right) c_{max} + \sum_{j=1}^{i-1} \left(\frac{r'_{i,k}}{\alpha_j \cdot p_j} + 1 \right) c_{max},$$

which can be solved for $r'_{i,k}$ to obtain:

$$r'_{i,k} \leq \frac{\frac{k}{\alpha_i} c_{max} + C_{i,k}}{1 - \tilde{U}_{i-1}}, \tag{7}$$

with $C_{i,k} = \hat{b}_i + (k-1)c_{max} + 2 \cdot i \cdot c_{max}$ and $\tilde{U}_{i-1}$ given by $\sum_{j=1}^{i-1} \frac{c_{max}}{p_j} + \sum_{j=1}^{i-1} \frac{c_{max}}{\alpha_j p_j}$. Further, if the right-hand side of (7) is less than or equal to $D_{i,k} = d_i + (k-1) \cdot p_i$, m_i's k-th

transmission can always meet its deadline:

$$\frac{\frac{k}{\alpha_i}c_{max} + C_{i,k}}{1 - \tilde{U}_{i-1}} \leq D_{i,k},$$

which we can now solve to obtain a lower bound on α_i:

$$\frac{k \cdot c_{max}}{D_{i,k} \cdot (1 - \tilde{U}_{i-1}) - C_{i,k}} \leq \alpha_i. \tag{8}$$

Note that (8) assumes that we know all α_j for $1 \leq j \leq i$, i.e., the authentication frequencies of m_j messages with higher priority than m_i, assuming that messages are sorted in that order. That is, we need to start computing the α_i parameter in the order of decreasing priority for (8) to be valid. In addition, it is necessary to compute (8) for all k transmissions of a message m_i with $1 \leq k \leq 1 + \lfloor \frac{t'_{busy} - d_i}{p_i} \rfloor$. Here, t'_{busy} is the longest possible busy interval on CAN when PAE is used, i.e., the longest time interval without idling given by the fixed point of the following equation [37]:

$$t'_{busy} = \sum_{i=1}^{n} \left\lceil \frac{t'_{busy}}{p_i} \right\rceil c_{max} + \sum_{i=1}^{n} \left\lceil \frac{t'_{busy}}{\alpha_i \cdot p_i} \right\rceil c_{max}.$$

Upper bound: Similarly, we can obtain an upper bound on α_i by using the concept of plausibility check. Basically, if too few messages are authenticated, an attacker may alter values causing malfunction or even damage. For a given time-varying parameter $\beta(t)$, let us assume that a maximum deviation of $\pm\beta_{max}$ can be tolerated. Further, let us consider that a differential plausibility check with a constant threshold given by $|T_{dif}|$ can be identified for $\beta(t)$. An unauthenticated message arriving via CAN will be discarded, if it exceeds the previously authenticated/validated value by more than $|T_{dif}|$. As a result, it will take an attacker a given number of transmissions to alter $\beta(t)$ for more than $\pm\beta_{max}$, which we use to derive an upper bound on α_i:

$$\left\lfloor \frac{|\beta_{max}|}{|T_{dif}|} \right\rfloor \geq \alpha_i. \tag{9}$$

That is, if more than α_i transmissions of m_i are left unauthenticated, an attacker can potentially cause malfunction/damage.

VI. Evaluation

For the sake of evaluating the proposed PAE under different conditions, we make use of OMNeT++ and MATLAB/Simulink. First, we analyze how an increasing authentication frequency affects the communication delay. As expected and shown later in detail, the communication delay increases with a higher authentication frequency, since this generates more overhead (i.e., more frames) on the bus.

Second, we evaluate the impact of breaching cyberattacks on the stability of emergency braking and ACC in our case study. We here consider the worst case of a plausibility check failing to detect altered/malicious messages, i.e., altered values are pessimistically assumed to always be in the range of expected values. As discussed later, this impact increases with a decreasing authentication frequency, since plausibility checks are exposed for a longer period of time.

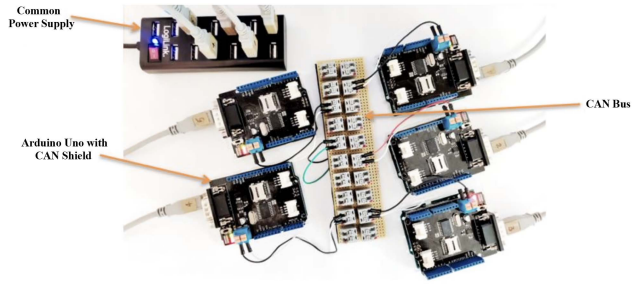

Fig. 5. Implementation on Arduino UNO boards.

TABLE I
Timings in Our Implementation

Symbol	Description	Value [μs]
t_{ECU}	Overhead on the node accounts for CAN stack, etc.	125
t_{CAN}	Average transmission time of packets sent	250
t_{GCM}	Time for encryption/decryption by AES-GCM	878
t_{CTR}	Time for encryption/decryption by AES-CTR	533

A. Communication Delay vs. Authentication Frequency

We compare the average and longest communication delay of PAE with respect to the case of no encryption (NE) based on an OMNeT++ simulation of CAN [38], [39] and taking different priorities into account. CAN's transmission speed used in our simulation is 500 kbps, as for most automotive applications, and the key size for AES is 128 bits.

In the following, we first implement AES-CTR and AES-GCM on real hardware to realistically assess processing (t_{ECU}) and encryption/decryption times (t_{CTR} and t_{GCM}, correspondingly). Further, we consider synthetic and realistic data. We assume a standard ID assignment, i.e., the two frames of an authenticated message are assigned the same ID as the original message. However, other ways of assigning IDs/priorities to reduce contention on the bus can be used [40].

Implementation: We implemented the proposed approach on Arduino UNO boards equipped with CAN-BUS Shields, as shown in Fig. 5. In particular, we let the nodes transmit around 10,000 CAN messages and collected the relevant timing information. The resulting average values are presented in Table I.

When using AES-CTR, our approach has an average end-to-end delay of 1566 μs, which results from:

$$D_{CTR} = 2 \cdot (t_{CTR} + t_{ECU}) + t_{CAN}, \tag{10}$$

where $2 \cdot (t_{CTR} + t_{ECU})$ is the computing overhead on both the transmitting and receiving node.

When using AES-GCM, our approach has an average end-to-end delay of 2506 μs, which is this time given by:

$$D_{GCM} = 2 \cdot (t_{GCM} + t_{ECU} + t_{CAN}), \tag{11}$$

where $2 \cdot (t_{GCM} + t_{ECU})$ is the computing overhead on the transmitting and receiving node. Here, two frames are sent over CAN and, hence, t_{CAN} needs to be accounted for each of the frames.

Note that t_{ECU}, t_{CTR} and t_{GCM} are rather constant times that depend on the platform used, whereas t_{CAN} depends on CAN's bandwidth and on contention.

TABLE II
SYNTHETIC SET OF CAN MESSAGES (ALL MESSAGES HAVE AN 8-BYTE PAYLOAD)

CAN ID	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Period $[ms]$	25	30	35	40	40	50	55	60	65	65	145	150	165	185	185	190	190	220	225	230

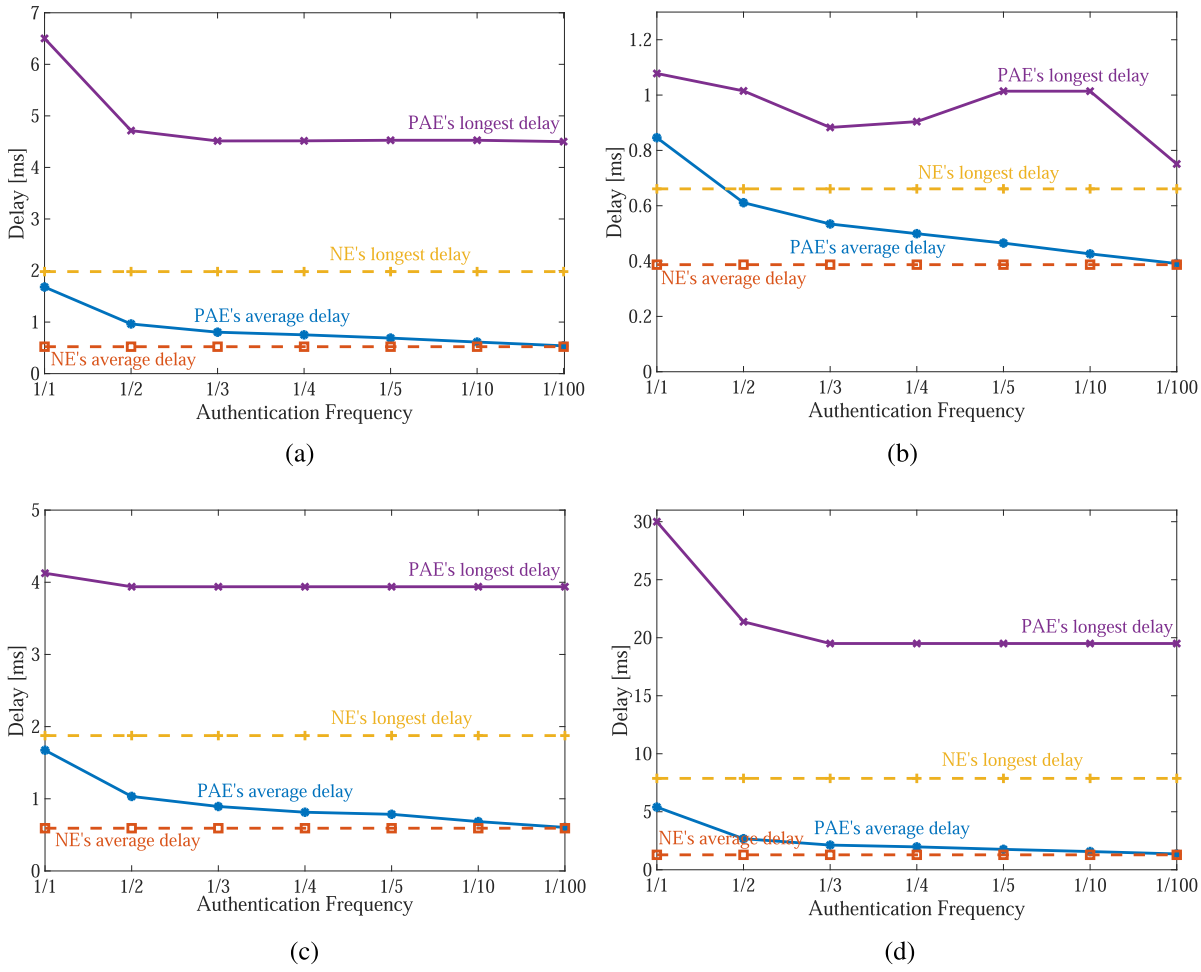

Fig. 6. Delay under different authentication frequencies for the synthetic message set of Table II. (a) Average over all priorities (b) The highest-priority message (i.e., ID 0) (c) An intermediate-priority message (i.e., ID 9) (d) The lowest-priority message (i.e., ID 19).

Synthetic data: We simulated a CAN bus consisting of 20 nodes using an OMNeT++ simulation from the literature [38], [39]), with each node sending one message. Message periods were randomly selected between 25 ms and 250 ms. Further, the payload of all the messages was chosen to be 8 bytes, resulting in the longest possible transmission time of 250 μs, as shown in Table II. The message priorities/IDs on CAN have been assigned according to Rate Monotonic (RM), i.e., the higher the rate of a message, the higher its corresponding priority.

We configured this simulation to consider the previously obtained t_{ECU} as well as t_{CTR} and t_{GCM} where applicable. As mentioned above, t_{CAN} depends on the level of contention on the bus, yielding different communication delays as a function of PAE's authentication frequency.

Fig. 6(a) shows both the average and measured longest delay of the proposed PAE and the case of no encryption (NE), see again Section V.[4] The shown curves in Fig. 6(a) are independent of the message priority/ID, i.e., they show the corresponding average delays of all messages. Further, the timing behavior of ID 0 (i.e., the highest priority), ID 9 (i.e., a middle priority) and ID 19 (i.e., the lowest priority) are shown in Fig. 6(b) to (d). As expected, both the average and longest delays decrease rapidly as the authentication frequency decreases from $\frac{1}{1}$ to $\frac{1}{100}$, i.e., from the case where every message is authenticated to the case where only one over 100 messages is authenticated. It is worth noting that this improvement slowly stagnates from frequencies of $\frac{1}{3}$ onward. Hence, authenticating every third message on the bus might be

[4]Note that NE's behavior is very similar to the behavior of only encrypting (but not authenticating) messages, however, without the overhead induced by the encryption algorithm.

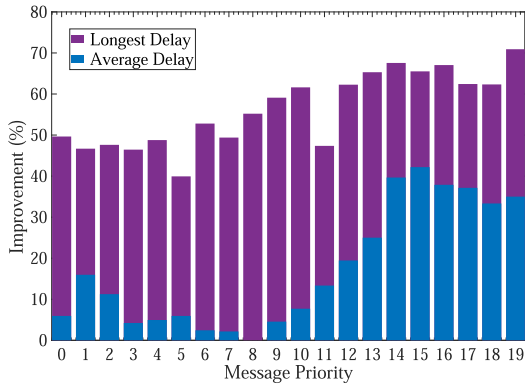

Fig. 7. Improvement of average and longest delay of $\frac{1}{10}$ from $\frac{1}{1}$ authentication frequency.

TABLE III
SET OF CAN MESSAGES IN BMW E90 [41]

CAN ID	Length (Bytes)	Period (ms)	Description
0x0A8	8	10	Status of torque, clutch and brake
0x0AA	8	10	Engine RPM and throttle
0x0C0	2	200	ABS / Brake counter
0x0CE	8	10	Individual wheel speeds
0x0D7	2	200	Counter (related to airbag / seat belt)
0x130	5	100	Ignition and key status
0x19E	8	200	ABS / Braking force
0x1A6	8	100	Speed
0x1D0	8	200	Engine temperature, Pressure sensor and handbrake
0x21A	3	5000	Status of lighting
0x26E	8	200	Status of ignition
0x335	8	1000	Unknown
0x349	5	200	Fuel level sensors
0x34F	2	1000	Status of handbrake
0x380	7	-	VIN (vehicle identification number)
0x39E	8	-	Set date and time
0x3B4	8	4000	Status of battery charge and voltage
0x581	8	5000	Status of seat belt

sufficient to achieve a decent trade-off between timing/safety and security.

Fig. 7 shows the relative improvement of both average and measured longest delay when one over 10 messages are authenticated with respect to the case where every message is authenticated. Independent of the message priority/ID, an improvement of around 30% to 40% can be observed for the average delay. The longest measured delay is not always improved. However, in general, we can conclude that rather lower-priority messages (i.e., IDs greater than 10 in this setting) benefit from a lower authentication frequency when it comes to the longest measured delay. This is to be expected because, with a lower authentication frequency less higher-priority messages are authenticated and, hence, they can be sent within one frame. Hence, lower-priority messages are prevented for a lesser time.

Real-world data: We also simulated BMW E90's CAN messages as shown in Table III [41]. Again, we used t_{ECU}, t_{CTR} and t_{GCM} as obtained above and analyzed the effect of authentication frequency on t_{CAN} and, hence, on the overall delay with respect to the NE case.

Fig. 8(a) shows the combined average and longest measured delays for message IDs 0x0A8, 0x1D0, and 0x581. The individual timing behaviors of these messages are shown in Fig. 8(b) to (d).

As expected, the average delay decreases rapidly as the authentication frequency decreases from $\frac{1}{1}$ to $\frac{1}{100}$, i.e., from the case where every message is authenticated to the case where only one over 100 messages is authenticated. Similar to the case of synthetic data, it can be noticed that the improvement slowly stagnates from frequencies of $\frac{1}{3}$ onward. Fig. 8(b) and (c) also indicate that there is almost no improvement on the average delay when authentication frequency drops below $\frac{1}{100}$, i.e., PAE's delay gets very close to NE's delay, for the highest- and the middle-priority message.

In contrast to average delay, there is no significant improvement of the longest measured delay due to the low bus utilization by these messages. In general, the higher the bus utilization, the greater the improvement by the proposed PAE is going to be.

B. Comparison With Related Work

In this section, we compare our proposed approach to the literature. Approaches like [4], [10], [11], [12], and [13] need to modify the CAN protocol and/or the CAN controllers. As a result, they are not comparable to the proposed approach, since they significantly increase costs and compromise CAN's competitiveness over other in-vehicle buses.

Similar approaches to secure CAN without modifying the protocol itself, like vatiCAN [24], incur an end-to-end delay of 3.3 ms and have a complicated key/certificate management. In contrast, our proposed approach based on PAE has an end-to-end delay of 1.5 ms (encrypted) and 2.5 ms (encrypted and authenticated), which leads to an average overhead of 1.7 ms with an authentication frequency of $\frac{1}{5}$ and 1.6 ms with an authentication frequency of $\frac{1}{10}$. This delay can be drastically reduced, if encryption is done by hardware instead.

Approaches like [4], [5], [6], [7], [8], and [9] do not offer encryption and are, as opposed to our approach, vulnerable to sniffing attacks. Furthermore, approaches like [10] and [11] do not offer authentication and are hence susceptible to spoofing/replaying. Note that we have purposely forgone any numeric comparisons with previously known approaches, but rather compare to the case of no encryption/authentication that serves as a common baseline. Comparisons to this baseline are more objective and generic than directly comparing individual approaches.

C. Impact of Breaching Cyberattacks

In this section, we investigate the impact of breaching cyberattacks in the context of our case study from above. To this end, we simulate the full-fledged vehicle model on MATLAB/Simulink as discussed in Section IV.

We assume that both the accelerometer and radar distance sensor in our case studies are available and normally send (unaltered) deceleration values with a certain frequency (i.e., the sampling frequency). However, both the brake and ACC ECU are assumed to also receive malicious messages from an attacker. Whereas authenticated messages can be easily validated under

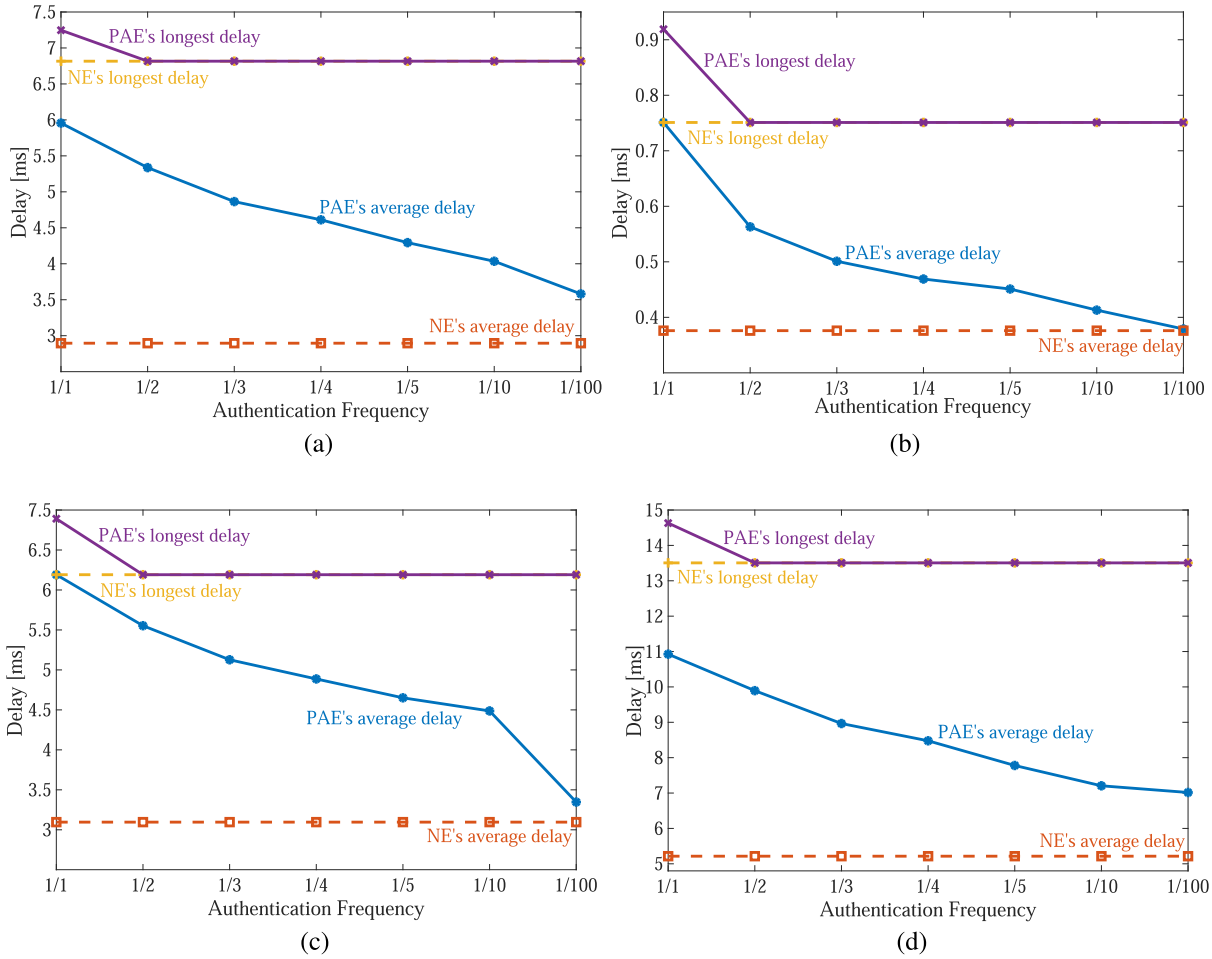

Fig. 8. Delay under different authentication frequencies for BMW E90's message set (see Table III). (a) Average over all three priorities (b) The highest-priority message (i.e., ID 0x0A8) (c) An intermediate-priority message (i.e., ID 0x1D0) (d) The lowest-priority message (i.e., ID 0x581).

PAE, the systems are vulnerable to replay/spoofing attacks when only plausibility checks are in place, i.e., in between two authenticated messages.

As discussed before, plausibility checks establish a range of plausible/valid values, based on which deviating messages can be discarded. The messages failing plausibility and/or authentication checks will be discarded without affecting the application, since the previous state is maintained, as already explained. Following messages sent with the same CAN ID will not be affected by this and will be checked by plausibility or authentication checks depending on the configuration (authentication frequency). In the following, we assume the unlikely worst case that the attacker always manages to alter the unauthenticated messages such that plausibility checks are unable to detect them. That is, the attacker injects the greatest possible, but still plausible deceleration values (or relative distance values) with every update/sample. Depending on the case study, the brake ECU or ACC ECU receives both the actual value from the sensor and the altered value from the attacker, but cannot tell which one is right. We consider at this point that the ECUs always selects the attacker's message and ignores the one from the accelerometer or radar sensor, which again constitutes the worst case. As a result, in our emergency braking case study, the brake ECU can be deceived into reducing the brake force, since it interprets that the vehicle is slowing down at a higher rate than desired/configured. This attack, hence, ends up prolonging the stopping distance and, potentially, causing damage. As for our ACC case study, the ACC ECU can be deceived into accelerating until reaching the driver-set speed, since it interprets that the lead car is speeding up at its maximum acceleration. The attack ends up reducing the actual relative distance between the lead and ego car to an unsafe value, which may potentially lead to a collision.

Emergency braking: Let us first consider the evaluation results of our emergency braking case study. Fig. 9 demonstrates the impact of such a cyberattack on the stopping distance as a function of PAE's authentication frequency. We consider two different vehicles, again, on a flat road: One with a maximum deceleration magnitude of 7.28m/s^2, referred to as the *best* (braking) vehicle. The other one, referred to as the *worst* (braking) vehicle, has a maximum deceleration magnitude of only 4.76m/s^2.

During the best vehicle's transient phase, a plausible change in deceleration magnitude from one sample to the next (which are separated by 20 ms, as discussed before) is in the range of 1.46m/s^2 to 0.07m/s^2. Similarly, in the steady state, this

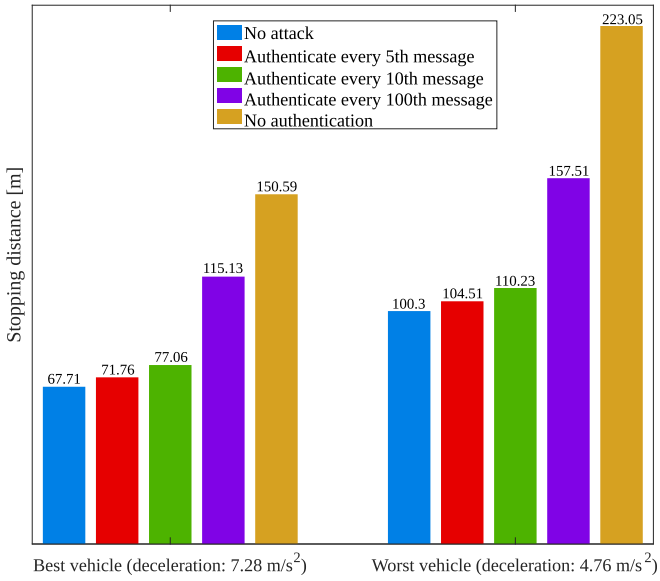

Fig. 9. Impact of cyberattacks on the stopping distances of two vehicles types.

range is from 0.95m/s^2 to 0.07m/s^2. Again, we assume that the attacker always manages to alter messages by an amount equal to 1.46m/s^2 at transient and to 0.95m/s^2 in the steady state, i.e., to the greatest possible change in deceleration magnitude,[5] with the additional constraint that injected values do not exceed the vehicle's maximum deceleration magnitude. Otherwise, the attackers messages would be rejected due to failing the plausibility check.

Let us now consider the case of an authentication frequency of $\frac{1}{5}$, i.e., messages sent with a period of 20 ms are authenticated once every 100 ms. Therefore, from the start of the maneuver to 100 ms later, the attacker is assumed to successfully inject the values -1.46m/s^2, -2.92m/s^2, -4.32m/s^2, and -5.83m/s^2 at the times 20 ms, 40 ms, 60 ms, and 80 ms respectively, which are plausible and processed by the ECU. At time 100 ms, an authenticated message arrives from the accelerometer containing the vehicle's actual deceleration, such that the brake ECU recovers to some extent. The attacker then keeps injecting altered deceleration values this way, however, periodically arriving authenticated messages allow bringing the system back to normalcy. In this case, note that the best vehicle's stopping distance is still 71.76 m — see the 2nd bar on the left-hand side of Fig. 9 — being only 4 m longer than the stopping distance when no attack occurs (cf. the 1st bar on the left).

Further, increasing the time duration between two authenticated messages, i.e., decreasing PAE's authentication frequency, prolongs the stopping distance when under attack. This is the case for both vehicle types as, again, observed in Fig. 9. Clearly, when no authentication is performed, injected false values by the attacker cannot be compensated at all. In this case, the stopping distances are the longest for both the best and worst vehicle with around 150 m and 223 m respectively — see the 5th bar on the left-hand and the right-hand side of Fig. 9. The same vehicles under no cyberattack can reach standstill in around 67 m and 100 m respectively, i.e., the stopping distance can be more than doubled, if PAE's authentication frequency is set too low. On the other hand, as discussed before, the higher the authentication frequency, the more the overhead caused, which may also lead to malfunction due to missing deadlines on the CAN bus. A good trade-off in our case study seems to be given by authentication frequencies between $\frac{1}{5}$ and $\frac{1}{10}$.

[5]Note that, if the attacker would alter the deceleration magnitude to a lower value, the brake ECU would then apply a greater brake force than what it is actually necessary, thereby eventually saturating at the maximum brake force and, as a result, the stopping distance is rather reduced with less or no negative impact on the vehicle's braking behavior.

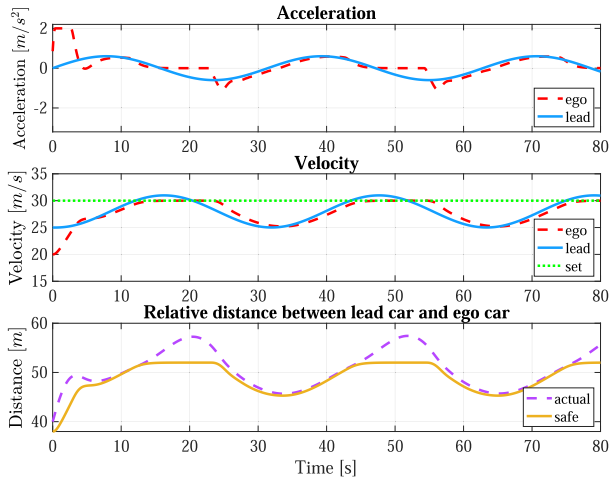

Fig. 10. ACC under no attack.

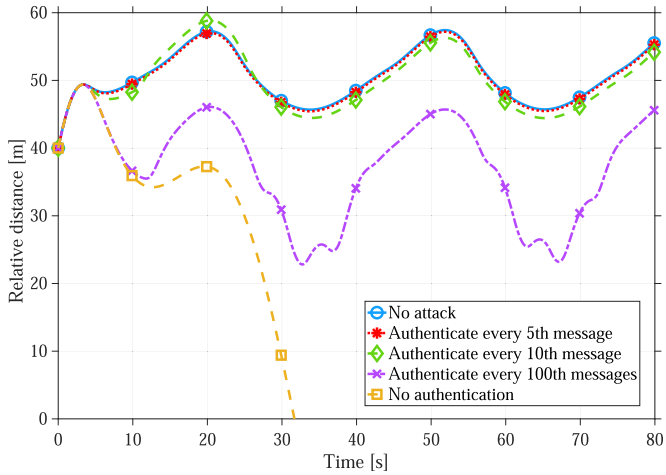

Fig. 11. Relative distance between lead & ego car.

Adaptive Cruise Control: Let us now analyze the evaluation results of our ACC case study with a simulation time of 80 s. The acceleration and speed/velocity of the lead and the ego car as well as their relative distance between are shown in Fig. 10 for the case of no attack.

Further, we assume the attack scenario described in Section IV-B. Fig. 11 illustrates the impact of such an attack on the relative distance between lead and ego car as a function of PAE's authentication frequency. For this simulation, we assume

that the ego car follows the lead car on a flat road, that the lead car has a maximum acceleration magnitude of 10.5m/s^2 and that the ego car's acceleration ranges from -3.5m/s^2 to 2.5m/s^2 according to [36].

During the attack, a plausible increase in relative distance is calculated by (5), with a maximum possible acceleration difference between the two cars of 8 m/s^2, as discussed in Section V-B. Again, we assume that the attacker always manages to alter messages the most such that altered messages cannot be detected by any plausibility check in place.[6]

Let us now again consider an authentication frequency of $\frac{1}{5}$. This time, however, messages are authenticated once every 500 ms (since, as discussed before, ACC's sampling period is 100 ms). Therefore, from the start of the maneuver to 500 ms after, the attacker is assumed to successfully inject the relative distance values of 0.04 m, 0.16 m, 0.36 m, and 0.64 m at the times 100 ms, 200 ms, 300 ms, and 400 ms respectively. These are all in accordance with (5) and, hence, pass the plausibility check and are processed by the ACC system. At time 500 ms, an authenticated message arrives from the radar sensor containing the actual relative distance, such that the ACC system recovers to some extent. The attacker then continues injecting altered values successfully, however, periodically arriving authenticated messages allow bringing the system back to normalcy. In this case, note that the actual relative distance between vehicles is still maintained remarkably close to the case of no attack, as shown in Fig. 11.

Further, increasing the time duration between two authenticated messages, i.e., decreasing PAE's authentication frequency, decreases the relative distance between vehicles. As observed in Fig. 11, clearly, when authenticating only every 100th message, injected false values by the attacker cannot be compensated well. In this case, the actual relative distance can become as short as 23 m, almost half of the relative distance of 40 m under no attack. On the other hand, as discussed before, the higher the authentication frequency, the more the induced overhead, which may also lead to malfunction due to missing deadlines on the CAN bus. A good trade-off in the ACC case study seems also to be given by authentication frequencies between $\frac{1}{5}$ and $\frac{1}{10}$, similar to the results shown in the emergency braking case study. Finally, note that the relative distance drops down to zero, i.e., a collision takes place, if no authentication is carried out.

Hardening PAE: We now suggest one modification toward further hardening PAE against cyberattacks such that negative effects become minimal.

Now, since most control systems are based on samples sent on a periodic basis, we can use that fact to our advantage and define *reception windows* for each concerned message based on the sample period, jitter and transmission delay. More specifically, unauthenticated messages arriving outside a certain *reception window* can be discarded without further ado. Even if one configures a relatively small reception window, in the worst case, the attacker might still manage to send its messages within it; however, in any case, it will be more difficult this time.

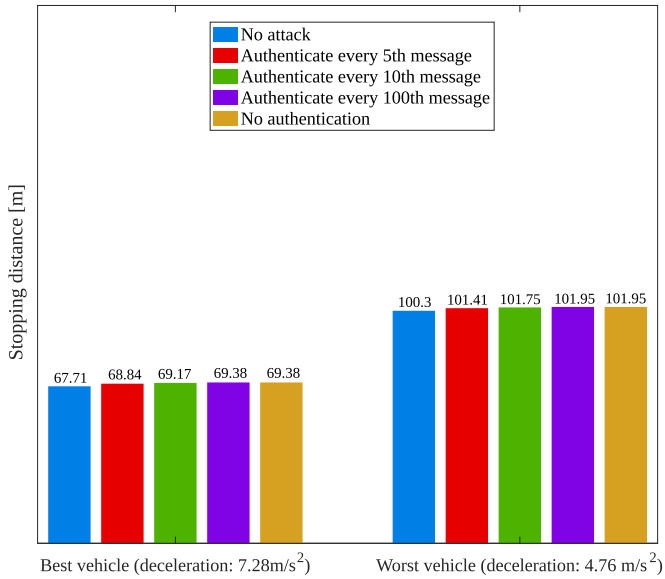

Fig. 12. Impact on stopping distances of two vehicles types when averaging valid and malicious deceleration values.

Let us assume that one message from the attacker and, by design, also the valid message from the corresponding sensor get through the reception window.[7] For example, taking contention/jitter on the bus into account, the brake ECU in the emergency braking case study waits every 20 ms (i.e., every time a new sample from the accelerometer is expected) for 2 ms to 3 ms for messages to arrive. If messages arrive after this time has elapsed, they will be discarded. Once arrived, an average of the deceleration values from the accelerometer and the attacker is computed (rather than processing only the attacker's injected value as discussed before). Clearly, whenever an authenticated message is available, its corresponding value is used and no average needs to be computed. A similar *reception window* can be defined for the ACC case study too.

This rather minor modification significantly reduces the negative impact of cyberattacks on the stopping distance of vehicles, particularly, in the emergency braking case study as shown in Fig. 12. Here, the variation in stopping distance between the cases of no cyberattacks and a $\frac{1}{100}$ authentication frequency (i.e., with authenticated messages being sent once every 2 s) is below 2 m. This applies for both vehicle types under consideration. For example, the worst vehicle under no cyberattack can come to a standstill in 100.30 m. Even under a cyberattack, where an authenticated message is available only once every 2 s, the stopping distance increases only to 101.95 m (in comparison to 157.71 m — see the 4$^{\text{th}}$ bar from the left for the worst vehicle in Fig. 9).

[6]Note that, if the attacker would alter the relative distance to a lower value, the ACC spacing control would be enabled to reduce the speed of ego car until reaching a safe distance. As a result, the actual relative distance is rather kept in a safe range with less or no negative impact on the vehicle's safety.

[7]In principle, the attacker can also send multiple messages, flooding the bus, with the purpose of catching the reception window; however, such a procedure would rather lead to a DoS than a replay/spoofing attack, which as such is easy to detect.

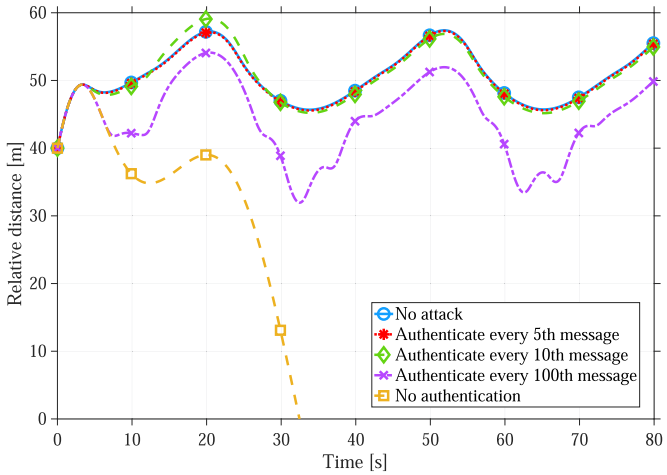

Fig. 13. Relative distance between lead & ego car when averaging valid and malicious distance values.

Note that the benefit of a higher authentication frequency is not as prominent as for emergency braking. However, it should be considered that the experiments in both Figs. 12 and 13 were carried out under the premise that only one malicious message arrives with each sample and/or that reception windows are small enough. If this is not the case and multiple malicious messages breach through, the resulting average deceleration or relative distance, depending on the case study, would rather approximate that in the malicious messages. As a consequence, the resulting stopping distances or relative distances would approximate those in Figs. 9 or 11 respectively, where higher authentication frequencies lead to significantly better results.

D. Summary of Findings

For both synthetic and real-world data, we achieve the most significant improvement, i.e., a greatest reduction of overhead/transmission delay on the bus, when decreasing the authentication frequency from $\frac{1}{1}$ to $\frac{1}{2}$. This holds true for both the average and the longest delay. For an authentication frequency from $\frac{1}{2}$ to $\frac{1}{10}$, this improvement gradually decreases, saturating for an authentication frequency of $\frac{1}{100}$ and below.

The degree of the performance improvement depends on the overall bus utilization. In particular, the higher the bus utilization, the greater the improvement that is achievable by varying the authentication frequency. This is because, at a high utilization, the frequency of messages transmitted on the bus is already high. Here, sending two frames (i.e., an additional frame with an authentication tag) for each such message, or only for a few of them, has a considerable impact on the transmission delay.

On the other hand, based on our case study consisting of emergency braking and ACC, we demonstrated plausibility checks in terms of an acceptable/plausible range of values. We showed that, even though it is possible to mitigate cyberattacks by ignoring values outside the plausible range, in the worst case, an attacker can still breach through. In the emergency braking case study, breaching cyberattacks can increase the stopping distance by several meters, endangering the vehicle's passengers and other road participants. In such extreme situations, the proposed hardening technique that establishes a reception window for messages to arrive and computes the average deceleration value from the available messages (i.e., from the actual and the malicious one), considerably restricts the impact of cyberattacks on the vehicle's stopping distance to less than a couple of meters.

In the ACC case study, even though the proposed hardening technique also improves the relative distance (especially for a low authentication frequency), this is less effective due to the nature of the plausibility check in place. Whereas the possible variation between consecutive accelerometer readings is bounded for emergency braking, ACC's increase in relative distance varies over time. That is, even though the acceleration difference between cars is bounded, the speed difference grows with time from the last authentication. In other words, the longer the time elapsed, the greater the speed difference and, hence, the greater the increase in relative distance between two authenticated samples. Depite hardening, the authentication frequency continues to play an important role for ACC.

E. Discussion

Overhead on ECUs: As discussed in Section V-B, we consider plausibility checks that consist in verifying that signal/sensor values are within a given range or below a given threshold. This can be done with negligible, though admittedly non-zero, overhead on most ECUs. Assuming that such a verification takes a constant amount of time C on a particular ECU, it will then take $n \cdot C$ time to process n signals on that ECU, which results in a linear complexity, i.e., $\mathcal{O}(n)$. Whether this is acceptable or not needs to be decided by the designer.

Recognizing authenticated messages: There are different ways to distinguish between authenticated and non-authenticated, i.e., intermediate messages.

First, one can use separate (different) IDs for authentication messages, so that the receiver can easily distinguish them from intermediate messages. For example, one can use the ID mirroring approach proposed in [42].

Second, one can implement a watchdog on the receiver, which counts the time to the next authenticated message rather than the number of messages. Once this time has elapsed, the receiver would expect an authentication message to arrive. Since the watchdog has to compensate for time deviations, e.g, due to clock drift, the receiver will have to check all messages arriving within a given time window. This may have a potentially non-negligible impact on the computational overhead, if the attacker send multiple messages within that time window. However, on the one hand, we expect the time window to be rather small. On the other hand, we expect the attacker's messages to have a rather low arrival frequency, since otherwise this attack will resemble a DoS attack, for which other defense techniques are necessary, see Section III.

ACC case study: There are certain scenarios when plausibility checks may lead to a deviation from the expected ACC behavior. For example, the plausibility check discussed above may reject legit messages, if i) the lead car leaves or ii) a new car joins

the lane to become the new lead, since sensor values may fall outside the accepted range. If the latter is the case, clearly, the most (legit) messages will be rejected, if i) or ii) occurs right after one authenticated message has been received. This results in the greatest possible deviation from the expected ACC behavior until the next authentication message arrives.

Assuming a $1/10$ authentication frequency, this translates into a time interval $\Delta t = 1\ s$ (as per ACC's sampling period of 100 ms), in which ACC deviates from its expected behavior. In particular, ACC should have accelerated or decelerated at a rate a right after i) or ii) occurs and should have covered a distance d_{ACC}, which is given by: $d_{ACC} = v \cdot \Delta t + \frac{a \cdot \Delta t^2}{2}$. Here, v is the speed at which the ego car is traveling when i) or ii) occurs.

As a result of the proposed periodic authentication, the ego car covers a distance d_{act} that is $|\Delta d|$ lower or larger than d_{ACC} – depending on whether it should have accelerated or decelerated. $|\Delta d|$ is given by: $\Delta d = d_{act} - d_{ACC} = \frac{\Delta a \Delta t^2}{2}$. Δa is the difference between the actual and the expected acceleration/deceleration.

The worst-case deviation occurs when ACC is caught at a transition accelerating or decelerating at its maximum when i) or ii) occurs. We then have that $|\Delta a| = 6\ m/s^2$, i.e., the ego vehicle has to switch from decelerating at $-3.5\ m/s^2$ to accelerating at $2.5\ m/s^2$, and the other way round depending on whether i) or ii) is considered. As a result, the worst-case deviation from the expected trajectory is $|\Delta d| = 3$ m, which is actually quite less compared to the 40 m distance that is usually maintained. Problems will only arise when a third car enters the lane with an already unsafe distance.

It should be noted that this deviation is less critical in i) than in ii), since ii) may lead to a collision. However, this should be analyzed at design time in more detail. If necessary, the designer can further increase the authentication frequency. For example, an authentication frequency of $1/5$ reduces the worst-case deviation to $|\Delta d| = 1.5$ m.

VII. Conclusion

In this paper, we proposed a safety/security co-design technique that we denominate *Periodically Authenticated Encryption* (PAE), which is able to protect CAN buses against sniffing, replay, and spoofing attacks. The proposed technique does not modify CAN and, hence, it does not incur any additional costs compared to similar approaches from the literature. The idea is that altered messages from an attacker can be detected by plausibility checks, for instance, when a predefined value range is exceeded. This leads to the fact that not all messages require authentication, but only those that would otherwise fail to pass plausibility checks. We introduced the concept of authentication frequency, which states how often messages are authenticated under PAE. The higher this frequency is, the higher the level of security achieved, but also the higher the overhead produced on the bus, compromising timing/safety. However, on the other hand, such a low frequency makes the system vulnerable to cyberattacks. Based on a number of experiments, we showed that an authentication frequency between $\frac{1}{2}$ and $\frac{1}{10}$, i.e., authenticating every second to every tenth message on CAN, already leads to a good trade-off between safety and security.

References

[1] A. Aloseel, H. He, C. Shaw, and M. A. Khan, "Analytical review of cybersecurity for embedded systems," *IEEE Access*, vol. 9, pp. 961–982, 2021.

[2] D. Wang and S. Ganesan, "Automotive network security," in *Proc. IEEE Int. Conf. Electro Inf. Technol.*, 2021, pp. 193–196.

[3] H. Zhang, Y. Pan, Z. Lu, J. Wang, and Z. Liu, "A cyber security evaluation framework for in-vehicle electrical control units," *IEEE Access*, vol. 9, pp. 149690–149706, 2021.

[4] A. V. Herrewege, D. Singelee, and I. Verbauwhede, "CanAuth - A simple backward compatible broadcast authentication protocol for CAN bus," in *Proc. Embedded Secur. Cars Conf.*, 2011, pp. 20–26. [Online]. Available: https://www.escar.info/history/escar-europe/escar-europe-2011-lectures-andprogram-committee.html

[5] S. Fassak, Y. E. H. E. Idrissi, N. Zahid, and M. Jedra, "A secure protocol for session keys establishment between ECUs in the CAN bus," in *Proc. IEEE Int. Conf. Wireless Netw. Mobile Commun.*, 2017, pp. 1–6.

[6] H. Ueda, R. Kurachi, H. Takada, T. Mizutani, M. Inoue, and S. Horihata, "Security authentication system for in-vehicle network," *SEI Tech. Rev.*, vol. 81, pp. 5–9, 2015.

[7] Z. King and S. Yu, "Investigating and securing communications in the controller area network (CAN)," in *Proc. Int. Conf. Comput. Netw. Commun.*, 2017, pp. 814–818.

[8] J. Yeom and S. Seo, "A methodology of CAN communication encryption using a shuffling algorithm," in *Proc. IEEE Int. Conf. Connected Auton. Driving*, 2020, pp. 34–38.

[9] A.-I. Radu and F. D. Garcia, "LeiA: A lightweight authentication protocol for CAN," in *Proc. Eur. Symp. Res. Comput. Secur.*, 2016, pp. 283–300.

[10] W. A. Farag, "CANTrack: Enhancing automotive CAN bus security using intuitive encryption algorithms," in *Proc. IEEE Int. Conf. Model. Simul. Appl. Optim.*, 2017, pp. 1–5.

[11] T. Matsumoto, M. Hata, M. Tanabe, K. Yoshioka, and K. Oishi, "A method of preventing unauthorized data transmission in controller area network," in *Proc. IEEE Veh. Technol. Conf.*, 2012, pp. 1–5.

[12] B. Groza, S. Murvay, A. V. Herrewege, and I. Verbauwhede, "LiBrA-CAN: A lightweight broadcast authentication protocol for controller area networks," in *Proc. Int. Conf. Cryptol. Netw. Secur.*, 2012, pp. 185–200.

[13] O. Hartkopp, C. Reuber, and R. Schilling, "MaCAN - Message authenticated CAN," in *Proc. Escar Conf. Embedded Secur. Cars*, 2012.

[14] C. P. Szydlowski, "CAN specification 2.0: Protocol and implementations," in *Proc. Future Transp. Technol. Conf. Expo.*, 1992.

[15] R. G. Underwood, *Cryptography for Secure Encryption*. Berlin, Germany: Springer, 2022.

[16] J. Schwenk, *Guide to Internet Cryptography: Security Protocols and Real-World Attack Implications*. Berlin, Germany: Springer, 2022.

[17] J. Daemen and V. Rijmen, *The Design of Rijndael - The Advanced Encryption Standard (AES)*. Berlin, Germany: Springer, 2020.

[18] L. Yue, Z. Li, T. Yin, and C. Zhang, "CANCloak: Deceiving two ECUs with one frame," in *Proc. Int. Workshop Automot. Auton. Veh. Secur.*, 2021, pp. 17–22. [Online]. Available: https://www.proceedings.com/content/060/060892webtoc.pdf

[19] M. Takada, Y. Osada, and M. Morii, "Counter attack against the bus-off attack on CAN," in *Proc. IEEE Asia Joint Conf. Inf. Secur.*, 2019, pp. 96–102.

[20] G. Bloom, "WeepingCAN: A stealthy CAN bus-off attack," in *Proc. Int. Workshop Automot. Auton. Veh. Secur.*, 2021, pp. 25–30. [Online]. Available: https://www.proceedings.com/content/060/060892webtoc.pdf

[21] D. Souma, A. Mori, H. Yamamoto, and Y. Hata, "Counter attacks for bus-off attacks," in *Proc. Saf. Secur. Rel. Crit. Comput. Appl. Workshops*, 2018, pp. 319–330.

[22] K.-T. Cho and K. G. Shin, "Error handling of in-vehicle networks makes them vulnerable," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 1044–1055.

[23] O. Ikumapayi, H. Olufowobi, J. Daily, T. Hu, I. C. Bertolotti, and G. Bloom, "CANASTA: Controller area network authentication schedulability timing analysis," *IEEE Trans. Veh. Technol.*, vol. 72, no. 8, pp. 10024–10036, Aug. 2023.

[24] S. Nürnberger and C. Rossow, "–vatiCAN– Vetted, authenticated CAN bus," in *Proc. Cryptographic Hardware Embedded Syst.*, 2016, pp. 106–124.

[25] J. Van Bulck, J. T. Mühlberg, and F. Piessens, "VulCAN: Efficient component authentication and software isolation for automotive control networks," in *Proc. 33rd Annu. Comput. Secur. Appl. Conf.*, 2017, pp. 225–237.
[26] A. Thangarajan, M. Ammar, B. Crispo, and D. Hughes, "Towards bridging the gap between modern and legacy automotive ECUs: A software-based security framework for legacy ECUs," in *Proc. IEEE 2nd Connected Autom. Veh. Symp.*, 2019, pp. 1–5.
[27] Z. Lu, Q. Wang, X. Chen, G. Qu, Y. Lyu, and Z. Liu, "LEAP: A lightweight encryption and authentication protocol for in-vehicle communications," in *Proc. Intell. Transp. Syst. Conf.*, 2019, pp. 1158–1164.
[28] H. Noura, O. Salman, A. Chehab, and R. Couturier, "Efficient and secure keyed hash function scheme based on RC4 stream cipher," in *Proc. IEEE Symp. Comput. Commun.*, 2020, pp. 1–7.
[29] M. Ring, J. Dürrwang, F. Sommer, and R. Kriesten, "Survey on vehicular attacks - Building a vulnerability database," in *Proc. Int. Conf. Veh. Electron. Saf.*, 2015, pp. 208–212.
[30] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, vol. 2015, no. S 91, pp. 1–91, 2015.
[31] M. Ring, R. Kriesten, and F. Kargl, "Plausibility checks in electronic control units to enhance safety and security," *Int. J. Adv. Secur.*, vol. 10, no. 1&2, pp. 126–133, 2017. [Online]. Available: https://www.iariajournals.org/security/sec_v10_n12_2017_paged.pdf
[32] K. Beckers, J. Dürrwang, and D. Holling, "Standard compliant hazard and threat analysis for the automotive domain," *Inf.*, vol. 7, 2016, Art. no. 36.
[33] J. Y. Wong, *Theory of Ground Vehicles*. Hoboken, NJ, USA: Wiley, 2001.
[34] L. Elmorshedy, B. Abdulhai, and I. Kamel, "Quantitative evaluation of the impacts of the time headway of adaptive cruise control systems on congested urban freeways using different car following models and early control results," *IEEE Open J. Intell. Transp. Syst.*, vol. 3, pp. 288–301, 2022.
[35] O. Gönüldinc and S. Hölzel, "Adaptive aerodynamik–innovation des porsche 911 turbo," in *Karosseriebautage Hamburg*, Fachmedien Wiesbaden: Springer, 2014, pp. 159–174. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-658-05980-4_18
[36] M. Althoff, S. Maierhofer, and C. Pek, "Provably-correct and comfortable adaptive cruise control," *IEEE Trans. Intell. Veh.*, vol. 6, no. 1, pp. 159–174, Mar. 2021.
[37] M. Zhang, P. Parsch, H. Hoffmann, and A. Masrur, "Analyzing CAN's timing under periodically authenticated encryption," in *Proc. Des. Autom. Test Europe Conf. Exhib.*, 2022, pp. 620–623.
[38] J. Matsumura, Y. Matsubara, H. Takada, M. Oi, M. Toyoshima, and A. Iwai, "A simulation environment based on OMNeT for automotive CAN–Ethernet networks," in *Proc. Int. Workshop Anal. Tools Methodol. Embedded Real-Time Syst.*, 2013, pp. 1–6. [Online]. Available: http://retis.sssup.it/waters2013/WATERS-2013-Proceedings.pdf
[39] K. Kawahara, Y. Matsubara, and H. Takada, "A simulation environment and preliminary evaluation for automotive CAN-Ethernet AVB networks," *1st OMNeT++ Community Summit*, Sep. 2014.
[40] M. Zhang and A. Masrur, "Improving timing behavior on encrypted CAN buses," in *Proc. IEEE Int. Conf. Embedded Real-Time Comput. Syst. Appl.*, 2019, pp. 1–6.
[41] R. Buttigieg, M. Farrugia, and C. Meli, "Security issues in controller area networks in automobiles," in *Proc. IEEE Int. Conf. Sci. Techn. Autom. Control Comput. Eng.*, 2017, pp. 93–98.
[42] M. Zhang and A. Masrur, "Improving timing behavior on encrypted CAN buses," in *Proc. IEEE 25th Int. Conf. Embedded Real-Time Comput. Syst. Appl.*, 2019, pp. 1–6.

Mingqing Zhang received the M.S. degree in applied computer science from TU Chemnitz, Chemnitz, Germany and the B.S. degree in software engineering from Xiamen University, Xiamen, China. He has been a Research Associate with TU Chemnitz since 2018. His research interests include cyber-security, cyber-physical systems, and autonomous driving.

Dharshan Krishna Murthy received the Ph.D. degree in cooperative and autonomous driving from TU Chemnitz, Chemnitz, Germany, in 2021 prior to joining industry. He received the M.S. degree in automotive software engineering from TU Chemnitz. His research interests include cyber-physical systems, brake-by-wire controllers, and autonomous driving.

Philip Parsch received the M.S. degree in electrical and information engineering from TU Munich, Germany, in 2013, the Ph.D. degree in the field of embedded system and communications from TU Chemnitz, Chemnitz, Germany, in 2019, and finished the Postdoc in the field of adaptive wireless communication from the University of Chicago, Chicago, IL, USA, before joining industry.

Henry Hoffmann received the B.S. degree in mathematical sciences from the University of North Carolina at Chapel Hill, Chapel Hill, NC, USA and the M.S. and Ph.D. degrees in electrical engineering and computer science from MIT, Cambridge, MA, USA. Since 2018, he has been a Professor with the University of Chicago, Chicago, IL, USA. His research interests include adaptive techniques for power, energy, accuracy, and performance management in computing systems.

Philipp H. Kindt received the Ph.D. degree in electrical Engineering from the Technical University of Munich, Munich, Germany, in 2019. He was an Assistant Professor of pervasive computing systems WITH the TU Chemnitz, Chemnitz, Germany until 2022 before joining industry. His research interests include wireless communication, mobile computing, and the IoT.

Alejandro Masrur received the M.S. degree in electrical engineering from Universidad Tecnolagicagica Nacional, Various, Argentina, in 2002 and the Ph.D. degree in electrical and information engineering from TU Munich, Munich, Germany, in 2010 and. He is a Professor at the Department of Computer Science at TU Chemnitz, Germany. His research interests include real-time systems and scheduling, high-level design of embedded systems, and automotive software among others.