AutoSec: Secure Automotive Data Transmission Scheme for In-Vehicle Networks

Trupil Limbasiya
iTrust, Singapore University of Technology and Design
Singapore
limbasiyatrupil@gmail.com

Amrita Ghosal
Department of Computer Science and Information Systems, University of Limerick
Limerick, Ireland
ghosal.amrita@gmail.com

Mauro Conti
Department of Mathematics, University of Padua
Padua, Italy
conti@math.unipd.it

ABSTRACT

Modern vehicles comprise hundreds of Electronic Control Units (ECUs) and sensors for enhancing numerous security and comfort-related functionalities. The ECUs perform real-time information exchange, such as automotive instructions over the Controller Area Network (CAN) bus. However, the CAN bus architecture supports very limited security features. Thus, in-vehicle communications over CAN are vulnerable to critical security threats. Also, as ECUs are resource-constrained in nature, the continuous message transmissions lead to drain out of energy during inter-ECU communication if the authentication scheme is not cost-effective.

This paper proposes *AutoSec*, a lightweight scheme, exploiting low-cost bit-wise XOR and concatenation operations to facilitate secure and efficient in-vehicle communications for connected vehicles. We show through qualitative analysis that AutoSec preserves the security properties of message integrity, user authentication and message confidentiality. We implemented AutoSec on Raspberry Pi 3B+ and performed exhaustive experiments to validate the security robustness and lightweightness of AutoSec. The results show that AutoSec reduces the computation time by $\sim$ 99% and energy consumption by $\sim$ 99%.

CCS CONCEPTS

• **Security and privacy → Security protocols**; **Symmetric cryptography and hash functions**.

KEYWORDS

Controller area network, Electronic control unit, In-vehicle networks, User authentication, Message integrity, Vehicle security;

ACM Reference Format:
Trupil Limbasiya, Amrita Ghosal, and Mauro Conti. 2021. AutoSec: Secure Automotive Data Transmission Scheme for In-Vehicle Networks. In *ICDCN '22: ACM International Conference on Distributed Computing and Networking, January 04–07, 2022, India.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/1122445.1122456

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
ICDCN '22, January 04–07, 2022, India
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

The Controller Area Network (CAN) [1] bus technology has become quite popular in the Intra-Vehicular Networks (IVNs) through its wide-spread deployment. The CAN provides for broadcast communication between the different Electronic Control Units (ECUs) in modern vehicles. Technological developments in recent years have allowed modern vehicles for accessing cloud services and communication with other vehicles using mobile cellular connections. These interfaces not only provide useful services but may also introduce new attack surfaces, leading to enhanced security vulnerabilities for the vehicle ECUs. Through the compromised ECU, the attacker is able to take control of the vehicle that may result in serious consequences, e.g., the attacker can alter the speed of the vehicle or stop the vehicle altogether [2].

Research works [3] presented that unavailability of security mechanisms for CAN have resulted in the exposure of IVNs to the attackers. For example, researchers ran experiments on a Jeep Cherokee to demonstrate that the remote attacker can easily send forged messages from the compromised ECUs, that enables it to take charge of the different vehicular functionalities [4], [5]. Another such work [6] revealed the security threats in different BMW models, including the capability of controlling the connected ECUs (via CAN) remotely. The CAN is primarily susceptible to attacks by adversaries due to the absence of encryption techniques and inadequate access control. Also, the broadcast nature of the CAN bus makes it more vulnerable to attacks and access control. The communication between the ECUs and the external networks should be designed using robust cryptographic techniques, leading to safeguarding the IVNs from both internal as well as external adversaries.

For in-vehicle communication, ECUs broadcast messages over the CAN bus for the management of automotive operations in autonomous cars [7]. Further, autonomous vehicles are connected to different components, such as other cars, pedestrians, wireless sensors, and other smart devices through the dedicated short-range communication (DSRC), 4G/5G, or Wi-Fi technology for smooth movement of vehicles on the road. Therefore, it becomes more important to deliver crucial messages to an ECU for further action(s) by preserving vital security properties, otherwise it may lead to severe damage to in-vehicle automotive operations [8]. Hence, in this work, we aim to address the important security challenges for in-vehicle message transmissions, such as (i) verification of sender (ii) message confidentiality (iii) data correctness. We identify the key security problems associated with CAN, as follows.

- Since the identity of a sender is unknown, and the sender's authenticity cannot be verified in CAN, an adversary can

masquerade any ECU with a similar message identifier. Hence, each ECU connected with the CAN bus can apply a masquerade attack in the system to affect the vehicle functions [9], [10].
- ECUs are connected over the CAN bus for supporting various automotive services in a vehicle. And CAN is associated with wireless interfaces over a gateway ECU that enables adversaries for launching attacks without gaining access to a vehicle, as demonstrated in [5], [6], [11]. Furthermore, it cannot be guaranteed that the compromised/other ECUs are always monitored in CAN. Hence, an adversary can send false messages, misleading the CAN bus [12].
- Messages are broadcast in CAN without using any encryption. All ECUs connected with the CAN bus can listen to any of the messages. This may lead to data confidentiality problems or allow adversary to inject false data into the system [13].

Considering the security shortcomings associated with CAN, an adversary can make an attempt of stealing cars, deactivating automotive systems, sending erroneous messages, and obscuring defective functionalities [3]. Therefore, it is essential to preserve the important security properties during message exchange in autonomous vehicles.

Contributions: We propose a secure lightweight scheme for CAN data transmission in autonomous vehicles. Our contributions in this paper are three folds.

- We propose a secure lightweight scheme, AutoSec for secure data transmission over CAN that preserves the security properties of message integrity, user authentication and message confidentiality. For efficient computation and verification, we use SHA-512, bit-wise XOR, and concatenation on ECUs having limited processing power.
- We analyze AutoSec to confirm its security robustness to satisfy message integrity, user authentication, message confidentiality, and session key attacks.
- We evaluate the performance of AutoSec in terms of the overheads, i.e, computation time, energy consumption, and communication overhead. Experimental results show that AutoSec significantly improves computation time and energy consumption compared to other relevant existing works.

Organization: The rest of the work is organized as follows. Section 2 discusses the related works. Section 3 presents the system overview, the adversary model and the security requirements of the proposed protocol AutoSec. We introduce the proposed scheme AutoSec in Section 4. Section 5 provides a comprehensive security analysis of AutoSec. In Section 6, we present the performance evaluation of AutoSec. Finally, we conclude the work in Section 7.

2 RELATED WORKS

Many schemes were reported with an objective for securing the communication in IVNs. We present here some of the works more relevant in our context.

Ying et al. [14] proposed a scheme named TACAN (Transmitter Authentication in CAN) for providing secure authentication of the ECUs on the CAN bus using the concept of covert channels. Here no additional bits or CAN messages are used and there are no modifications in the CAN protocol. TACAN utilizes the concept of covert channel for developing a defensive technique for enabling transmitter authentication by means of a centralized, trusted monitor node. TACAN is composed of three covert channels that are used for ECU authentication. The authors implemented TACAN on the University of Washington's EcoCAR (Chevrolet Camaro 2016) testbed [15]. Thorough evaluation was also done considering the bit error, throughput and detection performance of TACAN. The results demonstrate that TACAN has high effectiveness in detecting CAN bus attacks and attesting the general functionalities of ECUs. As this scheme is based on shared cryptographic keys and inter-arrival times, which lead to computational overhead. Also, the security level is reduced that makes it insufficient for present day requirements.

Frame authentication in CAN is usually likely in a restrictive manner mainly due to decreased bandwidth, low payload and constrains for computational resources. To overcome this problem, the authors in [3] proposed a method where the identity of the actual sender was determined through observation in differences of the CAN signal. This scheme has the ability of lessening the required resources to a significant extent and having high identification rates of 99.98%. The authors compared their scheme with the most lightweight scheme to produce reduced memory footprints and computational requirements. The scheme also has the capability of adapting to incremental signal changes during operation. This work has been evaluated on a prototype as well as two production vehicles under changing conditions over a time frame of one week using various electronic consumers. However, this scheme requires a powerful analog-to-digital converter and high computation capabilities. This significantly enhances the implementation costs, that has an important impact on the automotive industry.

Palaniswamy et al. [18] analyzed the present frame-level authentication protocol for the CAN bus for identifying the weaknesses and the limitations. The authors provide a protocol suite for entity authentication, key management, secure message flow for remote transmission request frames and session key update required for vehicle communication with external devices. The security of the proposed protocol is determined using the random oracle model and judges its defensive capabilities against known attacks. The simulation results demonstrate the efficiency of the protocol compared to other existing schemes.

The authors in [8] use a real vehicle and malicious smartphone application for demonstrating that a long-range wireless attack is feasible in a connected car environment. They propose a security protocol for CAN that is designed according to the present CAN specifications. The proposed security protocol is evaluated using CANoe software and a DSP-F28335 micro-controller. The evaluation results show the effectiveness of the proposed security protocol with respect to authentication delay and communication load. The main disadvantage of this technique is that as the application runs on a mobile device, therefore, there is a possibility that attacks can be launched through the cellular network.

The work in [16] proposes a novel and efficient scheme during the system design stage for providing optimal security and safety. The design optimization guarantees that execution of every real time applications takes place within the deadline as well as reduction in the number of transmission messages over the CAN bus. After the optimization operation, the authors apply a hash message

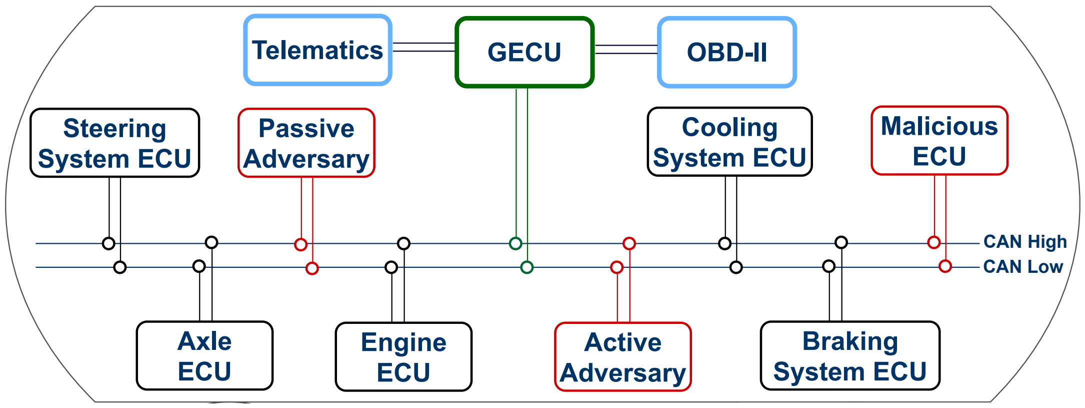


Figure 1: System model for automotive in-vehicle communication over the CAN bus.

authentication code on particular messages for ensuring secure communication between the ECUs and protection against cyber attacks. The security analysis together with the experimental results prove the efficacy of the proposed scheme in terms of countering attacks on the CAN bus in a timely manner. Though this technique uses selective message authentication that can reduce communication overhead on automotive CAN, but introduces delay generating from the transmission of an extra packet.

Groza et al. [17] authenticate the identifiers of the CAN frames using an ordered CMAC buffer and verify the legitimacy of the sender node. Also, the authors consider the real-world scenarios and show that the achieved security level is very close to the length of the ID field in spite of the constraints arising due to ordering. The procedure adapted in [17] is able to evade replay attacks as well fuzz testing on the bus. The authors carry out practical implementations on automotive-grade microcontrollers and CAN-bus traffic allotments from a high-end vehicle. The computational requirements [17] for securing the CAN bus are also quite low.

3 SYSTEM OVERVIEW

In this section, we present the system model and background knowledge on which we base our work. Specifically, Section 3.1 discusses the system architecture. We then introduce the adversary model in Section 3.2 with the adversary's objectives and capabilities. Finally, Section 3.3 presents the security requirements.

3.1 System Architecture

The system model is illustrated with an objective to understand the general outline of in-vehicle communications with the functionality of different entities. Figure 1 shows a basic overview of in-vehicle networks in which the Gateway ECU (GECU) and different types of ECUs are connected via a single CAN bus [3], [8], [18], [19]. We describe the role of each component, as follows.

- *GECU*: It is a main ECU that connects the outer world with the in-vehicle networks as intermediaries. And it is connected with telematics and On-Board Diagnostic-II (OBD-II) to perform different functionalities in a vehicle. It acts as the central authority to establish/update the session key among the ECUs and keeps a record of IDs of all the installed ECUs.
- *ECU*: Each ECU collects the data from sensors (attached with automobile parts, such as engine, braking system, gearbox, cooling system, axle, steering system, etc.) to broadcast relevant messages with other ECUs and GECU over the CAN bus. Since an ECU is configured with low processing power, it takes more time to execute advanced cryptographic operations that satisfy a high-level of security.
- *Adversary*: The system mainly includes two types of adversaries, i.e., internal and external, that can introduce passive and active attacks during in-vehicle communications. Here, a malicious ECU means that it is a legal component of the system, but it is attempting to forge messages during in-vehicle communications. Section 3.2 discusses the objectives of the adversaries.

3.2 Adversary Model

CAN messages are broadcast to all ECUs over the CAN bus, and it is demonstrated in [5], [6], [11] that adversaries can connect to the automotive connected system via local or remote access. Accordingly, we consider the following intentions of an adversary, who aims to perform malicious activities on in-vehicle communications.

- An adversary wants to intercept or interrupt the in-vehicle communications for various objectives, such as stop transmitting vital information, trace vehicle users, modify messages, delay real-time data, disseminate erroneous/bogus messages, miss essential messages, access the system messages without being a legal user, and understand communication messages. The ultimate aim of an adversary is to apply remote attacks on in-vehicle communications to disrupt the functionalities.
- An adversary attempts to reduce the performance efficiency of the CAN bus by sending multiple dummy messages over it. Thus, all ECUs become busy for verifying the correctness of the multiple messages received.

We consider the following assumptions, assuming the adversary's security capabilities [1], [8].

- All ECUs are directly connected with a gateway ECU (which is associated with wireless interfaces for purposes of different functionalities) that leverages an adversary to inject erroneous messages into the CAN bus.

- An adversary can periodically send multiple messages over CAN within a short period of time to create confusion, i.e., based on which instructions, the receiver ECU should proceed further. As a result, it leads to a problem of the overwritten messages over the CAN bus.
- An adversary can send multiple messages intentionally in the CAN bus to increase the verification overhead at the receiver. Thus, the ECU gets disconnected from the bus.
- If any malicious ECU is connected to the CAN bus, then it can broadcast frames on the CAN bus. Hence, an adversary can launch different malicious activities on in-vehicle network.
- We consider that an adversary knows the design and specifications of the targeted automotive system.

3.3 Security Requirements

Our objective is preserving the important security properties of in-vehicle communication by achieving authentication, integrity, and confidentiality. This is attained through the secret group key communication and unmodifiable identifier to prove the sender while computing messages using one-way hash function to preserve integrity and confidentiality. Considering these security properties, the proposed scheme can protect from unauthorized access of messages, message alteration, and data impersonation. We discuss the importance of different security properties in CAN, as follows.

- **Authentication:** When messages are sent into the CAN, the receiver extracts the sender through the sender ID (present in the data frame), and there is no verification mechanism to confirm the sender in CAN. If any compromised/malicious ECU has used the sender ID of another ECU for forgery, then also it is difficult to know the original sender of the forged messages. Hence, authentication plays a critical role in CAN for verifying the sender of the messages to prevent transmission of illegal messages.
- **Integrity:** Messages are sent in plain-text in CAN, and only CRC checksum is not adequate to check the correctness of CAN messages because an adversary can manage to replace the CRC information correspondingly. As a consequence, ECUs may follow modified messages to control the functionalities, resulting in accidents. Hence, it is better to confirm the accuracy of the CAN messages before proceeding.
- **Confidentiality:** Messages are broadcast on the CAN as per the CAN standard. Thus, at least all ECUs connected to the CAN bus get these transmitted messages, which leads to the disclosure of important information. Therefore, the system should maintain the message confidentiality while sending it over the CAN bus.

4 THE PROPOSED SCHEME: AUTOSEC

Autonomous vehicles are fully dependent on the on-time and accurate functionality of automotive operations. Thereby, it is indispensable to transmit crucial information (i.e., automotive instructions) securely and efficiently between the GECU and ECUs to quickly take better decision(s) while a vehicle on the move. The existing protocols for in-vehicle communications cannot withstand crucial security issues, such as session key update leakage, presence of authentication attacks, private key storage concern, and encryption key compromising. Moreover, they are designed with comparatively high cost and more number of operations, taking more time for the execution. Modern vehicles are enabled more features, such as telematics, advanced driver assistance systems, and infotainment. Thereby, the required number of ECUs in each vehicle is also increased that needs to perform more computational operations by ECUs. Hence, it is necessary to design the protocol that takes less computation time while performing various operations.

We propose a secure data transmission scheme (named as *AutoSec*) using SHA-512, bit-wise XOR ($\oplus$), and concatenation ($||$) for secure and cost-effective data exchanges, preserving the vital information from different security threats. Since AutoSec is designed to protect in-vehicle message communication with low-cost cryptographic operations, it does not require any hardware modifications and network improvements. Hence, the proposed scheme can be applied to the existing CAN architecture easily. The AutoSec mainly consists of two phases: (i) basic setup and (ii) message communication protocols. We discuss the basic setup procedure for AutoSec in Section 4.1 to generate and load long-term keys in ECUs. In Section 4.2, we provide an elaborate discussion of the secure communication protocol associated with AutoSec. Table 1 describes the various notations used in the design of AutoSec.

Table 1: List of Symbols with its Description

Symbol	Description
ECU_i	i^{th} electronic control unit (ECU)
$GECU$	Gateway ECU
ID_{ECU_i}	Identity of ECU_i
ECU_r	Any one ECU out of all ECUs, who requires some information
ECU_s	Any one ECU out of all ECUs, who sends some information to ECU_r
K_i	Long-term pre-loaded symmetric key between ECU_i and $GECU$
GK	Long-term pre-loaded symmetric key between all $ECUs$ and $GECU$
K_{rs}	Long-term pre-loaded symmetric key between ECU_r and ECU_s
$r_i/s_i/p_i/x_i$	Random nonce
$KDF_x()$	Key derivation function using x as key
EK_i/AK_i	Generated keys from KDF_x for ECU_i to distribute the session key
EK_{rs}/AK_{rs}	Generated keys from KDF_x for communication between ECU_r and ECU_s
$h(\cdot)$	One-way hash function (i.e., SHA-512)
ΔT	The maximum time delay
T_{i1}/T_{i3}	Current time-stamp at ECU_i
T_{i2}/T_{i4}	Current time-stamp at $GECU$
M	Message for data exchange

4.1 AutoSec: Basic Setup

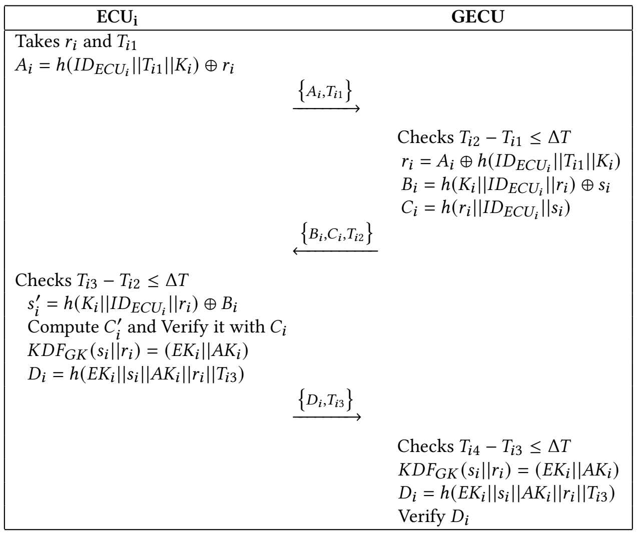
GECU (acting as the central ECU) registers with the Road Transport Authority (RTA) so that it can execute different operations in the in-vehicle networks. A list of IDs and respective long-term keys of all installed ECUs are securely stored in the protected memory of GECU. All ECUs are initially loaded with long-term keys (K_i, GK, and K_{rs}) in tamper-resistant trusted platform modules [20]. To verify the integrity of the firmware, ECU_i computes a firmware digest $H(K_i||ID_{ECU_i}||Image_{Firmware})$ and sends it to the GECU. After receiving it, the GECU confirms the receipt of the firmware digest with the computed value for validation.

4.2 AutoSec: Message Communication Protocol

ECUs are attached with different automobile components of a vehicle to exchange real-time information among the ECUs over the CAN bus for automotive operations. This phase comprises of three protocols: (i) Initial Session Key Computation and Verification Protocol (ISCVP), (ii) Remote Frame Transmission Request Protocol (RFTRP), and (iii) Session Key Update Protocol (SKUP). We describe these protocols in details, as follows.

4.2.1 Proposed ISCVP. Each ECU performs the following steps with the GECU for initial session key computation and verification procedure. Since a vehicle contains multiple ECUs, this process is performed with the GECU in a fixed order. This process is also shown in Figure 2.

(1) ECU_i generates a random nonce (say r_i) to compute $A_i = h(ID_{ECU_i}||T_{i1}||K_i) \oplus r_i$, where T_{i1} is a current time-stamp. ECU_i sends $\{A_i, T_{i1}\}$ as a request to GECU for the initial session key computation and verification.
(2) The GECU checks the freshness of the received request by $T_{i2} - T_{i1} \leq \Delta T$, where T_{i2} is the request receiving time-stamp. If it holds, then the GECU computes $r_i = A_i \oplus h(ID_{ECU_i}||T_{i1}||K_i)$, $B_i = h(K_i||ID_{ECU_i}||r_i) \oplus s_i$, and $C_i = h(r_i||ID_{ECU_i}||s_i)$ to send $\{B_i, C_i, T_{i2}\}$ as a response to ECU_i for the key computation.
(3) ECU_i verifies the validity of a response through $T_{i3} - T_{i2} \leq \Delta T$, where T_{i3} is the response receiving time-stamp. If it holds, then ECU_i calculates $s_i' = h(K_i||ID_{ECU_i}||r_i) \oplus B_i$, $C_i' = h(r_i||ID_{ECU_i}||s_i')$ and confirms $C_i' \overset{?}{=} C_i$. If both are equal, then ECU_i computes $KDF_{GK}(s_i||r_i) = (EK_i||AK_i)$ and $D_i = h(EK_i||s_i||AK_i||r_i||T_{i3})$ to send $\{D_i, T_{i3}\}$ to GECU as the key verification.
(4) The GECU confirms the freshness of $\{D_i, T_{i3}\}$ by calculating $T_{i4} - T_{i3} \leq \Delta T$, where T_{i4} is the key verification message receiving time-stamp. The GECU computes $KDF_{GK}(s_i||r_i) = (EK_i||AK_i)$, $D_i' = h(EK_i||s_i||AK_i||r_i||T_{i3})$ to check the legality of $\{D_i, T_{i3}\}$ by comparing D_i' with D_i. If it holds, then it is considered as a valid key computation.

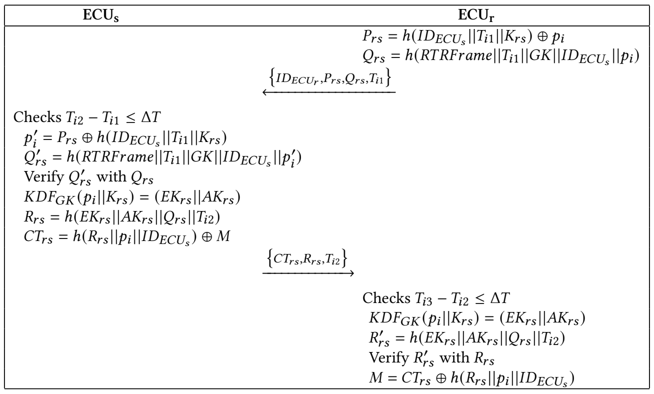


$\mathrm{ECU_i}$
GECU
Takes r_i and T_{i1}
$A_i = h(ID_{ECU_i}||T_{i1}||K_i) \oplus r_i$
$\xrightarrow{\{A_i, T_{i1}\}}$
Checks $T_{i2} - T_{i1} \leq \Delta T$
$r_i = A_i \oplus h(ID_{ECU_i}||T_{i1}||K_i)$
$B_i = h(K_i||ID_{ECU_i}||r_i) \oplus s_i$
$C_i = h(r_i||ID_{ECU_i}||s_i)$
$\xleftarrow{\{B_i, C_i, T_{i2}\}}$
Checks $T_{i3} - T_{i2} \leq \Delta T$
$s_i' = h(K_i||ID_{ECU_i}||r_i) \oplus B_i$
Compute C_i' and Verify it with C_i
$KDF_{GK}(s_i||r_i) = (EK_i||AK_i)$
$D_i = h(EK_i||s_i||AK_i||r_i||T_{i3})$
$\xrightarrow{\{D_i, T_{i3}\}}$
Checks $T_{i4} - T_{i3} \leq \Delta T$
$KDF_{GK}(s_i||r_i) = (EK_i||AK_i)$
$D_i = h(EK_i||s_i||AK_i||r_i||T_{i3})$
Verify D_i

Figure 2: AutoSec: ISCVP Design Description

4.2.2 Proposed RFTRP. When an ECU (say ECU_r) requires some information from another ECU (say ECU_s), the receiver ECU (ECU_r) sends a request to the sender ECU (ECU_s) to establish a connection for data exchange between them. The RFTRP procedure is displayed in Figure 3 to confirm the legitimacy of both (ECU_s and ECU_r) and the messages sent by each before connection for data transmission.

(1) ECU_r takes a random nonce, p_i and computes $P_{rs} = p_i \oplus h(ID_{ECU_s}||T_{i1}||K_{rs})$, $Q_{rs} = h(RTRFrame||T_{i1}||GK||ID_{ECU_s}||p_i)$ to send $\{ID_{ECU_r}, P_{rs}, Q_{rs}, T_{i1}\}$ as a request to ECU_s.
(2) After getting it, ECU_s confirms its freshness through $T_{i2} - T_{i1} \leq \Delta T$, where T_{i2} is the request receiving time-stamp. ECU_s calculates $p_i' = h(ID_{ECU_s}||T_{i1}||K_{rs}) \oplus P_{rs}$ and $Q_{rs}' = h(RTRFrame||T_{i1}||GK||ID_{ECU_s}||p_i')$ to confirm the legality of a request through $Q_{rs}' \overset{?}{=} Q_{rs}$. If it matches, then ECU_s proceeds to compute $KDF_{GK}(p_i||K_{rs}) = (EK_{rs}||AK_{rs})$, $R_{rs} = h(EK_{rs}||AK_{rs}||Q_{rs}||T_{i2})$, and $CT_{rs} = h(R_{rs}||p_i||ID_{ECU_s}) \oplus M$ to send a response as $\{CT_{rs}, R_{rs}, T_{i2}\}$.
(3) ECU_r checks the validity of a response message through $T_{i3} - T_{i2} \leq \Delta T$, where T_{i3} is the response receiving time-stamp. ECU_r computes $KDF_{GK}(p_i||K_{rs}) = (EK_{rs}||AK_{rs})$ and $R_{rs}' = h(EK_{rs}||AK_{rs}||Q_{rs}||T_{i2})$ to verify its legality by matching R_{rs}' with R_{rs}. If both are equal, then ECU_r retrieves a message as $M = CT_{rs} \oplus h(R_{rs}||p_i||ID_{ECU_s})$.



$\mathrm{ECU_s}$
$\mathrm{ECU_r}$
$P_{rs} = h(ID_{ECU_s}||T_{i1}||K_{rs}) \oplus p_i$
$Q_{rs} = h(RTRFrame||T_{i1}||GK||ID_{ECU_s}||p_i)$
$\xleftarrow{\{ID_{ECU_r}, P_{rs}, Q_{rs}, T_{i1}\}}$
Checks $T_{i2} - T_{i1} \leq \Delta T$
$p_i' = P_{rs} \oplus h(ID_{ECU_s}||T_{i1}||K_{rs})$
$Q_{rs}' = h(RTRFrame||T_{i1}||GK||ID_{ECU_s}||p_i')$
Verify Q_{rs}' with Q_{rs}
$KDF_{GK}(p_i||K_{rs}) = (EK_{rs}||AK_{rs})$
$R_{rs} = h(EK_{rs}||AK_{rs}||Q_{rs}||T_{i2})$
$CT_{rs} = h(R_{rs}||p_i||ID_{ECU_s}) \oplus M$
$\xrightarrow{\{CT_{rs}, R_{rs}, T_{i2}\}}$
Checks $T_{i3} - T_{i2} \leq \Delta T$
$KDF_{GK}(p_i||K_{rs}) = (EK_{rs}||AK_{rs})$
$R_{rs}' = h(EK_{rs}||AK_{rs}||Q_{rs}||T_{i2})$
Verify R_{rs}' with R_{rs}
$M = CT_{rs} \oplus h(R_{rs}||p_i||ID_{ECU_s})$

Figure 3: AutoSec: RFTRP Design Description

4.2.3 Proposed SKUP. The session key between ECU_i and $GECU$ is periodically updated to improve the security in the system with a fresh computed session key. However, if we follow the SKUP protocol of [18], then it makes availability of the fresh session key (of ECU_i) to all connected ECUs effortlessly. Hence, we design an enhanced SKUP to address this present issue of the session key update process, as follows. The proposed SKUP is also described in Figure 4.

(1) GECU proceeds to calculate $X_i = h(ID_{ECU_i}||T_{i1}||K_i||GK) \oplus x_i$, $Y_i = h(GK||ID_{ECU_i}||x_i||ID_{GECU})$, and $KDF_{GK}(x_i||K_i) = (EK_{k+1}||AK_{k+1})$ to initiate a procedure for the session key update. After that, GECU sends $\{ID_{ECU_i}, X_i, Y_i, T_{i1}\}$ to ECU_i as the session key update initialization.

(2) ECU_i first checks the validity of $\{ID_{ECU_i}, X_i, Y_i, T_{i1}\}$ through $T_{i2} - T_{i1} \leq \Delta T$, where T_{i2} = initialization message receiving time-stamp. If it holds, then ECU_i computes $x_i = X_i \oplus h(ID_{ECU_i}||T_{i1}||K_i||GK)$, $Y'_i = h(GK||ID_{ECU_i}||x'_i||ID_{GECU})$ to confirm $Y'_i \overset{?}{=} Y_i$. If both are equal, then ECU_i sends $\{Z_i, T_{i2}\}$ as a response to GECU, where $KDF_{GK}(x_i||K_i) = (EK_{k+1}||AK_{k+1})$, $Z_i = h(x_i||EK_{k+1}||T_{i2}||AK_{k+1}||Y_i)$.
(3) Upon the receiving $\{Z_i, T_{i2}\}$, GECU checks its freshness by $T_{i3} - T_{i2} \leq \Delta T$, where T_{i3} = response receiving time-stamp. If it is valid, then only GECU computes $KDF_{GK}(x_i||K_i) = (EK_{k+1}||AK_{k+1})$ and $Z'_i = h(x_i||EK_{k+1}||T_{i2}||AK_{k+1}||Y_i)$ to verify the legality of the updated session key by $Z'_i \overset{?}{=} Z_i$. If it matches, the session key is updated successfully.

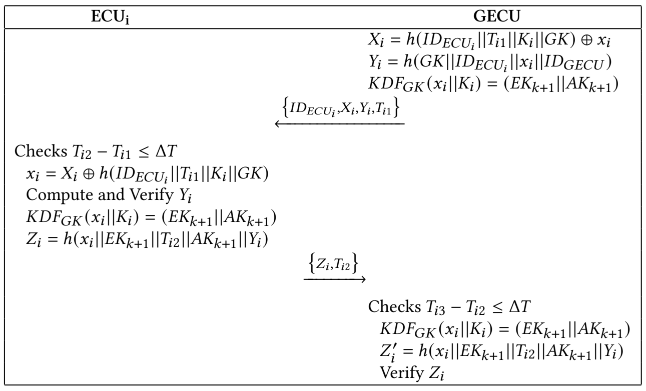

ECUi
GECU
$X_i = h(ID_{ECU_i}||T_{i1}||K_i||GK) \oplus x_i$
$Y_i = h(GK||ID_{ECU_i}||x_i||ID_{GECU})$
$KDF_{GK}(x_i||K_i) = (EK_{k+1}||AK_{k+1})$
$\{ID_{ECU_i}, X_i, Y_i, T_{i1}\}$
Checks $T_{i2} - T_{i1} \leq \Delta T$
$x_i = X_i \oplus h(ID_{ECU_i}||T_{i1}||K_i||GK)$
Compute and Verify Y_i
$KDF_{GK}(x_i||K_i) = (EK_{k+1}||AK_{k+1})$
$Z_i = h(x_i||EK_{k+1}||T_{i2}||AK_{k+1}||Y_i)$
$\{Z_i, T_{i2}\}$
Checks $T_{i3} - T_{i2} \leq \Delta T$
$KDF_{GK}(x_i||K_i) = (EK_{k+1}||AK_{k+1})$
$Z'_i = h(x_i||EK_{k+1}||T_{i2}||AK_{k+1}||Y_i)$
Verify Z_i

Figure 4: AutoSec: SKUP Design Description

5 SECURITY ANALYSIS

We perform security analysis on the proposed protocols to verify their security robustness against crucial security attributes. We discuss the related definitions, theorems, and their respective proofs based on the Random Oracle Model (ROM) for AutoSec. In this, a game is constructed between an adversary ($\mathcal{A}$) and the challenger ($\mathcal{C}$) to confirm whether $\mathcal{A}$ can win a game with a non-negligible probability for the given challenges by $\mathcal{C}$ in polynomial time or not [21]. We also present a comparison table to understand the security features of the existing schemes.

Definition: The proposed protocols are secure in the constructed game if $\mathcal{A}$ has trivial advantage(s) in polynomial time.

RFTRP — Oracle: We consider that $\mathcal{A}$ is acting as $\mathbf{ECU_s}$ and wants to send bogus/modified response message for $\{ID_{ECU_r}, P_{rs}, Q_{rs}, T_{i1}\}$. Thus, $\mathcal{A}$ sends $\{CT^{\mathcal{A}}_{rs}, R^{\mathcal{A}}_{rs}, T^{\mathcal{A}}_{i2}\}$ to $\mathbf{ECU_r}$. In the proposed RFTRP protocol, $\mathbf{ECU_r}$ (acting as $\mathcal{C}$) confirms the freshness of the received message through $T_{i3} - T_{i2} \leq \Delta T$ and verifies the received R_{rs} with the computed R_{rs} for validation.

SKUP — Oracle: We consider that $\mathcal{A}$ is acting as **GECU** to update the session key between $\mathbf{ECU_i}$ and **GECU**. Thus, $\mathbf{ECU_i}$ acts as $\mathcal{C}$ to confirm the connectivity with the original **GECU** while updating the session key. $\mathcal{A}$ sends $\{ID^{\mathcal{A}}_{ECU_i}, X^{\mathcal{A}}_i, Y^{\mathcal{A}}_i, T^{\mathcal{A}}_{i1}\}$ to $\mathbf{ECU_i}$. In the proposed SKUP design, the validity and legality are verified by ΔT and Y_i respectively at $\mathbf{ECU_i}$ side to confirm the authenticity. If it holds, then only $\mathbf{ECU_i}$ sends $\{Z_i, T_{i2}\}$ to the requested ECU. At the receiver side, it is also confirmed to verify the correctness of exchanges parameters.

Theorem-1: The proposed protocols can withstand message integrity based attacks by considering the ROM.

Proof. We assume that an adversary ($\mathcal{A}$) wants to send a modified/bogus message response to $\mathbf{ECU_r}$ (acts as $\mathcal{C}$) for $\{ID_{ECU_r}, P_{rs}, Q_{rs}, T_{i1}\}$ without knowledge of $\mathbf{ECU_s}$ during the RFTRP phase. Therefore, $\mathcal{A}$ should compute $R_{rs}[= h(EK_{rs}||AK_{rs}||Q_{rs}||T_{i2})]$ and $CT_{rs}[= M \oplus h(R_{rs}||p_i||ID_{ECU_s})]$. We consider, an adversary gets P_{rs} and Q_{rs} from a common communication channel, but $\mathcal{A}$ cannot compute K_{rs} (only known to ECU_s and ECU_r) and p_i (unavailability of all required values). Consequently, $\mathcal{A}$ cannot calculate R_{rs} and CT_{rs} correctly. In addition, if $\mathcal{A}$ sends $\{CT^{\mathcal{A}}_{rs}, R^{\mathcal{A}}_{rs}, T^{\mathcal{A}}_{i2}\}$ to $\mathbf{ECU_r}$, then $\mathcal{C}$ verifies through $T_{i3} - T^{\mathcal{A}}_{i2} \leq \Delta T$ and R_{rs} with $R^{\mathcal{A}}_{rs}$. $\mathcal{A}$ specifically fails to prove message legality in $R_{rs} \overset{?}{=} R^{\mathcal{A}}_{rs}$ verification if s/he has done any modification(s) in computation parameters. Thus, $\mathcal{A}$ has no opportunity to do any change(s) during message transmissions in the proposed RFTRP. □

Theorem-2: The proposed schemes preserve the property of user authentication.

Proof. $\mathcal{A}$ can attempt to infringe the authentication property by impersonating either ECU_r or ECU_s in the proposed RFTRP.

To impersonate ECU_r, $\mathcal{A}$ should send valid computed parameters to ECU_s so that the receiving entity can proceed further after the verification. If $\mathcal{A}$ sends forged parameters to ECU_s, then the request is discarded due to the verification failure in $Q'_{rs} \overset{?}{=} Q_{rs}$. In the proposed RFTRP, $\mathcal{A}$ should calculate $P^{\mathcal{A}}_{rs}$, $Q^{\mathcal{A}}_{rs}$ to send $\{ID_{ECU_r}, P^{\mathcal{A}}_{rs}, Q^{\mathcal{A}}_{rs}, T^{\mathcal{A}}_{i1}\}$ to ECU_s, but $\mathcal{A}$ does not know/have K_{rs} (which is known only to ECU_r and ECU_s.). Thus, $\mathcal{A}$ cannot proceed further to impersonate ECU_r in the proposed RFTRP.

To launch an impersonation attack on ECU_s, $\mathcal{A}$ should know ID_{ECU_s} and K_{rs} because these values are used in the computation of P_{rs} and Q_{rs}. In addition, $\mathcal{A}$ needs p_i (which is selected by ECU_r.) to generate EK_{rs} and AK_{rs}. However, $\mathcal{A}$ cannot compute all required values to perform malicious activities in the proposed RFTRP. If $\mathcal{A}$ sends forged parameters ($CT^{\mathcal{A}}_{rs}$, $R^{\mathcal{A}}_{rs}$, and $T^{\mathcal{A}}_{i2}$) to ECU_r, then it is directly identified through ΔT and $R_{rs} \overset{?}{=} R^{\mathcal{A}}_{rs}$. Consequently, $\mathcal{A}$ cannot impersonate ECU_s in the proposed RFTRP. □

Theorem-3: The proposed RFTRP satisfies message confidentiality.

Proof. $\mathcal{A}$ is interested to learn transferred message (M) during the proposed RFTRP. M is used in the computation of CT_{rs} and thus, $\mathcal{A}$ requires R_{rs}, p_i, CT_{rs}, and ID_{ECU_s}. $\mathcal{A}$ can get R_{rs} and CT_{rs} from $\{CT_{rs}, R_{rs}, T_{i2}\}$, but it does not know p_i and ID_{ECU_s}. And it is difficult to compute p_i without knowing K_{rs} and ID_{ECU_s}. Therefore, $\mathcal{A}$ cannot get M due to unavailability of all essential values in the proposed RFTRP. □

Theorem-4: The proposed scheme is secure against session key attacks based on the ROM.

Table 2: Security Attributes Comparison for CAN Data Transmission Schemes

Security Attributes	Woo et al. [8]	Palaniswamy et al. [18]	AutoSec
Private key storage issue	Yes	Yes	No
Storage table requirement	Yes	Yes	No
Presence of authentication attacks	Yes	Yes	No
Encryption key compromising	Yes	Yes	No
Compatibility of session key	No	Weak	Strong
Session key update leakage	Fully	Partial	No
Session key request verification	No	ECC based	SHA-512 based

Proof. We consider that $\mathcal{A}$ wants to update the session key between ECU_i and $GECU$ illegally, by acting as $GECU$. Thus, ECU_i is the challenger ($\mathfrak{C}$) in this case to confirm the genuineness of the received parameters. $\mathcal{A}$ should compute X_i [$= h(ID_{ECU_i}||T_{i1}||K_i||GK)$ $\oplus x_i$] and Y_i [$= h(GK||ID_{ECU_i}||x_i||ID_{GECU})$] to proceed for the session key updation. We assume that $\mathcal{A}$ manages GK as an internal adversary, but s/he does not know K_i (because only ECU_i and $GECU$ know K_i.). Furthermore, $\mathcal{A}$ requires x_i and K_i to compute $Z_i^{\mathcal{A}}$ to send reply message (i.e., $\{Z_i, T_{i2}\}$) for the session key update confirmation. However, $GECU$ confirms computed Z_i (at $GECU$ side) with $Z_i^{\mathcal{A}}$ (sent by $\mathcal{A}$). If any change in mutual parameters, then $Z_i \overset{?}{=} Z_i^{\mathcal{A}}$ fails in the verification procedure. Hence, $\mathcal{A}$ cannot update the session key between $GECU$ and ECU_i. □

Table 2 summarizes a comparative analysis for [8], [18], and AutoSec, showing the security strengths in different attributes. The schemes in [8] and [18] are designed with the concept of group key-based authentication, enabling the adversary to launch authentication attacks through a compromised ECU. Furthermore, the session key can be leaked fully/partially, as the encryption key of i^{th} session remains the same for all ECUs in [8] and [18]. Moreover, a Trusted Platform Module (TPM) is required in [8] and [18] to store crucial values, whereas the AutoSec does not have the requirement of a TPM. Since the AutoSec is designed with one-way hash (i.e., SHA-512), and the session key computation values are different for each ECU in every session, the AutoSec is robust in various security attributes.

6 PERFORMANCE EVALUATION

Messages are broadcast over the CAN, and an ECU is a resource-constrained component for processing/transmitting information for in-vehicle communications. Therefore, it is necessary to take computation time, communication overhead, and energy consumption into account for cost-effective automotive data exchanges. For efficient results, it is indispensable to reduce the number of different cryptographic operations and parameters while satisfying security. Hence, we compare our proposed scheme with that of recent schemes that have used CAN in IVNs.

6.1 Testbed Configuration

ECU hardware in autonomous vehicle is configured with 64-bit ARM core processors to perform computations [22], and Raspberry Pi 3B+ is also designed based on the ARM processor. Thus, we consider a testbed environment with two Raspberry Pi 3B+ devices to measure the computational efficiency of AutoSec. The configuration of Raspberry Pi 3B+ is as 1.4 GHz quad-core 64-bit ARM Cortex-A53 processor with BCM2837B0 chip, 1 GB SRAM, 2.5 Amp power, and 5 V voltage supply [23].

6.2 Results Analysis

We present performance results for AutoSec and relevant communication schemes based on different performance parameters, such as computation time, communication overhead, and energy consumption to analyze their efficiency for efficient in-vehicle communications. A detailed explanation for results analysis is as follows.

Computation Time. It is calculated based on the required number and types of cryptographic operations required during the data transmission phase for execution. The computation time is generally measured in milliseconds (ms). To calculate the execution time for the different operations, we execute the corresponding cryptographic operations (i.e., AES 128-bit encryption/ decryption (AES), EC Small-scale Multiplication (ECSM) with a 256-bit large prime number, SHA-512 [$h(\cdot)$), and KDF_{GK}] using Python libraries (i.e., pycrypto and py-ecc) in Python 3.7 on Raspberry Pi 3B+ platform. Here, KDF_{GK} is a key derivation function based on SHA-512 that generates two key values of 512 bits, achieving more security during the key generation process. Since SHA-512 is practiced for both computations ($h(\cdot)$ and KDF_{GK}), the execution time remains the same for both the functions. After 100 runs, the average execution time is 1.1410 ms for T_{AES}, 3.0520 ms for T_{ECSM}, 0.0014 ms for $T_{h(\cdot)}/KDF_{GK}$. Since the execution time of $\oplus$ and $||$ is highly negligible, it is not taken into consideration for all the schemes.

The scheme in [8] requires to execute $6T_{h(\cdot)} + 2T_{KDF}$ in ISCVP, $2T_{AES} + 2T_{h(\cdot)}$ in RFTRP, and $2T_{AES} + 3T_{h(\cdot)} + 2T_{KDF}$ in SKUP. The protocol design in [18] needs to perform $4T_{h(\cdot)}$ in ISCVP, $4T_{h(\cdot)} + 2T_{AES}$ in RFTRP, and $8T_{h(\cdot)} + 4T_{ECSM}$ in SKUP. However, the AutoSec requires only $9T_{h(\cdot)} + 2T_{KDF}$, $9T_{h(\cdot)} + 2T_{KDF}$, and $6T_{h(\cdot)} + 2T_{KDF}$ for ISCVP, RFTRP, and SKUP phases respectively. Considering the average execution time of each operation, we calculate the computation time for Woo et al. [8], Palaniswamy et al. [18], and AutoSec based on the required operations in each of the scheme. Figure 5 shows the required computation time for relevant schemes. Since AutoSec is designed with low-cost and fewer operations, the computation time in AutoSec is comparatively less than [8] and [18].

Energy Consumption. When different operations are performed for data transmission, it consumes energy to execute different operations during message/key generation and verification. Thus, it

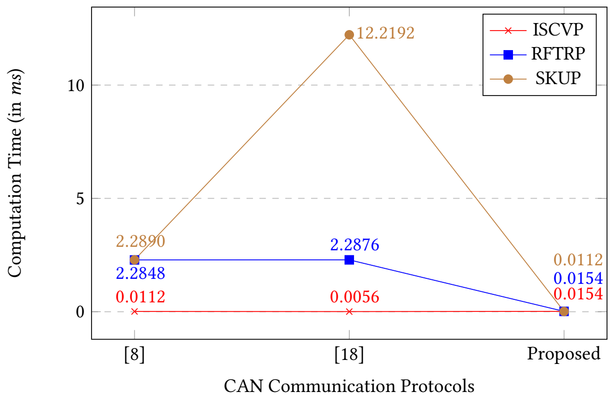


Figure 5: Computation Time Comparison for Relevant CAN Communication Schemes

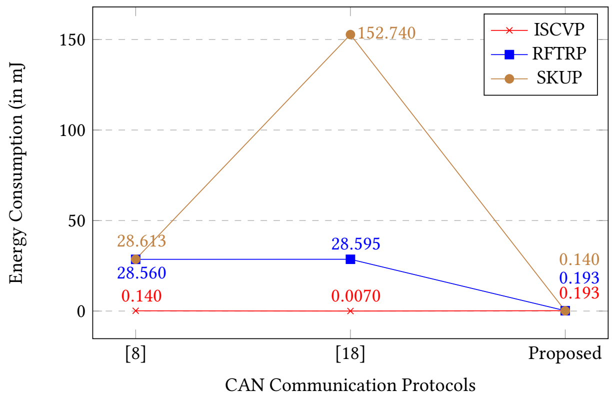


Figure 6: Comparison of the Required Energy in Relevant CAN Communication Protocols

is necessary to evaluate the energy requirement. It is computed as $EC_{CO} = T_{CT} \cdot P_{CPU}$, where EC_{CO} = energy consumption, T_{CT} = computation time, $P_{CPU} = V \cdot I$ = CPU maximum power, V = voltage power, and I = current [24]. The energy consumption is measured in millijoule (mJ). Raspberry Pi 3B+ is used as the implementation platform, the CPU maximum power is 12.5 W [23]. Normally, the communications phases of the messages (i.e., ISCVP, RFTRP, and SKUP in Section 4.2) are routinely performed for different operations, whereas the basic setup phase is initially executed once. Thus, the energy consumption for message communication has more impact for energy-efficient data exchange schemes, calculating the consumed energy for these communication protocol designs. We have compared the required energy for each of the schemes [8], [18], and AutoSec in Figure 6. AutoSec consumes less energy, as the computation time is relatively less than [8] and [18].

Communication Overhead. Both (sender and receiver) need to exchange different overhead parameters to perform initial session key computation and verification, remote frame transmission request, and session key update (see Section 4.2). The cost involved in transferring these parameters is known as the communication overhead (measured in bytes), and it is calculated based on the type

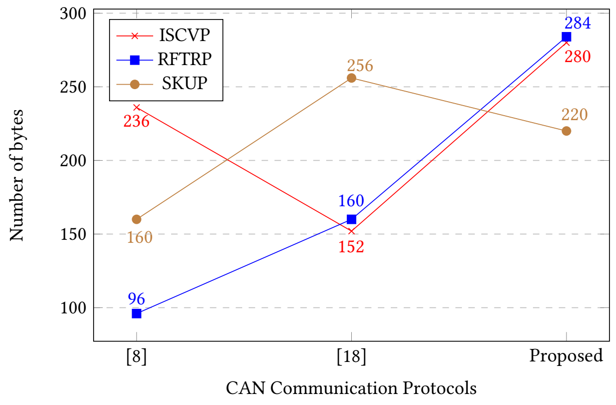


Figure 7: Communication Overhead in Relevant CAN Communication Mechanisms

and required number of parameters during each protocol. One-way hash (i.e., SHA-512 $h(\cdot)$), AES encryption (AES), EC multiplication ($ECMP$), random nonce/identity (RN/ID) and time-stamp (TS) require 64 bytes, 32 bytes, 64 bytes, 12 bytes, and 8 bytes respectively.

In AutoSec, the communication overhead is 280 bytes [$4h(\cdot)+3TS$], 284 bytes [$4h(\cdot)+1ID+2TS$], and 220 bytes [$3h(\cdot)+1ID+2TS$] in ISCVP, RFTRP, and SKUP respectively. In [8], the communication overhead is 236 bytes [$3h(\cdot)+1RN+1AES$] for ISCVP, 96 bytes [$1h(\cdot)+1AES$] for RFTRP, and 160 bytes [$2h(\cdot)+1AES$] for SKUP, whereas the scheme in [18] requires 152 bytes ($2h(\cdot)+2RN$), 160 bytes [$2h(\cdot)+1AES$], and 256 bytes [$2h(\cdot)+2ECMP$] for ISCVP, RFTRP, and SKUP respectively. Figure 7 shows a comparative study of the communication overhead for the related works.

The seed value and random nonce are broadcast over the CAN bus during ISCVP of [8], and these values are used as inputs for the key computation and verification, enabling a compromised ECU (based on broadcast values) to perform malicious activities later. Though communication overhead is minimized in [8], mutual authentication is not provided during exchange of messages in RFTRP, and its protocol design can only satisfy 128-bit security. Time-stamps are not used, and the adversary can derive seed value from the transferred parameters in SKUP that makes it easy to launch authentication attacks in [8]. In [18], important values are sent in plain text, and they do not use time-stamp that reduces the communication cost, but increases security vulnerabilities during data exchanges, and the scheme [18] can achieve 128-bit security level.

AutoSec is mainly constructed based on SHA-512 to satisfy 256-bit security, and different types of values (i.e., SHA-512 and time-stamp) are exchanged during the exchanges of overhead parameters that resists data modification, information disclosure, and message freshness over the CAN bus. Thus, the communication overhead in AutoSec is comparatively more than [8] and [18]. However, AutoSec performs better in other crucial performance measures (by taking very less computation time and energy consumption) than [8] and [18] (refer Figure 5 and Figure 6). Security and computation efficiency together are important factors for in-vehicle networks, and AutoSec outperforms collectively, having the trade-off between security and efficiency.

7 CONCLUSION

We presented AutoSec, a secure message communication scheme for CAN in-vehicle networks. AutoSec provides secure CAN message transmission using the SHA-512, bit-wise XOR and concatenation operations. These operations cater to efficient computation and verification for resource-constrained ECUs in vehicles. The efficacy of AutoSec is also analyzed to prove that it ensures user authentication, message integrity, message confidentiality, and defense against session key attacks. The proposed scheme involves low-cost cryptographic operations for executing in-vehicle message communication, thereby eliminating the requirements of hardware modifications or network improvement. Performance evaluation of AutoSec through comparative analysis with competent schemes reveals that it outperforms them in terms of energy efficiency and low communication time.

REFERENCES

[1] Humayed, A., Li, F., Lin, J., Luo, B.: CANSentry: Securing CAN-based cyberhhysical systems against denial and spoofing attacks. In: European Symposium on Research in Computer Security (ESORICS), LNCS, vol. 12308, pp. 153–173 (2020).

[2] Kim, K., Kim, J. S., Jeong, S., Park, J. H., Kim, H. K.: Cybersecurity for autonomous vehicles: Review of attacks and defense. In: Computers & Security, 102150, pp. 1–27 (2021).

[3] Kneib, M., Schell, O., Huth, C.: EASI: Edge-based sender identification on resource-constrained platforms for automotive networks. In: Network and Distributed System Security Symposium (NDSS), pp. 1–16 (2020).

[4] Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., Koscher, K., Czeskis, A., Roesner, F., Kohno, T.: Comprehensive experimental analyses of automotive attack surfaces. In: USENIX Security Symposium, vol. 4, no. 447-462, pp. 1-16, (2011).

[5] Miller, C., Valasek, C.: Remote exploitation of an unaltered passenger vehicle. In: Black Hat USA, pp. 91 (2015).

[6] Cai, Z., Wang, A., Zhang, W., Gruffke, M., Schweppe, H.: 0-days & mitigations: Roadways to exploit and secure connected BMW cars. In: Black Hat USA, pp. 39 (2019).

[7] Aliwa, E., Rana, O., Perera, C., Burnap, P.: Cyberattacks and countermeasures for in-vehicle networks. In: ACM Computing Surveys (CSUR), 54(1), pp. 1–37 (2021).

[8] Woo, S., Jo, H. J., Lee, D. H.: A practical wireless attack on the connected car and security protocol for in-vehicle CAN. In: IEEE Trans. on Intelligent Transportation Systems, 16(2), pp. 993–1006 (2014).

[9] Nowdehi, N., Lautenbach, A., Olovsson, T.: In-vehicle CAN message authentication: An evaluation based on industrial criteria. In: IEEE 86th Vehicular Technology Conference (VTC-Fall), pp. 1–7 (2017).

[10] Choi, W., Jo, H. J., Woo, S., Chun, J. Y., Park, J., Lee, D. H.: Identifying ecus using inimitable characteristics of signals in controller area networks. In: IEEE Trans. on Vehicular Technology, 67(6), pp. 4757–4770 (2018).

[11] Foster, I., Prudhomme, A., Koscher, K., Savage, S.: Fast and vulnerable: A story of telematic failures. In: 9th *USENIX* Workshop on Offensive Technologies (*WOOT*), pp. 15 (2015).

[12] Miller, C., Valasek, C.: Adventures in automotive networks and control units. In: Def Con, 21, 260-264 (2013).

[13] Woo, S., Jo, H. J., Kim, I. S., Lee, D. H.: A practical security architecture for in-vehicle CAN-FD. In: IEEE Trans. on Intelligent Transportation Systems, 17(8), pp. 2248–2261 (2016).

[14] Ying, X., Bernieri, G., Conti, M., Poovendran, R.: TACAN: Transmitter authentication through covert channels in controller area networks. In: 10th ACM/IEEE International Conference on Cyber-Physical Systems, pp. 23-34 (2019).

[15] Ying, X., Sagong, S. U., Clark, A. Bushnell, L., Poovendran, R.: Shape of the cloak: Formal analysis of clock skew-based intrusion detection system in controller area networks. In: IEEE Trans. on Information Forensics and Security, 14(9), pp. 2300–2314 (2019).

[16] Mun, H., Han, K., Lee, D. H.: Ensuring safety and security in CAN-based automotive embedded systems: A combination of design optimization and secure communication. In: IEEE Trans. on Vehicular Technology, 69(7), pp. 7078–7091 (2020).

[17] Groza, B., Popa, L., Murvay, P. S.: Highly efficient authentication for CAN by identifier reallocation with ordered CMACs. In: IEEE Trans. on Vehicular Technology, 69(6), pp. 6129–6140 (2020).

[18] Palaniswamy, B., Camtepe, S., Foo, E., Pieprzyk, J.: An efficient authentication scheme for intra-vehicular controller area network. In: IEEE Trans. on Information Forensics and Security, 25(15), pp. 3107–3122 (2020).

[19] Yu, D., Hsu, R. H., Lee, J.: EC-SVC: Secure can bus in-vehicle communications with fine-grained access control based on edge computing. In: arXiv preprint arXiv:2010.14747, pp. 1–13 (2020).

[20] Fisher, D. A., McCune, J. M., Andrews, A. D.: Trust and trusted computing platforms. In: Technical Note CMU/SEI-2011-TN-005, Carnegie Mellon University, pp. 1–26 (2011).

[21] Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: ACM Conference on Computer and Communications Security (ACM CCS), pp. 62–73 (1993).

[22] Autopilot, Processors, and Hardware. [Online]: https://teslatap.com/articles/autopilot-processors-and-hardware-mcu-hw-demystified/, Accessed on May 20 (2021).

[23] Raspberry Pi 3 Model B+. [Online]: https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus, Accessed on June 15 (2021).

[24] He, D., Chen, C., Chan, S., Bu, J.: Secure and efficient handover authentication based on bilinear pairing functions. In: IEEE Trans. on Wireless Communications, 11(1), pp. 48–53 (2011).