ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume **22**, Number 3–4, 2019, 228–243

Security enhancement in In-vehicle Controller Area Networks by Electronic Control Unit authentication

Murugesan LAKSHMANAN[1] and Senthil Kumar NATARAJAN[2]

[1]Electronics & Communication Engineering, Arasan Ganesan Polytechnic College, Sivakasi
[2]Electrical & Electronics Engineering, Mepco Schlenk Engineering College, Sivakasi
E-mails: l.murugesan1968@gmail.com, nsk_vnr@mepcoeng.ac.in

Abstract. Controller Area Network (CAN), the most popular network of Electronic Control Units (ECUs) was designed by Bosch in 1990s to prioritize reliability and safety but with absence of security. An attacker can take control of the ECUs and probably harm the safety critical operations inside the vehicle. Hence, security especially ECU authentication is most important for CAN. In this paper, an ECU authentication scheme named *Security Enhancement using Truncated Message Authentication Code* (SETMAC) for CAN is proposed. The authentication code generated using International Data Encryption Algorithm (IDEA) is truncated to four bytes to accommodate the message data into the maximum payload size of eight bytes. 8-bit message counter is incorporated among the ECUs to provide opposition to replay attacks. The proposed algorithm avoids the need of an extra frame for sending authentication messages and thus effectively reduces the bus load. The algorithm is tested with CANoe software using the CAN data set captured from a real vehicle. The simulation results show that the proposed authentication algorithm can be implemented in existing real time CAN bus networks in 500kb/s and 1Mb/s bit rates at maximum busloads of 25.05% and 12.52% respectively.

Key-words: Authentication, Security, Electronic Control Unit, Encryption, In-vehicle network, Controller Area Network, International Data Encryption Algorithm.

1. Introduction

Manufacturers of cars started replacing mechanical parts with electronic components due to strict rules with regards to emissions, efficiency improvement and enhancing engine performance. Tens of ECUs are used to the aspects of the electrical system inside a vehicle, such as the engine, brakes, windows, airbags and lights [1]. Accordingly, vehicles ended up having numerous devices connected to each other using point-to-point connections, which resulted in an increase in complexity and the number of wires used. This problem was solved by implementing in-vehicle networks namely Controller Area Network (CAN), Local Interconnect Network (LIN), Media Oriented Systems Transport (MOST) and FlexRay [2]. Due to the increase in the information transfer between the connected devices, vehicles often have multiple networks which connect different subsets of ECUs. Amongst these in-vehicle networks CAN is the mostly used robust network.

Although CAN is the most extensively used network protocol used in vehicles, regrettably, there is no built-in way to impose security. Security was not even considered during its design stages since it was thought that vehicles would remain closed systems. External attack surfaces of modern automobiles allow the possibility of a remote exploitation through various means of channels such as diagnostic tools like OBD-II, CD player, Bluetooth, cellular radio and internet connectivity. Present automobiles trust heavily on software due to the introduction of various ECUs. High chances that the code may contain bugs or vulnerabilities which when exploited; they can lead to potential risks, thus making vehicles unsafe.

Due to lack of authentication mechanisms in the CAN protocol, an attacker may also connect an external malicious device directly with the CAN bus with the intention to harm passengers or even cause dangerous accidents [1] [3]. In this paper, a novel ECU authentication scheme is suggested using International Data Encryption Algorithm (IDEA) [4] for in-vehicle CAN bus. The authentication code is truncated to four bytes and the payload is reduced to four bytes after investigation of the CAN dataset. An 8-bit counter is also introduced at the transmitter side and the receiver side to avoid replay attacks.

Section 2. presents the details of CAN bus and its security including the previous works conducted in solving the security issues of CAN. The previous works carried out for ECU authentication are discussed in Section 3.. The concept of IDEA is discussed in Section 4.. The simulation setup and the proposed authentication scheme are given in Section 5.. The performance analysis and discussion are presented in Section 6.. Finally, brief conclusion and scope for future works are given in Section 6..

2. CAN bus and Security

2.1. Electronic Control Unit

Modern cars consist of a large number of electronic devices that control almost every aspect of automobiles. These devices are self-contained automotive embedded systems called Electronic Control Units (ECUs). Virtually every component of a car, including the breaks, the engine, the throttle, lighting controls, entertainment system and so on are governed by these embedded systems. This leads to almost 100 ECUs in a modern luxury car. These ECUs monitor and control not only sensors and actuators but also technologies supporting the driver, the passengers or even certain driving situations. Those features encompass technologies like Anti-skid Braking System (ABS), Electronic Stability Control (ESP) or further advanced driver assistant systems. Historically the first ECU in a car was in charge of controlling and monitoring the engine.

2.2. In-vehicle Communication Systems

Until the beginning of the 1990s most ECUs were connected via point to point links. The growing number of ECUs made it necessary to use more sophisticated communication systems.

The varying characteristics of a state of the art vehicle demanded various attributes of such a system. For example, fast communication for multimedia services or safe real time connections for critical power train functions. This requirement led to multiple, separate vehicular networks in state of the art vehicles. These network technologies encompass protocols like CAN, Local Interconnect Network (LIN), FlexRay or Media Oriented Systems Transport (MOST).

2.3. Controller Area Network

The CAN protocol was first introduced in 1986 at an SAE conference in Detroid. In 1991 it was the first bus system that was implemented in a vehicle in mass production. The current version of the CAN protocol is the version 2.0. This version is divided into two parts - 2.0a and 2.0b. Both specifications are identical except for the additional support of 29-bit message identifiers in version 2.0b, whereas 2.0a just uses 11-bit identifiers [3] [5] [6].

Due to the larger number of different message IDs version 2.0b is mostly used for heavy-duty vehicles, while 2.0a principally is used for passenger cars. The CAN protocol is standardized since 1993 in the ISO Standard 11898. Besides these ISO standards, the CAN protocol is also standardized by other organizations like the SAE (J2284-1 to 3). In this paper the discussion is limited to the CAN 2.0a version and the ISO 11898-2 standard, as this is the most widely used standard for CAN networks. According to the ISO 11898-2 standard [7] the CAN network is based on a two-wire differential bus. These wires are named CAN-H (CAN High) and CAN-L (CAN Low). The idle state of both wires lies around 2.5 V. This voltage is also the recessive state of the CAN bus and represents a binary 1. In the dominant state both wires change their voltage by 1 V.

For the CAN-H this means a dominant state of 3.5 V and for the CAN-L a drop to 1.5 V. The dominant state of the CAN bus represents the binary 0. If two or more nodes simultaneously transmit different messages, the dominant bit always overwrites the recessive bit. Dependent on the speed, the maximum bus length differs. The speeds, bus lengths and bit times of CAN bus are listed in Table 1.

Table 1. Bit rates, maximum bus lengths and corresponding bit times of CAN bus.

Bit Rate	Bit Time	Bus Length
1 Mbit/s	1 μs	25 m
800 kbit/s	1.25 μs	50 m
500 kbit/s	2 μs	100 m
250 kbit/s	4 μs	250 m
125 kbit/s	8 μs	500 m
62.5 kbit/s	16 μs	1000 m
20 kbit/s	50 μs	2500 m
10 kbit/s	100 μs	5000 m

2.3.1. Frame Formats

The CAN protocol differentiates four types of frames - data frame, remote frame, error frame and overload frame. The data frame is the only frame that actually transmits message data. The other frames are used to request the transmission of a data frame, to allow for fault confinement, for synchronization and for flow control.

2.3.2. Data Frame

The data frame as shown in Fig. 1 is used to transmit information on the bus. As the CAN network is a broadcast medium, every node receives every message. The CAN protocol does not use any source or destination address. Instead it uses an identifier which defines the message's content (for instance the vehicle speed, engine parameters and so on). Based on this identifier each CAN node decides if the message will be further processed or if it will be ignored.

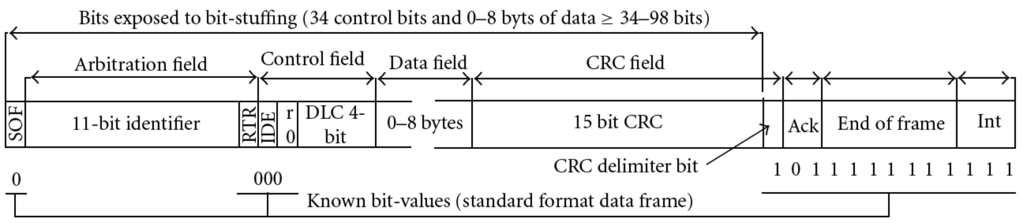

Fig. 1. CAN 2.0a Data Frame.

This approach allows for a high flexibility of the network. It does not need any information or configuration of the nodes. A new node just connects to the network and it can send and receive messages immediately. However, without any prior knowledge of the messages sent on the bus, a new node cannot determine their meaning or ascertain which messages are relevant.

The data frame in the CAN version 2.0a is depicted in Fig. 1. After a Start of Frame bit (SOF) the data frame starts with an ID of eleven bits. After the ID section, the frame has a remote transmission request (RTR) bit. This bit states, if the frame is a data frame (binary value 1), or if it is a remote frame (binary value 0). Combined with this RTR bit the ID gives the arbitration field. The subsequent control field is composed of an identifier extension (IDE), a reserved bit and the data length code (DLC). The IDE represents the current version of the frame. A binary 1 represents the version 2.0a and a binary 0 refers to the version 2.0b. The DLC then states how many data bytes the message uses. The minimum value is 0 - no data at all - and the maximum length is eight (8 bytes). After the control field the actual data is transmitted.

The data frame uses a 15 bit checksum to provide a check for the correctness of the received frame. After this checksum, the frame consists of a delimiter and an acknowledgment bit. This bit is sent out with a value of 1 (recessive state). Every node that correctly receives the message overwrites this bit with a binary 0 (dominant state).

This means the initial recessive state for this bit, will be set to the dominant state by every recipient that received the message correctly. This way the sender notices if its message was received correctly by at least one node. It is, however, not possible to determine if every node received the message correctly (overwriting a dominant bit with a recessive state is not possible). The frame concludes with an acknowledgment delimiter and 7 recessive bits representing the end of frame (EOF). After these bits an inter frame spacing of another 3 bits is used.

2.4. Security vulnerabilities in CAN

2.4.1. Broadcast risk

CAN nodes transmit to all other nodes and any message can be read. As long as an attacker has access to the CAN bus, they can read, log & analyze messages and sniffing can be used to determine the effect of each message.

2.4.2. Message authentication

CAN nodes have no method of authenticating messages. It is impossible for a node to verify the sender of a received message. So, an attacker can send any message he likes. The attacker can essentially control the entire vehicle or system assuming he has the knowledge of the effects of valid CAN messages.

2.4.3. DoS and Replay attacks

The Denial-of-Service (DoS) and replay attacks [8] are particularly simple to employ on the CAN bus due to the bus arbitration technique. A low frame ID has a higher priority and consequently to run a DoS attack. An attacker simply sends a high priority message repeatedly and they effectively block all other messages on the network. In replay attack a message is resent rapidly or replayed. There is currently no protection against the replay attack in a CAN.

3. Previous Work

A handful of symmetric key authentication and encryption schemes have been proposed in the past few years for in-vehicular networks. Two very powerful schemes developed by Verbauwhede et al., CanAuth [9] and its successor LiBrA-CAN [10], rely solely on shared-key authentication, and accordingly, adding 3rd party passive listeners is not a supported operation. CanAuth does not support single source node authentication, but rather authenticates that the message is from some trusted source. LiBrA-CAN, as a progressive authentication scheme, requires $\binom{n}{s}$ keys, s some constant bounded by n. Practically, the number of keys required for this scheme could be quite large.

TESLA, another broadcast network alternative [11], is also a strong symmetric approach that uses delayed key disclosure. The protocol has been highly criticized for its need to store unauthenticated messages until their verifying key is disclosed; the potential backup of messages makes TESLA susceptible to DOS attacks. Later versions of TESLA have tried to address this but the attempted solutions have come with a significant increase in message latency.

One of the European-funded projects, E-safety Vehicle Intrusion proTected Applications (EVITA) developed a hardware security module (HSM) for On-Board network security [12]. Schweppe *et al.* suggested a communication security architecture for vehicles using EVITA-HSM [13]. They used a truncated 32-bit MAC considering the limited data payload of CAN data frames and explained that a 32-bit MAC is secure from collision attacks for 35 weeks due to the limited properties of an in-vehicle network (CAN bus load and bandwidth). However, the security architecture of Schweppe *et al.* is very abstract. It does not provide a detailed description regarding how to generate and transmit a 32-bit MAC.

Groza and Murvay suggested a CAN data authentication protocol using a TESLA-like protocol [14]. The TESLA-like protocol finds it difficult to provide real-time processing in CAN. Since the secret key shared between a sender and the communication master cannot be changed for each session, a replay attack after eavesdropping on the transmitted CAN data frame and MAC is possible.

Murvay and Groza [15] tried to take authentication up to unique physical characteristics of the frames that were placed by each node on the bus. For this they analyzed the frames by taking measurements of the voltage, filtering the signal and examining mean square errors and convolution in order to uniquely identify each potential sender. Yoshikawa *et al.* proposed PRESENT algorithm [16] which uses a substitution-permutation network structure with 31 rounds. The use of S-boxes in PRESENT algorithm increases its complexity in terms of hardware implementation.

The authentication scheme proposed in this paper uses less complicated IDEA (avoiding the use of S-boxes and lookup tables) to generate MAC. Truncated MAC (32 bits) is proposed to minimize the payload by half and thus improves the bandwidth usage. The introduction of message counter among ECUs provides opposition to replay attacks.

4. International Data Encryption Algorithm

4.1. Symmetric-key Cryptography

In symmetric-key cryptography [17], a secret key is shared among the participants. Each of the participants can encrypt or decrypt messages with this key. To uphold the confidentiality of the communication, the key must be kept secret. It must thus be exchanged over a previously established secure channel between all participants. Symmetric-key cryptography can be implemented with relatively small resource requirements in software and, due to the typical repetitiveness of the algorithms, also be implemented more easily in hardware [18] [19] [20]. Examples for symmetric-key algorithms are Advanced Encryption Standard (AES), Data Encryption Standard (DES) and 3DES.

4.2. The IDEA Algorithm

The International Data Encryption Algorithm (IDEA) is a symmetric-key, block cipher [16]. It was published in 1991 by Lai, Massey, and Murphy. IDEA is a candidate block cipher to the New European Schemes for Signatures, Integrity and Encryption (NESSIE) Project. NESSIE is a project within the Information Societies Technology (IST) Program of the European Commission. Although IDEA did not replace DES, it was incorporated into Pretty Good Privacy (PGP). IDEA entirely avoids the use of any lookup tables or S-boxes.

IDEA encrypts a 64-bit block of plaintext to 64-bit block of cipher text. It uses a 128-bit key. The algorithm consists of eight identical rounds and a “half” round final transformation. The algebraic idea behind IDEA is the mixing of three incompatible algebraic operations on 16-bit blocks: bitwise XOR, addition modulo 2^{16}, and multiplication modulo $2^{16} + 1$. The encryption algorithm has fourteen steps. For each of the eight complete rounds, the 64-bit plaintext block is split into four 16-bit sub-blocks: X_1, X_2, X_3 and X_4. The 64-bit input block is the concatenation of the sub blocks: $X_1 \| X_2 \| X_3 \| X_4$, where $\|$ denotes concatenation. Each complete round requires six sub keys. The 128-bit key is split into eight 16-bit blocks, which become eight sub keys. The first six sub keys are used in round one, and the remaining two sub keys are used in round two. The decryption algorithm follows the same steps of the encryption algorithm.

5. Proposed Authentication Scheme

The proposed ECU authentication scheme comprises shared secret keys, truncation of Message Authentication Code (MAC), limiting the size of message data and the use of 8-bit counter. IDEA algorithm, the less complicated and easily portable in resource constrained embedded ECUs, is implemented in the proposed authentication scheme for generating the MAC. The scheme and the key update process are explained the following sections.

5.1. Shared secret keys

Shared secret keys were used (i.e., symmetric keys) due to constraints for cost and embedded deployment. Developing those keys requires a small hardware component, the Hardware Security Module (HSM) at each ECU.

5.2. Truncation of MAC

As bandwidth is a scarce resource on automotive buses, the Message Authentication Code (MAC) used for security can be truncated i.e., the use of only fractions of a calculated MAC. Already there are several environmental limitations, such as low bus speed and high bus load, the size of authentication codes can be limited further and allow a minimum length of 32 bits. Given these limitations, the expectancy time of a brute force collision attack (i.e., random guesses) for 32-bit MACs at 100 possible tries per second corresponds to over 35 weeks.

5.3. Limiting the size of message data

Referring Table 2, one can understand that out of twenty messages only five messages (25%) are having number of bits more than 32. Hence, it has been arrived to the conclusion that the message length for the proposed scheme is limited to 32 bits. This will find enough place to accommodate the truncated MAC four bytes with the maximum size of payload of eight bytes in CAN protocol.

5.4. 8-bit counter

An 8-bit counter for each message ID is used at both transmitter ECU and receiver ECUs. The value of the counter is padded with the partial 120-bit key stored in the secured memory of HSM to arrive the 128-bits key for the MAC (IDEA) algorithm. The counters are necessary to have different values for the transmission of every message IDs, so that replay attacks on the CAN bus can be avoided.

5.5. Key update process

In order to minimize the complexity of key exchange and update operations, symmetric keys are used in the proposed scheme. The details of the number of messages and their IDs would be known during the design stage of the entire network system. The partial 120-bit keys for each message ID in all ECUs are generated randomly and the keys are stored in the secured memory of HSM during manufacturing of the ECUs at production field or during the delivery of the vehicle at the dealer shop by an experienced and trusted engineer through a secured communication channel. The same process has to be followed when new ECUs are added into the existing network.

5.6. Steps involved

The steps involved in the proposed authentication scheme at transmitter ECU and receiver ECU are summarized in the following sections:

5.6.1. Transmitter ECU

The 128 bits key used for computing the MAC is arrived by combining the partial key (120 bits) stored in HSM and the 8-bit counter value generated in the ECU. The MAC is computed by IDEA using the message data and the key. The 8 bytes MAC generated by IDEA is truncated into 4 bytes. The message data and the truncated MAC are packed into a single frame. The ECE transmits the frame over the CAN bus after incrementing the 8-bit counter value. The steps in the control flow of the transmitter ECU are listed below:

a) The partial key (120 bits) for the specified message ID is retrieved from the secured memory (HSM).

b) The 8-bit counter value is used as part of the key.

c) The counter value is inserted into the partial key to get 128 bits key.

d) The MAC is computed by IDEA using the message data and the key.

e) The MAC from IDEA (8 bytes) is truncated into 4 bytes by ignoring the most significant 4 bytes.

f) The message data and the truncated MAC are packed into single frame. The total length of the frame data is length of the message data plus 4 (length of truncated MAC).

   DLC = Length of message data (Max. 4) + Length of truncated MAC (4)

g) The 8-bit counter value is incremented.

h) The frame is transmitted over the CAN bus.

The above steps at the transmitter ECU are diagrammatically shown in Fig. 2.

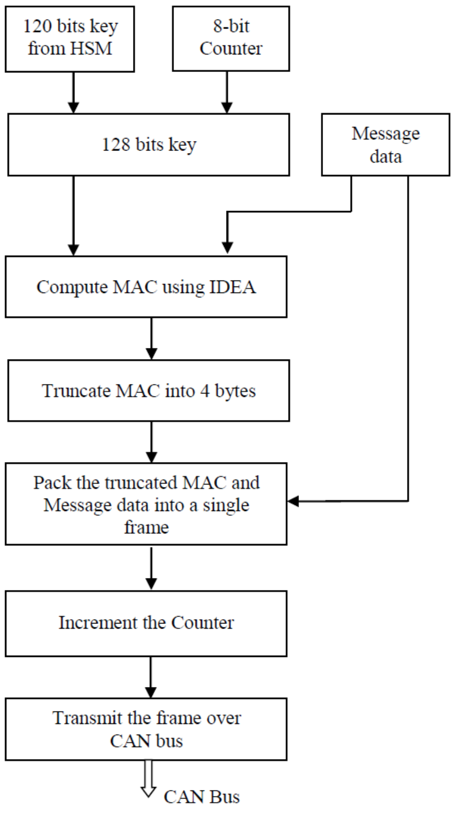
120 bits key from HSM
8-bit Counter
128 bits key
Message data
Compute MAC using IDEA
Truncate MAC into 4 bytes
Pack the truncated MAC and Message data into a single frame
Increment the Counter
Transmit the frame over CAN bus
CAN Bus

Fig. 2. Steps at the transmitter ECU

5.6.2. Receiver ECU

The frame received by the intended receiver of the specified the message ID is unpacked into message data and MAC. The length of the message data is arrived from the DLC of the message and the length of the MAC is 4 bytes. The 120-bits partial key retrieved from the HSM for the specified ID is inserted into the 8-bits counter value generated in the ECU to get the 128-bits key. The MAC is computed by IDEA using the received message data and the key. The MAC from IDEA (8 bytes) is truncated into 4 bytes by ignoring the most significant 4 bytes. The receiver ECU compares the truncated MAC and the received MAC to identify the malicious activity. Finally, the receiver ECU increments the 8-bit counter value and waits for the next frame. The steps in the control flow of the receiver ECU are listed below:

a) The intended receiver of the particular message ID receives the frame from the CAN bus.

b) The frame is unpacked into message data and truncated MAC.

   Length of message data = DLC – Length of truncated MAC (4)

c) The partial key (120 bits) for the particular message ID is retrieved from the secured memory (HSM).

d) The 8-bit counter value maintained in the ECU is used as part of the key.

e) The counter value is inserted into the partial key to get 128 bits key.

f) The MAC is computed by IDEA using the received message data and the key.

g) The MAC from IDEA (8 bytes) is truncated into 4 bytes by ignoring the most significant 4 bytes.

h) The truncated MAC and the received MAC are compared.

i) If match is occurred, it is recognized that the message is from an authenticated ECU and necessary action will be taken based on the message data.

j) If match is not occurred, it is presumed that the message is from a malicious ECU and the received message data is ignored and an alert is issued to the driver.

k) The 8-bit counter value is incremented.

The above steps at the receiver ECU are diagrammatically shown in Fig. 3.

6. Performance analysis and discussion

A simulation setup using Vector CANoe software [21], shown in Fig. 4, was developed to verify the proposed authentication scheme. In this simulation setup four ECUs are connected in one CAN channel. All the simulation experiments were conducted in four different bit rates, viz., 83 kb/s, 125 kb/s, 500 kb/s and 1 Mb/s. The code for various functions were implemented in CAPL, a ‘C’ like high level language available in CANoe simulation environment.

The real time CAN bus data captured from a USB-to-CAN device connected to the OBD-II port of a real vehicle were used. The data was collected during morning and evening drives for approximately 20 minutes long including a mix of driving in traffic, on side streets, and on highways. The message IDs, their period and number bits of the collected CAN bus traffic are listed in Table 2.

Table 2. CAN dataset

ID	Period (s)	Bits	ID	Period (s)	Bits
002	0.01	30	**154**	1.0	5
0D0	0.02	63	**156**	0.02	9
0D1	0.02	18	**282**	0.05	32
0D2	0.02	16	**360**	0.05	45
0D3	0.02	27	**361**	0.05	31
0D4	0.02	48	**370**	0.05	27
140	0.01	53	**372**	0.1	6
141	0.01	46	**374**	1.0	6
144	0.02	24	**376**	0.05	0
152	0.02	19	**660**	0.5	30

The following two experiments were carried out for analyzing the performance of the proposed authentication scheme:

A. The transmission latency of various data lengths (1 to 8 bits) at different bit rates like 83k, 125k 500k and 1Mbp/s are measured. The measurement data are shown in Table 3.

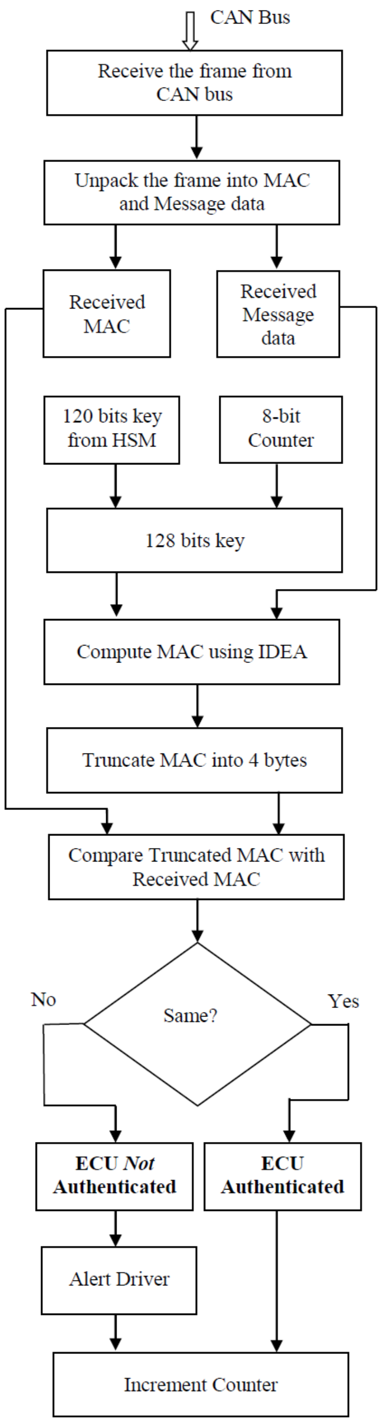

Fig. 3. Steps at the receiver ECU.

B. All the 20 messages (Ref. Table 2) are transmitted at the specified periods under the following four situations:

S1.: Original data set is transmitted

S2.: Message length is limited to four bytes and the higher length messages are transmitted in two frames

S3.: Message length is limited to four bytes and four bytes MAC are transmitted

S4.: Message length is limited to four bytes and four bytes MAC are transmitted and the higher length messages are transmitted in two frames including MAC

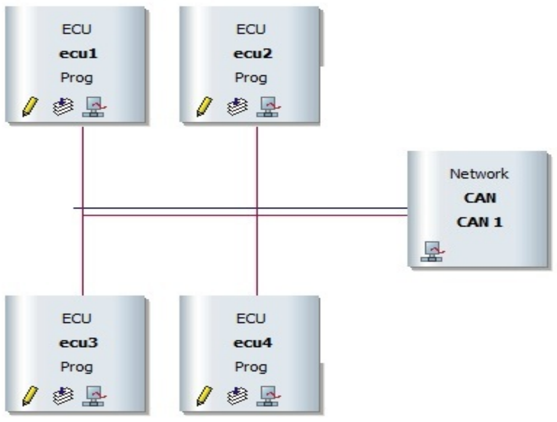

Fig. 4. Simulation setup in CANoe software.

Table 3. Transmission latency

Data Length	Transmission Latency (μs)			
(bits)	83kb/s	125kb/s	500kb/s	1000kb/s
1	684	456	114	59
2	792	528	132	66
3	888	592	148	75
4	984	656	164	84
5	1092	728	182	93
6	1176	784	196	103
7	1308	872	218	111
8	1404	936	234	122

The readings taken for the bit rates of 83k, 125k, 500k and 1Mbp/s and the schedulablility of the messages are shown in Table 4. The bus statistics like bus load, bursts, burst time, frames per burst, standard data (frames/second) and total standard data and measured for different bit rates under the above said four situations and tabulated in Table 5 and Table 6. The findings from the simulation results are discussed below:

At situation S1. - Original data set is transmitted without message length limitation and no authentication scheme:

Under this situation all the messages are scheduled at all the tested bit rates of 83k, 125k, 500k and 1000k (Ref. Table 4). Even though all the messages are scheduled at 83k the bus load is 85.69% (Ref. Table 5).

Table 4. Number of messages scheduled in various situations

Bit rate [bps]	83k				125k				500k				1000k			
Situation	S1.	S2.	S3.	S4.	S1.	S2.	S3.	S4.	S1.	S2.	S3.	S4.	S1.	S2.	S3.	S4.
Scheduled	20	16	15	8	20	20	20	20	20	20	20	20	20	20	20	20
Not scheduled	0	4	5	12	0	0	0	0	0	0	0	0	0	0	0	0

Table 5. Bus statistics for bit rates 83k and 125k

Bit rate [bps]	83k				125 k			
Situation	S1.	S2.	S3.	S4.	S1.	S2.	S3.	S4.
Bus load% [max]	85.69	100.00	100.00	100.00	57.13	69.84	72.40	100.0
Bursts [total]	13544	3	3	2	25502	25443	30144	8
Std. Data [fr/s]	814	1074	744	739	814	1134	814	1131
Std. Data [total]	275588	303229	211371	194636	230641	320558	275680	319208

Table 6. Bus statistics for bit rates 500k and 1000k

Bit rate [bps]	500k				1000k			
Situation	S1.	S2.	S3.	S4.	S1.	S2.	S3.	S4.
Bus load% [max]	14.18	17.46	18.10	25.05	7.14	8.73	9.05	12.52
Bursts [total]	28225	28258	28170	28236	28247	28192	28159	28148
Std. Data [fr/s]	814	1134	814	1134	814	1134	814	1134
Std. Data [total]	229746	320440	229302	320193	229922	319692	229204	319192

At situation S2. - Message length is limited to four bytes and the higher length messages are transmitted in two frames without implementing authentication scheme:

At 80k bit rate, four messages with low priorities (ID: 372, 374, 376 and 660) are not scheduled at all and the bus load reaches 100%. At all other bit rates all the messages are scheduled but the bus load reaches 69.84% in 125kbps.

At situation S3. - Message length is limited to four bytes and four bytes MAC are transmitted:

Only 80% of the messages are scheduled at 80k and the bus load is more 72.40% at 125k for scheduling all the messages. At 500k and 1000k bit rates, all the messages are scheduled and the bus load is reasonable 18.10% and 9.05% respectively.

At situation S4. - Message length is limited to four bytes and four bytes MAC are transmitted and the higher length messages are transmitted in two frames including MAC:

Only 40% of the messages are scheduled at 80k bit rate. Although all the messages are scheduled at 125k, the bus load reaches 100%. At 500k bit rate, all the messages are scheduled

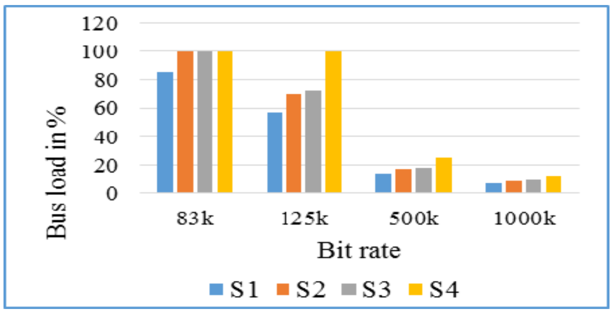

Fig. 5. Bus load at different situations

and the bus load is 25.05%. At 1000k bit rate, all the messages are scheduled and the bus load is maintained at 12.52%. The bus loads at four situations are shown in Fig. 5.

The simulation experiments and results prove that the proposed ECU authentication scheme using IDEA can be easily implemented in existing real time in-vehicle CAN bus networks at 500kb/s and 1Mb/s bit rates in both the situations S3. and S4. i.e., under limiting the message length to four bytes and also when the higher length messages are transmitted in two frames including MAC.

The message authentication as proposed in this paper has the following advantages:

1. The authentication is compatible with the existing CAN protocol. It does not require any change to the CAN protocol, which has been widely adopted and is difficult to modify.
2. It has low deployment overhead and low maintenance overhead.
3. Only a minor hardware modification is required in the existing ECUs for implementing secure key storage using HSM.
4. The authentication scheme has introduced only a small and tolerable delay so that the normal operation of the vehicle is not affected. That is, it is fast and real-time, with minimum latency.
5. The overall cost to implement the entire authentication scheme is low in terms of hardware and software as well as labour.

7. Conclusion and Future scope

Amongst the various in-vehicle networks, CAN is the most widely used robust network protocol being used in modern vehicles. The success of CAN in vehicular networks makes the possibility of extending the application of CAN in robotics and avionics also. The recent technological advancements in transportation such as driver less cars warrants high level security. Absence of security mechanism and authentication in CAN paves path to the attackers for invading into the vehicular networks.

In this research, a security enhancement is suggested for in-vehicle Controller Area Networks by ECU authentication using International Data Encryption Algorithm (IDEA). The proposed algorithm was simulated in Vector CANoe software and the simulation results show that the proposed algorithm can be easily implemented in existing real time in-vehicle CAN bus networks with 500kb/s and 1Mb/s bit rates at a maximum of 25.05% and 12.52% bus loads respectively.

Future researches may be carried out for implementing the proposed ECU authentication scheme in real time embedded system hardware and in real vehicles. Since IDEA uses no lookup tables and S-boxes and only the truncated authentication code is used in the proposed scheme it easier to implement in hardware.

References

[1] Kyusuk Han, Andre Weimerskirch, and Kang G. Shin, *Automotive Cybersecurity for In-vehicle Communication*, IQT Quarterly, Vol. 6, No. 1, 2014, pp. 22–25.

[2] Shane Tuohy, Martin Glavin, Ciarán Hughes, Edward Jones, Mohan Trivedi, and Liam Kilmartin, *Intra-Vehicle Networks: A Review*, IEEE Transactions on Intelligent Transportation Systems, Volume: 16, Issue: 2, April 2015, pp. 534–545.

[3] Jiajia Liu et al., *In-vehicle network attacks and countermeasures: Challenges and Future directions*, IEEE Network, vol. 31, no. 5, 2017.

[4] M.P. Leong et al., *A Bit-Serial Implementation of the International Data Encryption Algorithm IDEA*, Symposium on Field-Programmable Custom Computing Machines, IEEE, 2000.

[5] *CAN specification version* 2.0, Robert Bosch GmbH, 1991.

[6] Hiroshi Udea et al., *Security Authentication system for In-vehicle network*, SEI Technical review, Number 81, 2015, pp. 5–9.

[7] International Standard ISO 11898-2 Road Vehicles – *Controller Area Network (CAN)* accessed at https://www.sis.se/api/document/preview/921358/

[8] A. Boudguiga, W. Klaudel, A. Boulanger, and P. Chiron, *A Simple Intrusion Detection Method for Controller Area Network*, Communications (ICC), 2016 IEEE International Conference, 2016, pp. 1–7.

[9] Anthony Van Herrewege, Dave Singelee, and Ingrid Verbauwhede, *CANAuth - A Simple, Backward Compatible Broadcast Authentication Protocol for CAN bus*, ESCAR, 2011.

[10] Bogdan Groza et al., *LiBrA-CAN: Lightweight Broadcast Authentication for Controller Area Networks*, ACM Transactions on Embedded Computing Systems, vol. 16, no. 3, 2017, pp. 90.1–90.28.

[11] A. Perrig et al., *Efficient authentication and signing of multicast streams over lossy channels*, Symposium on Security and Privacy, IEEE, 2000.

[12] *The EVITA project*, 2008 accessed at http://evita-project.org

[13] Hendrik Schweppe et al., *Car2X Communication: Securing the Last Meter - A cost-effective approach for ensuring trust in Car2X applications using in-vehicle symmetric cryptography*, IEEE Vehicular Technology Conference, 2011.

[14] Bogdan Groza, and Stefan Murvay, *Efficient Protocols for Secure Broadcast in Controller Area Networks*, IEEE Transactions on Industrial Informatics, vol. 9, no. 4, 2013, pp. 2034–2042.

[15] P.S. Murvay, and B. Groza, *Source identification using signal characteristics in controller area networks*, IEEE Signal Processing Letters, vol. 21, no. 4, 2014, pp. 395–399.

[16] Masaya Yoshikawa et al., *Secure in-vehicle Systems using Authentication*, International Journal of Networked and Distributed Computing, vol. 3, no. 3, 2015, pp. 159–166.

[17] Richard Soja, *Automotive Security: From Standards to Implementation*, Automotive Security - White paper, Freescale accessed at https://www.nxp.com/docs/en/white-paper /AUTOSECURITYWP.pdf

[18] Takeshi Sugashima, Dennis Kengo Oka, and Camille Vuillaume, *Approaches for Secure and Efficient In-Vehicle Key Management*, DENSO Technical Review, vol.21, 2016, pp. 140–149.

[19] Bogdan Groza, and Pal-Stefan Murvay, *Security Solutions for the Controller Area Network - Bringing Authentication to In-Vehicle Networks*, IEEE Vehicular Technology Magazine, 2018, pp. 2–9.

[20] Soumya Sukumaran, and Neenu George, *Security Based Protocol Design for In-Vehicle Controller Area Network*, Indian Journal of Engineering, vol. 13, no. 32, 2016, pp. 263–269.

[21] *Vector CANoe Software Product Information* accessed at https://assets.vector.com/cms/content/products/canoe/canoe/docs/ProductInformations/CANoe_ProductInformation_EN.pdf