# Replay Attack on Lightweight CAN Authentication Protocol

Pakinam Noureldeen
Software Engineering dept., Ahram Canadian University,
Nile University
Cairo, Egypt
Pakinam.noureldeen@acu.edu.eg

Marianne A. Azer
Ministry of comm. and Information technology, National
Telecommunication Institute, Nile University
Cairo, Egypt
mazer@nu.edu.eg

Ahmed Refaat
Software Engineering dept.,
Nile University
Cairo. Egypt
ahmed.refaat@nu.edu.eg

Mahmoud Alam
Software Engineering Program Director
Nile University
Cairo, Egypt
Mahmoud.allam@nu.edu.eg

***Abstract*—Day after day, users' expectations of tomorrow vehicles' features are increasing. Although the industry's prime goal is users' satisfaction, many unsolved problems are still present. Amongst the major challenges are the huge interconnections and dependability within the Electronic Control Unit (ECU) used inside vehicles. Five years ago, the number of ECUs within a vehicle was about 70 ECUs. This number has doubled nowadays, and it is expected to double again in the near future. This adds challenges to both of network management and network security. The purpose of this paper is to enhance the CAN immunity against attacks. In this paper, we focus on the lightweight CAN authentication protocol, investigate the protocol immunity against Denial of service attack, and propose a solution for this attack. As a result, we achieve security protocol that is suitable for all security aspects.**

***Keywords—Authentication; CAN network; Embedded Security; Automotive Security; In-Vehicles network***

## I. Introduction

Automotive and electronics industry are both evolving at a very high pace. More features and capabilities have been introduced for vehicles; autonomous vehicles have become real after being a dream for many decades [1]. With the current technology revolution, the concept of traditional vehicles does not satisfy many users. This has forced the industry to enrich current low- cost vehicles with luxury features as shown in *Fig*. 1[2]. However, the more luxury features added, the more safety and security hazards emerge. Safety hazards threat the ECU correct behavior and functionality. In addition, security hazards threat the message payload privacy and confidentiality besides source authentication. For instance, when vehicles and mobiles interface through Wi-Fi, Bluetooth or any other communication technology, vehicles are exposed to many of the vehicle viruses [2].

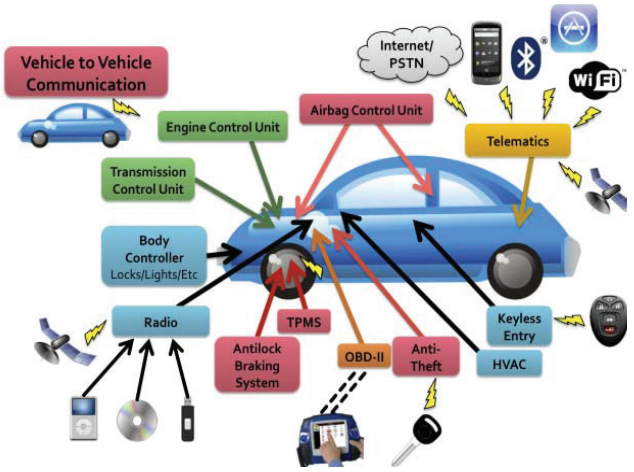


Fig. 1. Attacking surfaces [2].

Many attacks were introduced for each type of vehicles' network. Controller Area Network (CAN) [3] has gained the biggest share of the proposed attacks due to its wide usage [2] [4] [5] [6] [7]. Since its first emergence in 1986, CAN network has been the main focus from the automotive makers and suppliers. A message- based bus was designed to allow the different ECUs within the vehicle to interface with high fault-tolerant. In this type of communication bus, higher priority messages utilize the bus before lower priority ones. FlexRay [8], which is the future vehicle network, is vulnerable to some attacks too [9]. As a result, the automotive security field needs more focus from all stakeholders (i.e., automakers and automotive suppliers) [10]. Security specifications are still not sufficiently provided for any of the automotive networks (e.g., CAN, LIN [11], FlexRay, MOST [12], TTEthernet [13]). The automotive industry is considered late if compared to the fast growth of the facilities added to vehicles [14]. Proposed enhancements introduced by the security community [10][15] provide a chance to avoid various impacts ranging from simple intrusion to in-vehicle network or automaker financial losses [16] up to the injury, or

978-1-5386-1191-3/17/$31.00 ©2017 IEEE

even human death. Developing new authentication protocols is a mitigation trend. Each new protocol has been proposed to solve one or more problem. The objective of this paper is to examine the Lightweight Authentication Protocol (LCAP) [17] against the replay attack. The severity of the replay attack is tested for LCAP and presented here. Finally, a solution framework is proposed which enhances LCAP against a replay attack. It contributes to getting better security protocols that fulfill more security aspects required in the automotive domain.

The paper is organized as follows; in section 2, we present the work that has been in the literature to secure the CAN. We introduce the LCAP protocol in section 3. The replay attack simulation and our proposed solution are presented in section 4. Finally, conclusions and future work are presented in section 5.

## II. RELATED WORK

Any automotive security CAN protocol faces three main challenges which are the backward compatibility, replay attack and the message authentication [18]. Backward compatibility is an important requirement that has to be taken into consideration in each security protocol. The proposed solution in [18] is to use out-of- bound security data transferred over CAN message. This solution is based on the proposed CAN+ [19]. The second and the third challenges can be overcome using key management and authentication techniques. The Key distribution over CAN was discussed in [20]. The proposed solution in [18] utilizes a Hardware Security Module (HSM) that is attached to the ECU. One of its main responsibilities is securing the network keys. Each node owns two main keys. The first one has used for the authentication purposes. Whereas the second is used for encrypting the key exchanges messages, which are transferred between this node and its partners. All of the exchanged keys apply the driving cycle time limit criteria, where each generated key is expired after each driving cycle. The main drawback of this protocol is the proposed hardware module. This imposes a hard backward compatibility problem besides the cost increase problem.

In [21], a Message Authentication Code (MAC) was added to each message. MAC calculation was based on HMAC function. A counter was inserted in the MAC calculation in order to preserve the replay attack resistance requirement. Multicast authentication that utilizes symmetric cryptography was discussed in [22]. This contribution proposed to add multiple MACs in each CAN message. Each MAC is calculated using a separate key which belongs to a certain receiver. After message delivery, each receiver extracts his MAC and verifies it. The main drawback of this protocol is the limited size of CAN payload (8-bytes), hence half of the payload is reserved for carrying the MACs. Moreover, any increase in the number of receivers enforces to send each message multiple times but with different MACs. Also, the processing power and memory consumption linearly increase by increasing the number of receivers within the network. This leads to critical problems in the protocol scalability feature. CAN confidentiality was discussed in [23]. The author selected the AES encryption algorithm to preserve the CAN message confidentiality. The side-channel attack on the algorithm is discussed in the same contribution. The main drawback is the algorithm large output. The resulted cipher is 16 bytes; hence, it needs 2 CAN messages to carry them. In other words, each CAN message is encrypted and transferred in 2 CAN messages. In other words, the bus load has doubled besides, affecting the scalability feature for this protocol.

The Timed Efficient Stream Loss-tolerant Authentication (TESLA) was proposed in [24]. TESLA protocol addresses a powerful adversary which is capable of controlling the network with high authenticity. Its main idea is attaching a Message Authentication Code (MAC) to the message; this MAC is disclosed later in order to be able to interpret the messages. This version of the protocol requires a large amount of memory till the MAC key is disclosed. Therefore, it is not suitable for resource constrained environments. Moreover, the data freshness is a critical requirement in automotive domain and this scheme does not meet this requirement. Moreover, it is highly exposed to Denial of Service (DoS) attack. Some workaround was proposed in another version of the protocol in [25]. Then, the memory- based DoS attack solution was proposed in [26]. It is based on the immediate authentication of the received packet, instead of storing it at the receiver's side till the key is disclosed. In addition, a time synchronization technique was proposed to alleviate the DoS attack. Although this version presented an acceptable solution for the network's authentication, it does not fit the embedded system environment. Therefore, TESLA in [27][28]was more concerned with the limited resource environment. The newer version provides low communication overheads, ensures data freshness and replay protection. Besides, it is based on a master- slave architecture. However, TESLA does not propose a solution for information leakage, compromised sensors, and packet confidentiality. Therefore, it is not suitable for the automotive domain. The latest version (TESLA++) was introduced in [29], it is oriented towards vehicular systems. This version is more suitable for 'Vehicle to Vehicle' (V2V) and 'Vehicle to Infrastructure' (V2I) communication; however, it is designed for authenticity within the vehicle's internal network.

VeCure authentication protocol was introduced in [30]. Each CAN message has to be associated with another message, the ordinary CAN messages are followed by their authentication message. This protocol introduces a large overhead to the already loaded network. Therefore, the authors proposed to subdivide the network into zones depending on the criticality.

## III. LIGHTWEIGHT AUTHENTICATION PROTOCOL

Lightweight CAN Authentication Protocol (LCAP) aims to provide different security measurements to the CAN network from the security perspective. The main purpose of the LCAP is to keep messages authenticity and integrity for all messages traveling over the CAN bus [17]. The protocol phases start with the initialization phase which involves node and channel initialization, then the operation phase with two modes of operations; standard and extended modes. Finally, Robustness phase in which the LCAP offers two types of synchronizations; soft sync and hard sync in order to reestablish lost communication. The LCAP phases and operation are summarized in Table 1**.**

## IV. Proposed simulation setup

The LCAP authors mentioned the vulnerability of the replay attack in [17]. The replay attack scenario replays the channel request sent by a certain receiver. As a result, the sender responds with five control messages. The authors claim that this protocol is immune to this attack. His justification is that the receiver neglects the resulted control messages sent on this fake request. The receiver does not accept these new control messages because they do not belong to one of its valid channel requests. However, a critical problem has risen from this simple attack and it is described below in details. A malicious node can record the traveling messages on the bus. With a simple trial and error, it can detect the channel's request message. The next step is to replay this message with a reasonable periodicity. The reasonable periodicity is the amount of time required by the victim sender to send five consecutive control messages [17]. The continuity of this scenario simulates a Denial of Service (DoS) attack that is unfortunately caused by a legal behavior. The vulnerability to the DoS attack comes from the characteristics of the CAN.

The DoS attack depends on sending a valid control message with a small periodicity. The CAN policy is to send the highest priority messages first. The message priority is identified by the message ID. Moreover, the control messages have a higher priority than normal data messages. As a result, the sender gets stuck in sending these controlling messages. Therefore, all data messages fail to travel on the bus. The proposed replay attack [31][32] scenario may be as simple as discussed earlier; however, the result is of high severity. This simple attack may lead to a DoS attack that may cause high financial losses or even human death. Moreover, it is hardly detectable during investigations. The severity is linearly proportional to the number of receivers interested in the victim sender messages. It is maximum in case that the attacker is a compromised node on the network. A compromised node has a greater knowledge about the network. Therefore, its ability to attack is higher. Moreover, the ability to detect it by the investigation is harder.

### *A. Proposed replays attack simulation setup*

The ECU notation is (E) and the sub letter refers to its first letter in its full name (i.e. Engine ECU is $E_E$). The CAN message is notated as M. The message is sub-titled with the first letter of the source and first letter from the destination (i.e. $M_{E-D}$ is a message from Engine to Display). The control message is notated as MC and also sub-titled with the first letter of the source and destination (i.e. $MC_{E-D}$ is a control message from Engine to Display). PK is a notation of a pre-shared secret between each two ECU (i.e. $PK_{E-D}$ is the pre-shared key between the Engine and Display). Finally, SK is the notation for the session key of a certain ECU (i.e. $SK_E$ is the Engine session key for the current driving cycle).

Simulation material: The simulation is built on vector canoe simulation, which is amongst the most famous automotive network simulators worldwide. For simplicity, a system demo provided with the canoe software is used. This system consists of three ECUs (i.e. Engine {$E_E$}, Display {$E_D$}, and Light {$E_L$}). This is depicted in *Fig. 2***.**

The system uses two messages; the first travels from $E_E$ to $E_D$, and the second travel from $E_D$ to $E_L$. LCAP is deployed on the message $M_{E-D}$ which travels from $E_E$ to $E_D$. Therefore, a control channel Request message comes from $E_D$ to $E_E$, and then $E_E$ responds with five consecutive control messages.

TABLE I. LCAP phases and operations

| Phases | | Function | Encryption scheme | Key | Outcomes |
|---|---|---|---|---|---|
| **Initialization** | ***Node initialization*** | Sender gets ready (i.e., key generation) for responding to receivers' requests. | Symmetric encryption scheme | Pre- shared key | □(Session/HMAC) keys.<br>□Initial channel magic chain generated and delivered to receivers. |
| | ***Channel creation/ key distribution*** | Distribute generated keys to receivers. | | Session key | |
| | ***Message setup*** | The sender sends initial magic value per message to receivers | | Session key | The receiver gets initial magic value for each message it receives. |
| **Normal operation/ modes** | ***Standard mode operation*** | The message magic value is sent within the frame payload (6 bytes only for data) | | Session key / HMAC Key | Frame payload = message magic value (2bytes) + data (6bytes) |
| | ***Extended mode operation*** | Message magic value is set in the identifier extended field | | | Frame payload length is 8 bytes and magic value is stored in the identifier's extended field |
| **Robustness** | ***Soft sync*** | Sync certain receivers with message magic value chain | | Session key | Receiver re-synchronizes with message magic value |
| | ***Hard sync*** | A receiver loses either the session key or HMAC key, then it can get them again and re-sync with message magic value | | Pre- shared key | A receiver that loses sync can re-sync again as if it has just attached to network |

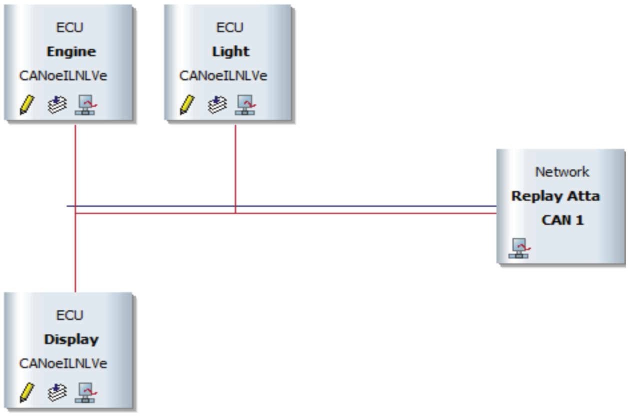


Fig. 2. System network ECUs.

We assume that EL is the outsider ECU newly attached to the network. In each driving cycle, EE and ED exchange LCAP six consecutive control messages that regulate the data transfer operation during this driving cycle. These control messages are encrypted using PKE-D and SKE. From the system point of view, control messages are assigned higher priority IDs than data ones. In addition, the request control messages are assigned a higher priority than response control messages. EL records the traveling messages on the bus. Later on, EL replays the recorded messages from higher priority to a lower priority. In its trial and error, the sequence stops after sending a message that is followed by five consecutive messages with the same ID every time. *Fig.3* shows the attack scenario flowchart. All of these messages are encrypted using PKE-D or SKE. Therefore, EL cannot interpret any of these messages. It only searches for a message that satisfies the previously mentioned criteria.

### B. *Propsed replay attack impact*

The system's behavior after applying this attack is equivalent to a system under DoS attack (as depicted in *Fig. 4*), it is as follows:

- Only control messages travel on the bus.
- $E_D$ does not report any data message anymore; it is kept busy with responding the coming channel requests.
- LCAP control messages are the only traveling messages over the bus.

No data messages travel anymore, as depicted in *Fig. 5* and *Fig. 4*, where *Fig. 4* shows the system trace is not available during the attack and no messages are traveling and *Fig. 5* shows that no signal availability anymore.

Therefore, there is no synchronization preserved or even control on any other ECU within the network. Moreover, the system operational signals availability is not preserved as depicted by the dashed lines in *Fig. 5*.

*Fig. 6* and *Fig. 7* show the bus load before and after the attack, the bus load before applying the attack is 4% by max, on the other side after applying the attack becomes 100%. The bus load statistics are summarized in *Fig. 8*. These results show the attack severity within this small network; therefore, increasing the network complexity will increase the attack severity.

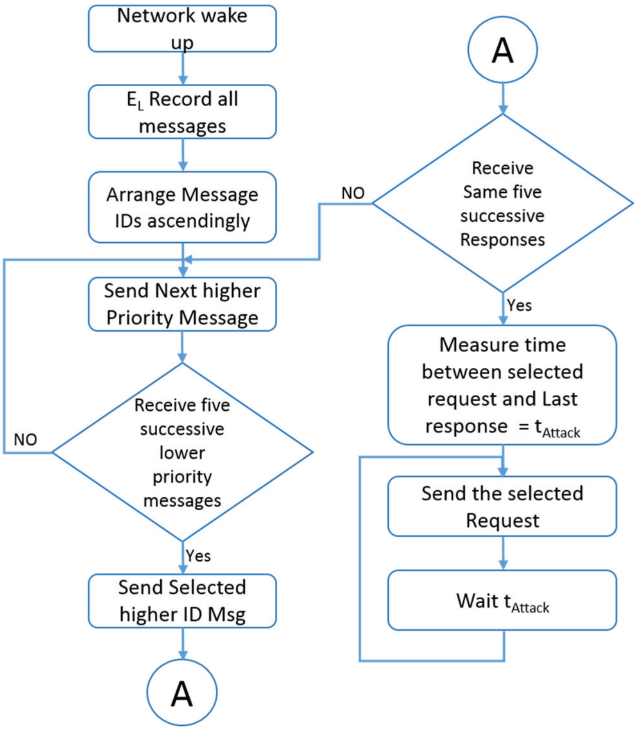
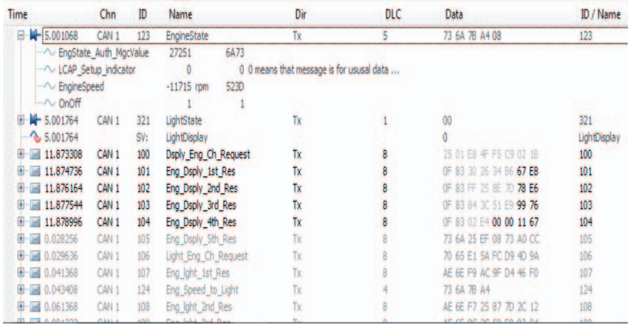

Network wake up
$E_L$ Record all messages
Arrange Message IDs ascendingly
Send Next higher Priority Message
Receive five successive lower priority messages
NO
Yes
Send Selected higher ID Msg
A
A
Receive Same five successive Responses
NO
Yes
Measure time between selected request and Last response = $t_{Attack}$
Send the selected Request
Wait $t_{Attack}$

Fig. 3. Replay attack scenario flowchart



```
Time         Chn    ID   Name                      Dir  DLC  Data                       ID / Name
⊟ 5.001068   CAN 1  123  EngineState               Tx   5    73 6A 7B A4 08             123
   EngState_Auth_MgcValue   27251       6A73
   LCAP_Setup_indicator     0           0  0 means that message is for ususal data ...
   EngineSpeed              -11715 rpm  523D
   OnOff                    1           1
⊞ 5.001764   CAN 1  321  LightState                Tx   1    00                         321
  5.001764          SV:  LightDisplay                        0                          LightDisplay
⊞ 11.873308  CAN 1  100  Dsply_Eng_Ch_Request      Tx   8    25 01 E8 4F F5 C9 02 1B    100
⊞ 11.874736  CAN 1  101  Eng_Dsply_1st_Res         Tx   8    0F 83 30 26 34 B6 67 E8    101
⊞ 11.876164  CAN 1  102  Eng_Dsply_2nd_Res         Tx   8    0F 83 FF 25 8E 7D 78 E6    102
⊞ 11.877544  CAN 1  103  Eng_Dsply_3rd_Res         Tx   8    0F 83 84 3C 51 E9 99 76    103
⊞ 11.878996  CAN 1  104  Eng_Dsply_4th_Res         Tx   8    0F 83 02 E4 00 00 11 67    104
⊞ 0.028256   CAN 1  105  Eng_Dsply_5th_Res         Tx   8    73 6A 25 EF 08 73 A0 CC    105
⊞ 0.029636   CAN 1  106  Light_Eng_Ch_Request      Tx   8    70 65 E1 5A FC D9 4D 9A    106
⊞ 0.041368   CAN 1  107  Eng_lght_1st_Res          Tx   8    AE 6E F9 AC 9F D4 46 F0    107
⊞ 0.043408   CAN 1  124  Eng_Speed_to_Light        Tx   4    73 6A 7B A4                124
⊞ 0.061368   CAN 1  108  Eng_lght_2nd_Res          Tx   8    AE 6E F7 25 87 7D 2C 12    108
```

Fig. 4. System trace during attack

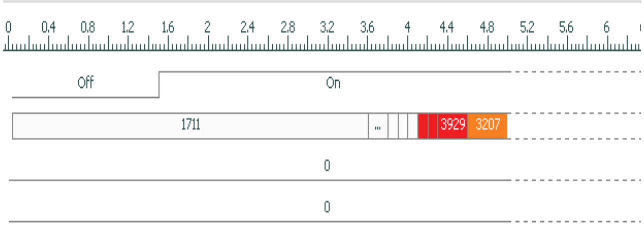
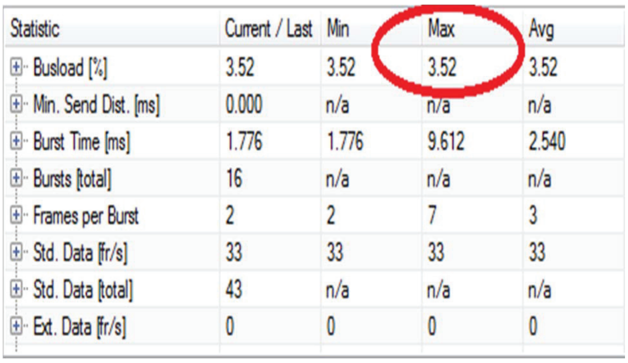
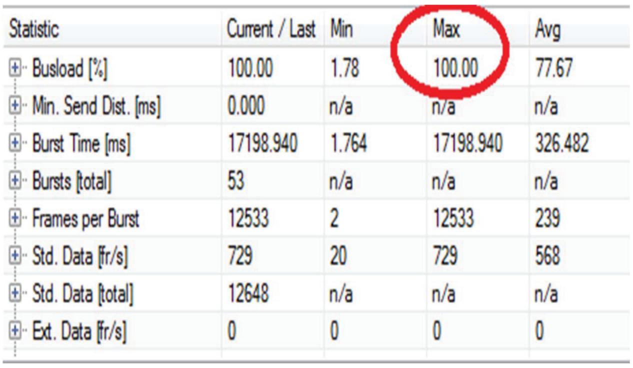


Fig. 5. Signal availability during attack



| Statistic | Current / Last | Min | Max | Avg |
|---|---|---|---|---|
| Busload [%] | 3.52 | 3.52 | 3.52 | 3.52 |
| Min. Send Dist. [ms] | 0.000 | n/a | n/a | n/a |
| Burst Time [ms] | 1.776 | 1.776 | 9.612 | 2.540 |
| Bursts [total] | 16 | n/a | n/a | n/a |
| Frames per Burst | 2 | 2 | 7 | 3 |
| Std. Data [fr/s] | 33 | 33 | 33 | 33 |
| Std. Data [total] | 43 | n/a | n/a | n/a |
| Ext. Data [fr/s] | 0 | 0 | 0 | 0 |

Fig. 6. The bus load before the attack



| Statistic | Current / Last | Min | Max | Avg |
|---|---|---|---|---|
| Busload [%] | 100.00 | 1.78 | 100.00 | 77.67 |
| Min. Send Dist. [ms] | 0.000 | n/a | n/a | n/a |
| Burst Time [ms] | 17198.940 | 1.764 | 17198.940 | 326.482 |
| Bursts [total] | 53 | n/a | n/a | n/a |
| Frames per Burst | 12533 | 2 | 12533 | 239 |
| Std. Data [fr/s] | 729 | 20 | 729 | 568 |
| Std. Data [total] | 12648 | n/a | n/a | n/a |
| Ext. Data [fr/s] | 0 | 0 | 0 | 0 |

Fig. 8. The bus load after the attack

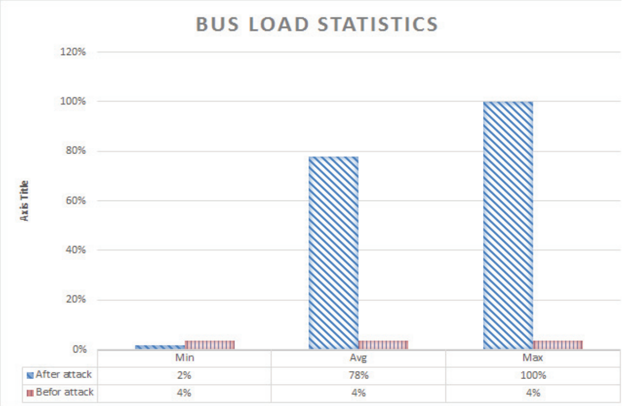


Fig. 9. The bus load chart after and before the attack

## V. PROPOSED SOLUTION

In order to mitigate this attack, we propose a solution that consists of three different stages. Each stage may be seen as a possible solution. However, deploying the full solution improves the system immunity against that attack.

The first stage of the solution is refusing a duplicate channel request nonce. Therefore, when $E_L$ records the channel request then resends it again, $E_E$ does not accept it as a valid channel request.

The second stage is to reconstruct the channel request message in such a way that represents both of sender and receiver ECU IDs. *Fig. 9* shows the new channel request frame. The protocol main version does not use the first 2 bytes of the CAN message payload. Our proposal is to concatenate the requester (i.e., receiver) ECU ID in the source field (1byte) and the responder (i.e., sender) ECU ID in the destination field. Both of the source field and destination field are concatenated to the request nonce and the resulted 6 bytes are occupied in the first 6 bytes of the CAN message payload. The message is encrypted by the $P_K$ that is shared between the requester and the responder. On the responder side, after it decrypts the message, it can get three main information. The responder gets the requester ID and it ensures that destination field contains its ECU ID. The next step is to increment the number of requests proposed from this requester, then the response process takes place by sending the five consecutive control messages. The request counter is used to limit the number of responses for each requester requests. It is the system configuration parameter, and it is set in our simulation 5 responses for each requester. Exceeding the maximum number of requests trails is encountered by ignoring this requests and any further requests come from this requester. Moreover, a feedback is reported to the vehicle driver to explain the severity of the current case [33].

In the final stage, a simple challenge-response procedure is created. The sender exploits the lowest traffic periods to transfer random values. When the sender suspects a request, it identifies its sender and challenges with the latest random value received from this sender. The authorized receiver responds with a hash of the latest random variable. The sender checks the validity of the received hash. In the case of correct hash, the sender continues responding to the request. Otherwise, the sender ignores the channel request. On the other side, the challenge is ignored in case the authorized receiver does not ask for a channel. We present a flow chart of the full proposed solution with its full stages in *Fig. 10*.

After applying the fix, the system can encounter the DoS attack, and the normal functionality is returned back even if the attacker is still replaying the request CAN message. The trace depicted in *Fig. 11* shows the system is recovered from the attack while the attacker still replaying the channel request message. The bus load is reduced after applying the fix where the max busload reached 100% and the average busload is reducing by time which means that the bus load is approximately returning back to its normal state by time as depicted in *Fig. 12*, where the average busload becomes 34% after 50sec only.

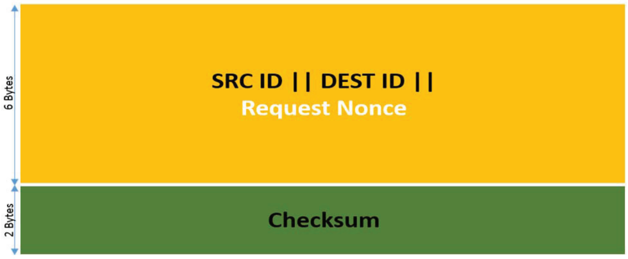


Fig. 7. The new format of the channel request message

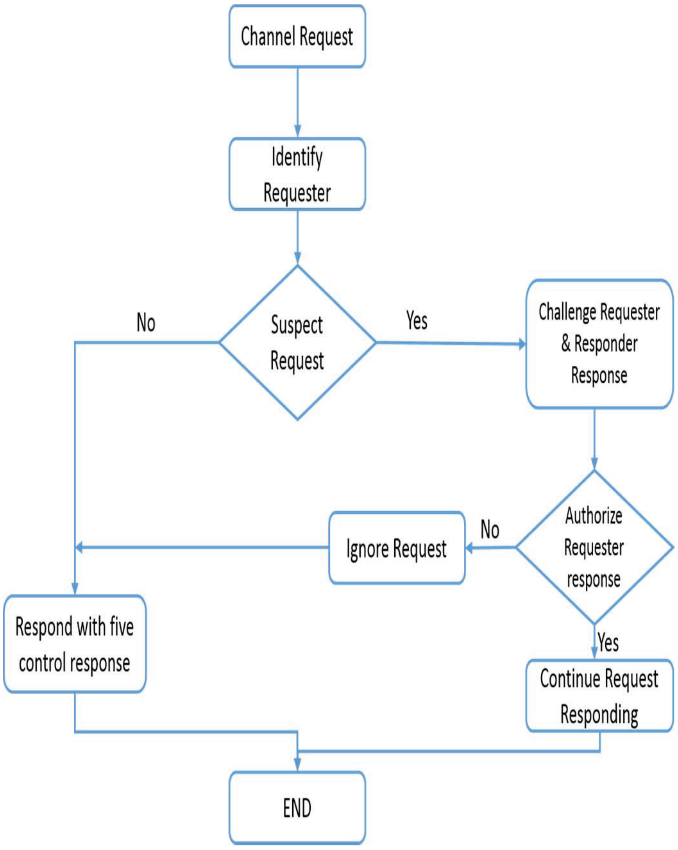
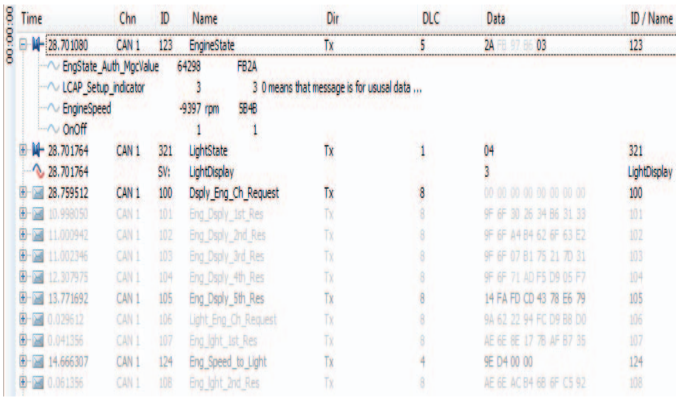
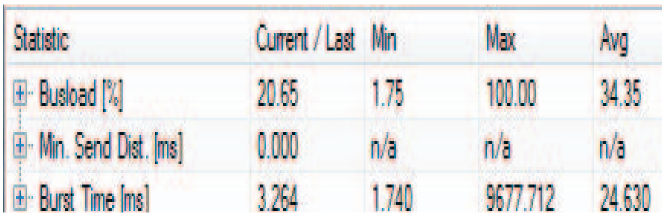

Channel Request
Identify Requester
Suspect Request
No
Yes
Challenge Requester & Responder Response
Authorize Requester response
No
Ignore Request
Yes
Respond with five control response
Continue Request Responding
END

Fig. 10. Flow chart of proposed solution



```
00:00:00
Time                      Chn     ID    Name                     Dir   DLC   Data                       ID / Name
28.701080                 CAN 1   123   EngineState              Tx    5     2A FB 97 B6 03             123
  EngState_Auth_MgcValue   64298      FB2A
  LCAP_Setup_indicator         3         3 0 means that message is for ususal data ...
  EngineSpeed              -9397 rpm  5B4B
  OnOff                        1         1
28.701764                 CAN 1   321   LightState               Tx    1     04                         321
28.701764                         SV:   LightDisplay                         3                          LightDisplay
28.759512                 CAN 1   100   Dsply_Eng_Ch_Request     Tx    8     00 00 00 00 00 00 00 00    100
10.998050                 CAN 1   101   Eng_Dsply_1st_Res        Tx    8     9F 6F 30 26 34 B6 31 33    101
11.000942                 CAN 1   102   Eng_Dsply_2nd_Res        Tx    8     9F 6F A4 B4 62 6F 63 E2    102
11.002346                 CAN 1   103   Eng_Dsply_3rd_Res        Tx    8     9F 6F 07 B1 75 21 7D 31    103
12.307975                 CAN 1   104   Eng_Dsply_4th_Res        Tx    8     9F 6F 71 AD F5 D9 05 F7    104
13.771692                 CAN 1   105   Eng_Dsply_5th_Res        Tx    8     14 FA FD CD 43 78 E6 79    105
0.029612                  CAN 1   106   Light_Eng_Ch_Request     Tx    8     9A 62 22 94 FC D9 B8 D0    106
0.041356                  CAN 1   107   Eng_lght_1st_Res         Tx    8     AE 6E 8E 17 7B AF B7 35    107
14.666307                 CAN 1   124   Eng_Speed_to_Light       Tx    4     9E D4 00 00                124
0.061356                  CAN 1   108   Eng_lght_2nd_Res         Tx    8     AE 6E AC B4 6B 6F C5 92    108
```

Fig. 11. Trace normal operations after fix



| Statistic | Current / Last | Min | Max | Avg |
|---|---|---|---|---|
| Busload [%] | 20.65 | 1.75 | 100.00 | 34.35 |
| Min. Send Dist. [ms] | 0.000 | n/a | n/a | n/a |
| Burst Time [ms] | 3.264 | 1.740 | 9677.712 | 24.630 |

Fig. 12. Bus load after fix

## VI. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a simple replay attack on the CAN. It depends on a weak point in the LCAP. Therefore, it can generate a legal behavior supported by LCAP for infinite time. This behavior introduces a DoS attack. DoS attack is considered one of the most dangerous attacks that a vehicle encounters, especially in operational condition. It may lead to huge financial losses or even human death. We also proposed a solution to mitigate this attack. The solution consists of three stages. It depends on getting the receiver identity and involves the vehicle driver to the current state of attack encountered by the vehicle and gets its feedback. The solution does not totally prevent the replay attack over CAN; however, it strengthens the weak point in LCAP in order to be more mature in facing such attack. In the future, we plan to test the immunity of the LCAP against active attacks in case of an authorized node is compromised by an attacker.

## REFERENCES

[1] J. Markoff. Google Cars Drive Themselves, in Traffic. New York Times, vol. 9, 2010.

[2] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. Mccoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage. Experimental Security Analysis of a Modern Automobile. pp. 1–16, 2010.

[3] W. Voss. A comprehensible guide to controller area network. Copperhill Media, 2005.

[4] S. Checkoway, D. Mccoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno. *Comprehensive Experimental Analyses of Automotive Attack Surfaces*. System, pp. 6–6, 2011.

[5] A. Wright. *Hacking cars*. Commun. ACM, vol. 54, no. 11, p. 18, 2011.

[6] T. Hoppe, S. Kiltz, and J. Dittmann. *Security Threats to Automotive CAN Networks – Practical Examples and selected Short-Term Countermeasures*. pp. 235–248, 2008.

[7] C. Miller and C. Valasek. *Adventures in Automotive Networks and Control Units*. Last Accessed from http//illmatics. com/ …, 2013.

[8] FlexRay Protocol Specification, FlexRay Consortium Std.Version 3.0.1, 2010.

[9] D. K. Nilsson, U. E. Larson, F. Picasso, and E. Jonsson. *A first simulation of attacks in the automotive network communications protocol FlexRay*. Adv. Soft Comput., vol. 53, pp. 84–91, 2009.

[10]L. Pike, P. C. Hickey, and J. Sharp. *Securing the Automobile : a Comprehensive Approach*. 2015.

[11]L. I. N. Consortium and others. *LIN specification package, revision 2.0*. Munich, Ger., 2003.

[12]M. Cooperation. *MOST Specification Rev. 3.0 E1*. Oct, 2009.

[13]W. Steiner, G. Bauer, B. Hall, and M. Paulitsch. *TTEthernet: Time-Triggered Ethernet*. Time-Triggered Communication. CRC Press, 2011.

[14]S. Chockalingam. *Alarming ! Security Aspects of the Wireless Vehicle : Review*. vol. 3, no. 4, pp. 200–208, 2014.

[15]P. Papadimitratos, V. Gligor, and J. P. Hubaux, Securing vehicular communications-assumptions, requirements, and principles in Workshop on Embedded Security in Cars (ESCAR), 2006.

[16] P. P. Koopman. A Case Study of Toyota Unintended Acceleration and Software Safety. 2014.

[17] A. Hazem and H. a H. Fahmy. *LCAP - A Lightweight CAN Authentication Protocol for Securing In-Vehicle Networks*. Escar 2012, 2012.

[18] A. Van Herrewege, D. Singelee, and I. Verbauwhede. *CANAuth - A Simple, Backward Compatible Broadcast Authentication Protocol for CAN bus*. Escar 2011, Embed. Secure. Cars, 2011.

[19] T. Ziermann, S. Wildernamm, and J. Teich. *CAN+: A new backward-compatible Controller Area Network (CAN) protocol with up to 16× higher data rates*. Des. Autom. Test, 2009.

[20] H. Schweppe, Y. Roudier, and B. Weyl. *Car2X communication: securing the last meter*. *Veh.* Technol. Conf., pp. 1–5, 2011.

[21] F. Information and P. Standards. *The Keyed-Hash Message Authentication Code*. no. July, 2008.

[22] C. Szilagyi and P. Koopman. A Flexible Approach to Embedded Network Multicast Authentication. 2nd Work. Embed. Syst. Secur., 2008.

[23] M. D. Hamilton, M. Tunstall, E. M. Popovici, and W. P. Marnane. *Side channel analysis of an automotive microprocessor*. In Signals and Systems Conference, 208.(ISSC 2008). IET Irish, 2008, pp. 4–9.

[24] A. Perrig, R. Canetti, D. Song, U. C. Berkeley, and I. B. M.

T. J. Watson. *Efficient Authentication and Signing of Multicast Streams over Lossy Channels*. vol. 28913, 2000.

[25] A. Perrig, D. Song, R. Canetti, J. D. Tygar, and B. Briscoe. *Timed efficient stream loss-tolerant authentication (TESLA): Multicast source authentication transform introduction*. 2005.

[26] A. Perrig, R. Canetti, D. Song, U. C. Berkeley, D. Fountain, and I. B. M. T. J. Watson. *Efficient and Secure Source Authentication for Multicast*. in Proceedings of the Internet Society Network and Distributed System Security Symposium, 2001, no. February, pp. 35–46.

[27] A. Perring, R. Szewczyk, V. Wen, D. Culler, J. D. Tygar,

A. Perrig, and R. Szewczyk. *SPINS : Security Protocols for Sensor Networks SPINS : Security Protocols for Sensor Networks*. pp. 521–534, 2002.

[28] M. Baugher and E. Carrara. *The use of timed efficient stream loss-tolerant authentication (TESLA) in the secure real-time transport protocol (SRTP)*. 2006.

[29] A. Studer, F. Bai, B. Bellur, and A. Perrig. *033_Flexible, Extensible, and Ef fi cient VANET Authentication*. Commun. Networks, vol. 11, no. 6, pp. 574–588, 2009.

[30] Q. Wang. *VeCure : A Practical Security Framework to Protect the CAN Bus of Vehicles*. pp. 13–18, 2014.

[31] S. Woo, H. J. Jo, and D. H. Lee. *A Practical Wireless Attack on the Connected Car and Security Protocol for In- Vehicle CAN*. IEEE Trans. Intell. Transp. Syst., vol. 16, no. 2, pp. 993–1006, 2015.

[32] T. Hoppe and J. Dittman. *Sniffing/Replay Attacks on CAN Buses: A simulated attack on the electric window lift classified using an adapted CERT taxonom*y. in Proceedings of the 2nd workshop on embedded systems security (WESS), 2007, pp. 1–6.

[33] A. Bouard, Lehrstuhl fur Sicherheit in der Informatik *Middleware-based Security for Future In-Car Networks*. 2014..