SecCAN: A Practical Secure Control Area Network for Automobiles

Mohammad Arman Ullah[1], Sheikh Ghafoor[1], Mike Rogers[1] and Stacy Prowell[2]
[1]Department of Computer Science, Tennessee Technological University, Cookeville, USA
[2]Computational Science and Engineering Division, Oak Ridge National Laboratory, Oak Ridge, USA
Mullah42@students.tntech.edu
sghafoor@tntech.edu
mrogers@tntech.edu
prowellsj@ornl.gov
DOI: 10.34190/IWS.21.106

Abstract: Controller Area Networks (CAN) are the backbone for communication among devices in modern automobiles. Although CAN bus is reliable, simple, low-cost, and low-power, which are desirable traits for embedded systems, it suffers from many security vulnerabilities. Unfortunately, security solutions for general purpose computers and networks do not generalize to CAN. First, many security solutions cannot be adopted by the automobile industry because they do not abide by constraints such as cost, real-time requirements, and backward compatibility. Furthermore, almost all the current proposed solutions violate one or more practical constraints of CAN, and so will be difficult for the automobile industry to adopt. Second, current research works in securing CAN only address a small subset of security vulnerabilities. We have developed a secure CAN protocol called SecCAN that uses lightweight encryption and message authentication using segmentation based shared secret group keys. Experiments with simulation and real ECU testbed show that our proposed protocol can effectively prevent masquerade and replay attacks.

Keywords: in-vehicle network security, CAN bus, embedded system network, vehicle security, controller area network

1. Introduction

A modern automobile consists of mechanical parts (such as the engine, chassis, brakes, etc.) that are controlled by software running on small embedded computers called Electronic Control Units (ECU). These ECUs are responsible for performing specific functions such as cruise control, deploying airbags during collisions, etc. In modern vehicles, virtually all operations are controlled by ECUs. For the proper functioning of a vehicle, these ECUs communicate with each other over a network using the Control Area Network (CAN) protocol. A modern vehicle has as many as 70 or more ECUs with millions of lines of code. These ECUs are used for different subsystems of a vehicle such as engine control, transmission control, airbags, cruise control, Anti-lock braking system, door control, etc.

CAN was built to be lightweight, functional, and reliable, and because of these characteristics has been a standard for in-vehicle networks for over two and a half decades. The designers of CAN assumed that all the ECUs would function according to their original purposes. However, CAN is not secure and has been shown to have many security vulnerabilities. For example, CAN is a broadcast protocol and a CAN message does not have a source or destination address and is therefore vulnerable to replay and masquerade attack. Also, CAN is a zero driven bus, and a compromised or malfunctioning ECU may take over the bus by continuously sending zeros to the bus, thereby disrupting all the communication among the ECUs. Furthermore, some of the ECUs are connected to the outside world via wireless or wired networks. For example, an attacker can manipulate the brakes in an automobile by wirelessly accessing an MP3 player that is connected to the CAN (Koscher et al, 2010).

Because of the aforementioned vulnerabilities, the CAN bus has the potential to cause a vehicle to operate poorly, or even to crash, because braking, acceleration, and steering are increasingly "drive by wire" systems managed by the CAN bus. Therefore, attackers can perform various life-threatening activities such as stopping the vehicle suddenly on an interstate, accelerating the vehicle abruptly, steering the vehicle left or right, enabling the windshield wiper-washer to blind the driver, etc. For example, hackers remotely controlled a Jeep Cherokee utilizing a faulty telematic unit and performed various malicious activities such as changing climate control, changing the radio station, turning on the windshield wipers, disabling the transmission, etc. (Greenberg et al, 2015).

Any pragmatic solutions to the security vulnerabilities of CAN must adhere to cost, hard real-time, and interoperability constraints. The automobile industry is very competitive, as a result, it is very sensitive to cost

increases. For example, if a solution increases the manufacturing cost even by 10 dollars per ECU (a seemingly small amount compared to the cost of a vehicle), it would increase the total cost of the vehicle by 700 dollars (a typical vehicle has about 70 ECUs). Such solutions are unlikely to be adopted by the auto industry.

We have developed a secure CAN protocol that addresses some of the existing vulnerabilities of CAN and prevents masquerade and replay attacks without violating any of the practical constraints. We have tested and evaluated our proposed technique using the BUSMASTER simulator as well as in a real testbed. The experimental results indicate that our proposed protocol can effectively prevent masquerade and replay attacks.

Our contributions are as follows. We give an overview of CAN and describe its vulnerabilities in Section 2. We discuss previous research works for securing CAN with a focus on how they are violating the practical constraints required by the current auto industry in Section 3. We present SecCAN, which is a practical secure CAN protocol that does not violate any practical constraints in Section 4. Section 5 describes the experiments and presents the results that show the efficacy of SecCAN.

2. Overview and vulnerabilities of can

CAN bus is a standard designed to enable communication among microcontrollers. CAN is a broadcast based shared medium protocol. The company Robert Bosch GmbH has published several CAN specifications (Bosch, 1991). The most common specification is CAN 2.0 that the company published in 1991. CAN 2.0 has two variants. CAN 2.0A is a standard format with an 11-bit message identifier, and CAN 2.0B is an extended format with a 29-bit identifier. In 2012, Bosch released CAN FD 1.0 (CAN with Flexible Data-Rate). CAN FD 1.0 is compatible with existing CAN 2.0 networks so that CAN FD 1.0 devices can coexist with CAN 2.0 devices.

Figure 1: CAN frame

2.1 Vulnerabilities of CAN

Traditionally, the major concerns in the design of CAN have been safety, reliability, and cost, whereas security has either been a minor concern or not considered at all. CAN does not have any built-in security support such as authentication, encryption, or prevention of Denial of Service (DoS) Attacks. Absent these important features, CAN suffers from vulnerabilities that attackers can exploit. Not only does CAN lack security features, but it also has features, such as a zero-driven bus, that are inherently easy to attack. The following describes in more detail both the features of CAN and features that are absent in the protocol that result in vulnerabilities that can be exploited.

- No source or destination address in CAN Frame
- No clock or sequence number to prevent replay
- Zero driven bus
- No encryption mechanism to encrypt CAN data
- No protection against malicious firmware or software updates
- No mechanism to detect a malicious device

2.2 Constraints and considerations for a secure implementation

To understand the details of CAN Bus security vulnerabilities, and be able to evaluate possible solutions and techniques others have proposed and implemented to-date, certain constraints that any implementation should meet to be a viable solution should be understood.

In particular, viable solutions should take into account that CAN Bus is an embedded real-time system in a market where cost savings are important. Violating constraints may yield implementations that few are willing or able to adopt. Following are the major constraints for implementing security in the CAN bus:

- **1. Monetary Cost:** As mentioned before, the markets utilizing CAN have competition and require the cost to be low. This was indeed a major focus the protocol addressed. Adaptations requiring substantial hardware changes, whether to fit new designs or to provide substantially more processing power for cryptographic operations, must consider the cost increase required.
- **2. Hard Real-time:** CAN is utilized in countless real-time environments. To ensure quick responses and bounded round trip times, any security mechanisms cannot significantly increase message-transmission and/or processing time.
- **3. Interoperability:** The length of a frame transmitted in a CAN bus is fixed, as well as the sizes of its fields and their position within the frame. Likewise, protocol behaviours, such as its arbitration method, are standardized. Solutions that require changes to the protocol's features or frame layout may make newly manufactured devices incompatible with existing CAN networks. Such solutions, unfortunately, will require complete re-implementations of existing CAN networks, and therefore, those solutions are not practical.

3. Related work

Various literature has been published to demonstrate CAN bus vulnerabilities experimentally. Furthermore, several works are published discussing different techniques to overcome the vulnerabilities of CAN bus. The techniques that are proposed in various literature are:

- Adding a MAC computed with the pre-shared secret key
- Using a counter to count the number of transmitted or received messages
- Using a counter to handle invalid authentication
- Encrypting messages
- Checking the hash value of software
- Checking the certificate of an ECU
- Using an intrusion detection system

Though these techniques solve some of the vulnerabilities of CAN, the implementation of these techniques violates some of the constraints of CAN. Figure 2 illustrates and summarizes, from the literature, different vulnerability mitigation techniques and constraints that their implementations violate.

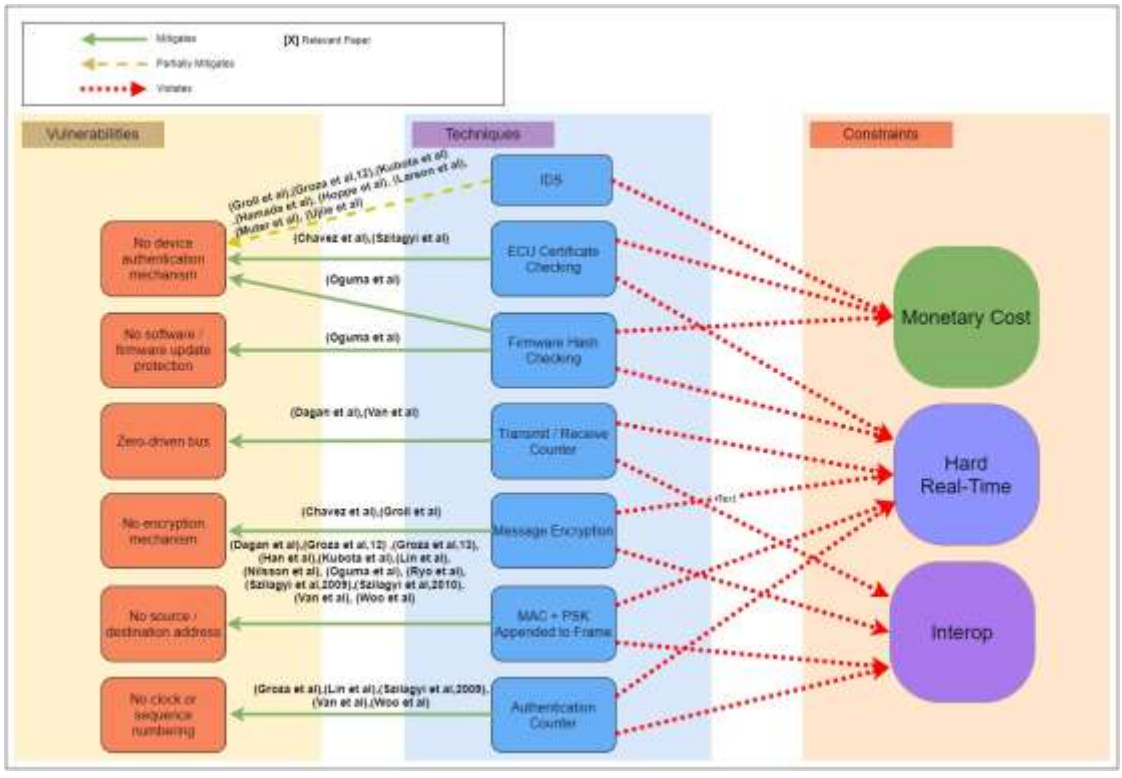

Figure 2: CAN vulnerabilities discussed in different literature

4. Proposed SecCAN protocol

The software architecture of a typical CAN implementation is shown in Figure 3. The Data layer gathers or generates data from sensors. On the sending side, the Encapsulation/Decapsulation layer creates a CAN message for the data and the Transmit/Receive layer transmits the message to the CAN bus. On the receiving side, the Transmit/Receive layer receives the message and the Encapsulation/Decapsulation layer extracts the data from CAN message and takes the necessary actions (for example, sending a reply message or sending a signal to an actuator). In the SecCAN protocol, we propose to replace the Encapsulation/Decapsulation layer with a secure version to ensure secure message transmission.

In normal operating condition, all the ECUs of a vehicle do not communicate with each other. The communications are generally localized among few ECUs. For example, a vehicle's cruise control system normally communicates with the rotating drive shaft, throttle, brake system etc. It does not communicate with the vehicle's entertainment system. In our proposed scheme, ECUs of a vehicle are segmented into groups based on their communication patterns. Instead of using pairwise shared keys for each pair of ECUs as done in (Szilagyi et al, 2009),(Szilagyi et al, 2010), each group shares a permanent common group MAC key (M_∂) and a permanent common group encryption key (K_∂), thereby reducing the number of keys, which simplifies key generation and key distribution. A group represents a vehicle subsystem, and the valid messages for a group are known by the manufacturer of the vehicle. Therefore, the groups are formed according to what messages are valid for each group.

4.1 Overview of key distribution and message transmission

An ECU in a group is selected as root and at the beginning of a session, the root generates a session MAC key, M_s, and a session encryption key, K_s. Then the root distributes the M_s and K_s to the other nodes in its group. The root sends the two session keys, M_s and K_s, in two successive messages by inserting a key in the data field of a CAN message and securing it using the permanent MAC key and the permanent encryption key. Other members of the group decrypt and authenticate the message using permanent keys and receive the session keys. Any further communication within the group is secured using the session keys. A session is defined as the time from the vehicle's ignition until it is switched off. A typical session may last from a few minutes (for a short drive such as going to the corner grocery store) to a few hours (long distance drive, such as going on a family vacation). Figure 4 shows the session key generation and distribution process.

The purpose of the message authentication code is to ensure the origin and data integrity of a message within a group. For any message, a 32-bit MAC is computed using the MAC key, a counter, and the data using a message digest algorithm. In our implementation we have used SHA-256 as the digest algorithm. The computed MAC is divided into two 16-bit parts; the 1st 16-bits is stored in the extended identifier field and the last 16-bits is stored in the CRC field. In SecCAN MAC serves two purposes: it provides source authentication as well as message integrity. Since MAC provides data integrity it also served the purpose of CRC. In normal CAN communication, out of 29 bits of the extended identifier, 11 bits are used for message ID and the rest are unused. SecCAN takes advantage of these unused bits to store the MAC without changing the current CAN frame format. The data field of the CAN message is encrypted using the encryption key. As a result, our proposed scheme is backward compatible and does not require any hardware change. The details of our proposed algorithm for key distribution and message transmission is provided in section 4.2.

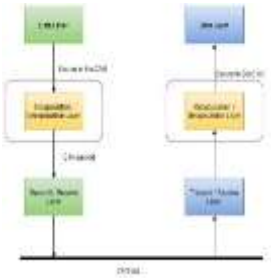

Figure 3: Software architecture

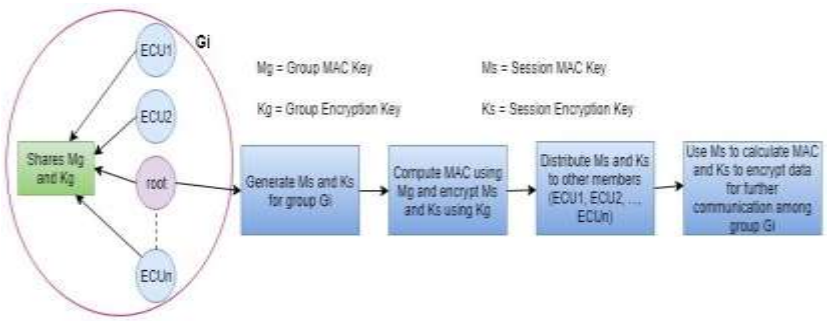

Figure 4: SecCAN simple diagram

4.2 Detail algorithm

The goal of our solution is implementing SecCAN by changing the software of an ECU without additional hardware. This solution enables the manufacturers or the dealers to change the software of existing vehicle's ECUs without incurring huge costs. In SecCAN, the ECUs of a vehicle are divided into two categories according to their vulnerabilities. The nodes that are connected to the outside world through the Internet, radio frequency, Wi-Fi, or Bluetooth are categorized as *exposed nodes*. The nodes which are not connected to the outside world are categorized as *internal nodes*. The group are formed according to the following two rules:

- 1. Two or more nodes will be in the same group if they communicate with each other in response to events.
- 2. The highest number of members of a group containing an exposed node must be two.

Our assumption is that it is more likely that an adversary will compromise an exposed node and send a malicious message than an internal node. If the session keys of an exposed node are compromised it will impact only one other node in that group thereby limiting the damage. Other nodes will not receive the messages sent by the compromised node as they belong to different groups (and thus have different session keys).

The root generates a new 64-bit Session MAC Key (M_s) and new 64-bit Session Encryption Key (K_s) and distributes the keys to other members of the group in each session. Following are the formal definitions of the necessary components for our SecCAN protocol followed by a detailed description of the algorithms for generation and distribution of session keys as well as subsequent transmission of messages.

G is the set of all communication groups of a system. The session number is initialized with a fixed number during the time of manufacturing and incremented every time when a vehicle is started. $G = \{G_1, G_2, G_3, \ldots, G_l\}$ where $G_i = \{e_1, e_2, e_3, \ldots, e_j\}$ and $G_i \subset E$ and $\exists e_j, e_j \in G_i : e_j \in E$.

GMK is the set of all group MAC keys. $GMK = \{M\partial_1, M\partial_2, M\partial_3, \ldots, M\partial_l\}$ where $M\partial_i$ is the group key of G_i and $M\partial_i \in GMk, M\partial_j \in GMk : M\partial_i \neq M\partial_j$.

GEK is the set of all group encryption keys.

$GEK = \{K\partial_1, K\partial_2, K\partial_3, \ldots, K\partial_l\}$ where $K\partial_i$ is the encryption key of G_i and $K\partial_i \in GEk, K\partial_j \in GEk : K\partial_i \neq K\partial_j$.

SMK is the set of group session keys.

$SMK = \{\exists M_s^j : i <= l$ and $j <= k\}$ and M_s^j is the jth session key of group i and initial value of $j = 0$.

SEK is the set of group session encryption keys.

$SEK = \{\exists K_s^j : i <= l$ and $j <= k\}$ and K_s^j is the jth session encryption key of group i and initial value of $j = 0$.

C is the set of counters.

$C = \{C_1, C_2, C_3, \ldots, C_m\}$ where C_i is the counter of message m_i where $m_i \in M$ and initially $C_i = 0$.

Algorithm for Session keys distribution in group G_i:

(1) Select an ECU in each group as session keys generator and distributor.
(2) $j = j + 1$ where j is the session number.
(3) Generate a Session MAC Key (M_s^j) using $M_s^j = f(M\partial_i, j)$.
(4) Compute 128-bit MAC using $MAC = HMAC(M\partial_i, M_s^j, j)$ and truncate it to 32 bit.
(5) Put the first 16-bit MAC into the extended identifier field, the last 16-bit MAC into CRC field and put the 64-bit M_s^j into data field to generate a message $MAC(m_j)$.
(6) Generate one time pad (OTP) by using $f(j, K\partial_i)$, where $f(j)$ is a one way function to change the session value.
(7) $Enc(MAC(m_j)) = MACW(m_j) \oplus OTP$ where $MACW(m_j)$ is a message m_j with MAC but without the first 11-bit header.
(8) Send the message to CAN bus.
(9) The other members of the group receive the message, verify the message integrity and origin integrity by using group key and encryption key.

(10) In the same way, Session Encryption Key K_{si}^{j} can be sent to other members of the group.

Receiving session keys in group G_i:

(1) $j = j + 1$ where j is the session number.
(2) Generate one time pad (OTP) by using $f(j, K\partial_i)$
(3) Get $MACW(m_j) = Enc(MAC(m_j)) \oplus OTP$
(4) Get M_{si}^{j} from $MACW(m_j)$
(5) Get MAC(ext) from $MACW(m_j)$
(6) Compute 128-bit MAC using $MAC = HMAC(M\partial_i, M_{si}^{j}, j)$ and truncate it to 32 bit.
(7) Verify MAC(ext) == *MAC*

Sending message m_j in group G_i at session t:

(1) $C_j = C_j + 1$.
(2) Compute 128-bit MAC using $MAC = HMAC(m_j, M_{si}^{t}, C_j)$.
(3) Produce $trunc(MAC)$ where trunc(MAC) is a function to truncate 128-bit MAC to 32-bit MAC.
(4) Put first 16-bit into extended identifier field and last 16-bit into CRC field and generate $MAC(m_j)$ where $MAC(m_j)$ is a message m_j with MAC.
(5) One Time Pad (OTP)= $f(C_j) \oplus K_{si}^{t}$ where $f(C_j)$ is a one-way function to change the counter value.
(6) $Enc(MAC(m_j)) = MACW(m_j) \oplus OTP$ where $MACW(m_j)$ is a message m_j with MAC but without the first 11-bit header.

Receiving message mj in group Gi at session t:

(1) $C_j = C_j + 1$.
(2) $OTP = f(C_j) \oplus K_{si}^{t}$.
(3) $MACW(m_j) = Enc(MAC(m_j)) \oplus OTP$ and construct $MAC(m_j)$ by adding the 11-bit header.
(4) Compute 128-bit MAC using $HMAC(m_j, M_{si}^{t}, C_j)$ and produce trunc(MAC).
(5) Extract MAC from $MAC(m_j)$ and compare the extracted MAC with the trunc(MAC).
(6) If both are the same, receive the message m_j. Otherwise, discard the message.

5. Implementation and evaluation

We have evaluated our protocol using a BUSMASTER (BUSMASTER, 2016)] simulator as well as in a real testbed. We have designed a set of experiments using 10 ECUs divided into 6 groups. One of the ECUs in each group has been selected as the root such that exposed node cannot be a root. The ECUs send and receive messages in a predefined order. Table 1 and Figure 5 shows the group formation and an example message scenario. We have conducted our experiments to evaluate the following:

- ***Group isolation****:* The objective of this evaluation is to ensure that an ECU can decrypt messages that it is supposed to receive i.e. only the messages sent by other ECUs in its group.
- ***Adherence to real time constraints****:* In real vehicles, there are hard real-time constraints for end-to-end message delays. These delays vary from 10 ms to 100 ms (Tindell et al, 1995) depending on what ECUs are involved in the communication. For example, communication involving the accelerator position (10ms) is much lower than the seatbelt warning (100ms). The objective of this experiment is to measure the overhead of SecCAN protocol in a real system and to estimate whether SecCAN would be able to abide by the real-time constraint in actual vehicles..

5.1 Group isolation

Table 1: Example message scenario

Sender	Signal	Receiver
Body Control Module	RKE (Remote Keyless Entry) Key press (Message ID: 0x101)	Transmission Control Module
Body Control Module	Ambient temperature (0x102)	Engine Control Module, Transmission Control Module

Sender	Signal	Receiver
Engine Control Module	Engine size(0x103), Engine RPM(0x104)	Body Control Module
Engine Control Module	Engine coolant temperate (0x105), Engine Size(0x103), VIN (0x106)	Transmission Control Module
Power Steering Control Module	Steering Position(0x107)	Anti-lock Braking System
Electronic Brake Control Module	Brake(0x108)	Anti-lock Braking System, Cruise Control Module
Air Bag Control Module	Airbag lamp status(0x109)	Mechanical Instrument Cluster
Engine Control Module	RPM (0x111)	Mechanical Instrument Cluster
Entertainment System	Radio(0x112)	Body Control Module

Figure 5: Group formation

We have evaluated group isolation using both BUSMASTER and the physical testbed. In the case of BUSMASTER, we have implemented an in-vehicle network with 10 ECUs and programmed them according to our protocol and a test scenario using the BUSMASTER simulation language. The simulation shows that an ECU can receive and decrypt the messages sent by ECUs in the same group, while the ECUs of other groups cannot decrypt the messages and thus ignore the messages. BUSMASTER has a feature of injecting CAN messages into the network. We have crafted normal CAN messages (unencrypted) and injected them into the network. These messages were ignored by the ECUs. We have also repeated the same experiment by crafting messages encrypted with fictitious keys. These messages were also ignored by the ECUS. For the second set of experiments, we have constructed a physical testbed, shown in Figure 6. Our testbed consists of two main types of hardware: ECUs and sniffers/injectors. The ECUs of the physical testbed were programmed to implement our protocol using MPLAB (MPLAB, 2018). We repeated the same experiments as in the BUSMASTER simulator. Except in case of experiments with the physical testbed we have used the push button switches to simulate sending of a message, and if a message is correctly received and decrypted the ECU lighted the LED. The experiment shows that ECUs of the same group successfully received the messages and lit the LED while the ECUs of other groups did not receive the messages successfully and did not light the LED. We have also injected CAN messages through the Kvaser leaf from the BUSMASTER to simulate message injection into the network by adversaries. These messages were not received by the ECUs. These experiments indicate that our protocol not only successfully secured the messages but also isolated them within a group. As a result, even if the session keys of a group are compromised the damage is contained within a group.

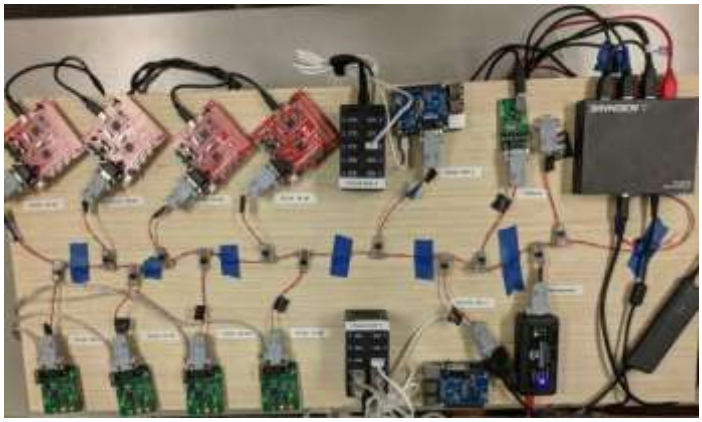

Figure 6: Physical testbed

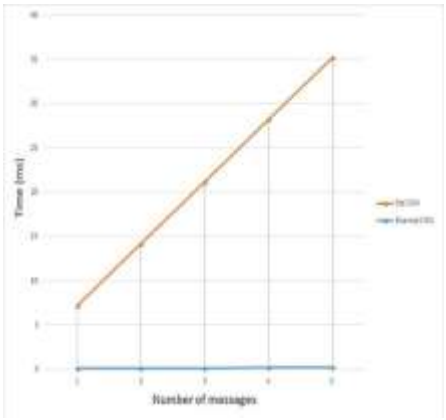

Figure 7: Normal CAN and SecCAN

5.2 Adherence to real time constraints

This experiment also used the physical testbed. The use of a physical testbed was particularly important for this experiment because BUSMASTER cannot simulate the processing power of physical ECUs. Whether the protocol can adhere to real-time constraints is directly impacted by both the power of the device on which the protocol will run as well as the computational complexity of the protocol.

For this experiment, we instrumented the code that was flashed on the ECUs to measure transmission time and compute time. For this implementation, all computation is done by the protocol's algorithms. In other words, in a normal ECU, some computational time might be used to compute temperatures, rotations, etc., but we did not include such functionality in our testbed.

Furthermore, we ran two sets of experiments. One set of experiments generated normal CAN traffic, and the second set of experiments generated CAN traffic using SecCAN, our secure protocol.

The results of the experiments are shown in Figure 7. The X axis shows the number of messages generated. The experiment, for 1 through 5 messages, was executed multiple times, and the average time to send and receive the messages was recorded, as shown in the Y axis. The time to send and receive the messages includes the computation time at both the sender and receiver as well as the message transmission time. However, transmission time for the normal CAN and SecCAN are the same because the message size does not change. Therefore, the difference in the two lines in the graph are because of the implementation of our protocol.

The SecCAN protocol adds additional overhead for communication. From figure 7, it can be seen that the end-to-end delay is 6.8 ms. Since in our prototype testbed we have used ECU chips and wires that are used by manufacturers in real vehicles, SecCAN performance overserved on the testbed should be comparable to real vehicles. Also, our experiments show that the end-to-end message delay is less than the minimum requirements of real-time constraints of CAN (Tindell et al. 1995). The data about message delays in real vehicles were not available at the time of writing the paper (as these data are manufacturer specific and is not available publicly). Due to the unavailability of data, we couldn't compare our experimental results with real systems.

6. Conclusion and future work

We have presented a discussion on the existing research work in securing CAN with a focus on how they are violating the practical constraints to be adopted by the current automobile industry. We have presented a secure CAN protocol called SecCAN. The basic principles of SecCAN are segmentation-based message authentication codes and encryption using shared group keys. The evaluation of our proposed scheme using the BUSMASTER simulator and a real ECU-based testbed indicate that it can effectively prevent masquerade and replay attacks. The main advantage of our proposed scheme is that it can be easily adopted in a vehicle by reflashing the ECUs without changing any hardware. Also, our proposed scheme does not violate hard real-time deadlines required by CAN and it does not change the CAN packet format. However, our proposed scheme does not address DoS attacks, nor does it detect malicious software updates. However, we believe that our proposed scheme is a step in the right direction. Our future plans include developing a protocol with additional hardware that can effectively prevent DoS attack and software modification attacks in addition to masquerade attacks, replay attacks and snooping.

References

Bosch, R. (1991). CAN specification version 2.0. Stuttgart.

BUSMASTER. (2016) *BUSMASTER* [Online]. Available at https://rbei-etas.github.io/busmaster/ [Accessed 25 Sept 2020].

Chavez, M.L., Rosete, C.H. and Henriquez, F.R., 2005, February. Achieving confidentiality security service for can. In 15th International Conference on Electronics, Communications and Computers (CONIELECOMP'05) (pp. 166-170). IEEE.

Dagan, T. and Wool, A., 2016. Parrot, a software-only anti-spoofing defense system for the CAN bus. ESCAR EUROPE, p.34.

Greenberg, A., 2015. Hackers remotely kill a jeep on the highway—with me in it. Wired, 7, p.21.

Groll, A. and Ruland, C., 2009, June. Secure and authentic communication on existing in-vehicle networks. In 2009 IEEE Intelligent Vehicles Symposium (pp. 1093-1097). IEEE.

Groza, B., Murvay, S., Van Herrewege, A. and Verbauwhede, I., 2012, December. Libra-can: a lightweight broadcast authentication protocol for controller area networks. In International Conference on Cryptology and Network Security (pp. 185-200). Springer, Berlin, Heidelberg.

Groza, B. and Murvay, S., 2013. Efficient protocols for secure broadcast in controller area networks. IEEE Transactions on Industrial Informatics, 9(4), pp.2034-2042.

Hamada, Y., Inoue, M., Horihata, S. and Kamemura, A., 2016, November. Intrusion detection by density estimation of reception cycle periods for in-vehicle networks: A proposal. In 14th Int. Conf. on Embedded Security in Cars (ESCAR 2016), Munich, Germany.

Han, K., Weimerskirch, A. and Shin, K.G., 2015. A practical solution to achieve real-time performance in the automotive network by randomizing frame identifier. Proc. Eur. Embedded Secur. Cars (ESCAR), pp.13-29.

Hoppe, T., Kiltz, S. and Dittmann, J. (2008). Security threats to automotive CAN networks–practical examples and selected short-term countermeasures. Computer Safety, Reliability, and Security, Springer, pp.235–248.

Koscher, K., Czeskis, A., Roesner, F., Patel, S., Kohno, T., Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H. and Savage, S., 2010, May. Experimental security analysis of a modern automobile. In 2010 IEEE Symposium on Security and Privacy (pp. 447-462). IEEE.

Krawczyk, H., Canetti, R. and Bellare, M. (1997). HMAC: Keyed-hashing for message authentication.

Kubota, T., Shiozaki, M. and Fujino, T., 2016. Proposal and Implementation of Key Exchange Protocol suitable for In-Vehicle Network based on Symmetric Key Cipher using PUF as Key Storage. In escarEU2016.

Larson, Ulf E, Nilsson, D.K. and Jonsson, E. (2008). An approach to specification-based attack detection for in-vehicle networks. In: IEEE. pp.220–225.

Lin, C.-W. and Sangiovanni-Vincentelli, A. (2012). Cyber-security for the controller area network (CAN) communication protocol. In: IEEE. pp.1–7.

MPLAB. (2018). MPLAB [Online]. Available at http://www.microchip.com/mplab/mplab-x-ide [Accessed 27 Sept 2020]

Müter, M. and Asaj, N. (2011). Entropy-based anomaly detection for in-vehicle networks. In: IEEE. pp.1110–1115.

Nilsson, D.K., Larson, Ulf E and Jonsson, E. (2008). Efficient in-vehicle delayed data authentication based on compound message authentication codes. In: IEEE. pp.1–5.

Oguma, H., Yoshioka, A., Nishikawa, M., Shigetomi, R., Otsuka, A. and Imai, H. (2008). New attestation based security architecture for in-vehicle communication. In: IEEE. pp.1–6.

Ryo, K., Takada, H., Mizutani, omohiro, Ueda, H. and Horihata, S. (2015). SecGW - secure gateway for in-vehicle networks[22] Szilagyi, C. and Koopman, P. (2009). Flexible multicast authentication for time-triggered embedded control network applications. In: IEEE. pp.165–174.

Szilagyi, C. and Koopman, P. (2010). Low cost multicast authentication via validity voting in time-triggered embedded control networks. In: ACM. p.10.

Tindell, K., Burns, A. and Wellings, A. (1995). Calculating controller area network (CAN) message response times. Elsevier, pp.29–34.

Ujiie, Y., Kishikawa, T., Haga, T., Matsushima, H., Wakabayashi, T., Tanabe, M., Kitamura, Y. and Anzai, J. (2016). A method for disabling malicious CAN messages by using a CMI-ECU. SAE Technical Paper.

Van Herrewege, A., Singelee, D. and Verbauwhede, I., 2011, November. CANAuth-a simple, backward compatible broadcast authentication protocol for CAN bus. In ECRYPT Workshop on Lightweight Cryptography (Vol. 2011, p. 20).

Woo, S., Jo, H.J. and Lee, D.H. (2015). A practical wireless attack on the connected car and security protocol for in-vehicle CAN. IEEE Transactions on Intelligent Transportation Systems, 16, pp.993–1006.

Chuck Easttom is the author of 30 books and 70 papers, and 22 patents. He holds a D.Sc, Ph.D. , and master's degrees. He is a Distinguished Speaker of the ACM, Distinguished Visitor of the IEEE, as well as a Senior Member of the ACM and the IEEE. He is an adjunct lecturer for Georgetown University

Éric Filiol is professor at ENSIBS, Vannes, France and at National Research University Higher School of Economics, Moscow, Russia in the field of information and systems security. He is also a senior consultant in cyber security and intelligence. He is editor-in-chief of the research journal in Computer Virology and Hacking Techniques published by Springer.

Ryan Gabrys is a scientist with the Naval Information Warfare Center Pacific. His research interests include coding theory with applications to systems, storage and security.

Dr Sheikh Ghafoor is a professor of Computer Science at Tennessee Tech University. His main area of research is High Performance Computing, Cyber Physical, System Security, Computational Earth Science, and Computer Science Education. His research in these areas has been funded by NSF, NASA, DoD, and DoE.

Dr. Scott Graham is an Associate Professor of Computer Engineering at the Air Force Institute of Technology, USA. His research interests center on cyber physical systems, computer architecture, networks, and security for critical infrastructure protection.

Prof. Greiman is an Assistant Professor at Boston University, where she teaches and conducts research in international law and global cyber law and governance. She formerly served in high level appointments at the U.S. Department of Justice and as legal adviser to the U.S. Department of State and USAID in Eastern and Central Europe, Africa and Southeast Asia.

George Grispos is an Assistant Professor of Cybersecurity in the School of Interdisciplinary Informatics at the University of Nebraska at Omaha. He obtained his PhD in Computing Science from the University of Glasgow, Scotland. His research interests include digital forensics, security incident response, information assurance, and applied computing science.

Dr. Thomas Heverin is a cybersecurity teaching professor for Drexel University. He holds a Ph.D. focused on cybersecurity and the CISSP. He has conducted research on cyber threats on industrial control systems and leads the CyberCorps program at Drexel. He also served in the U.S. Navy as a ship officer.

Michael Bennett Hotchkiss is an independent researcher with interests in the psychology of influence, criminology, and nation-state disinformation; especially in the context of Russian active measures. Michael earned a Master of Organization Development from Bowling Green State University; and a Bachelor of Arts in Psychology from the University of Connecticut, graduating Phi Beta Kappa.

Eduardo Arthur Izycki International Relations M.A. Student at the University of Brasília (UnB) and public servant. Eduardo Izycki worked on developing solutions for risk assessments in the cycle of major events in Brazil (2012/2016). He currently works in the Critical Infrastructure Protection Coordination of the Brazilian Institutional Security Office (GSI).

William A. Johnson is a direct-admit PhD and Cyber Corps scholar at Tennessee Tech University. His research areas include Remote Attestation for embedded devices, cryptographic solutions for emerging networks, and ethical hacking. He received his bachelors from Tennessee Tech University in Computer engineering in 2018.

Nida Kazi graduated with a Master of Information Systems from John Hopkins University, in August 2020 and received her Bachelor of Engineering in Computer Science from the University of Pune. Her research has been focused in the fields of Cybersecurity and Cryptography. She is an avid cook and voracious reader, and lives in Virginia with her best friends.

Anne Kohnke, Ph.D. is an Associate Professor and the Principal Investigator for the Center of Academic Excellence in Cyber Defense (CAE-CD) at the University of Detroit Mercy. Dr. Kohnke's research is focused in the area of cybersecurity, risk management, and cybercrime. She has recently coauthored six books and several peer-reviewed journal articles in this discipline.

Daniel Koranek is a research computer scientist with the Air Force Research Laboratory. He holds a B.S. in computer science from Cedarville University and an M.S. in Cyber Operations from the Air Force Institute of Technology (AFIT), and is in doctorate studies at AFIT. Daniel's research interests are in embedded systems security and machine intelligence.

MSc Tiina Kovanen is a PhD student at the university of Jyväskylä. She is interested in various cyber security topics for different cyber-physical systems. Currently she is working towards her degree by studying possibilities and challenges related to ships' remote pilotage environment, ePilotage.