SCIS 2016 2016 Symposium on
Cryptography and Information Security
Kumamoto, Japan, Jan. 19 - 22, 2016
The Institute of Electronics,
Information and Communication Engineers

A Countermeasure against Spoofing and DoS Attacks based on Message Sequence and Temporary ID in CAN

Soohyun Ahn * Hakju Kim † Jeseong Jeong * Kwangjo Kim *

Abstract: The development of Information and Communications Technologies (ICT) has affected various fields including the automotive industry. Therefore, vehicle network protocols such as Controller Area Network (CAN), Local Interconnect Network (LIN), and FlexRay have been introduced. Although CAN is the most widely used for vehicle network protocol, its security issue is not properly addressed. In this paper, we propose a security gateway, an improved version of existing CAN gateways, to protect CAN from spoofing and DoS attacks. We analyze sequence of messages based on the driver's behavior to resist against spoofing attack and utilize a temporary ID and SipHash algorithm to resist against DoS attack. For the verification of our proposed method, OMNeT++ is used. The suggested method shows high detection rate and low increase of traffic. Also, analysis of frame drop rate during DoS attack shows that our suggested method can defend DoS attack.

Keywords: CAN, Security Gateway, Spoofing, DoS

1 Introduction

The development of ICT has affected various fields. Among these, automotive field has also been greatly affected after the combination with ICT. Electronic Control Unit (ECU) that controls engine, break, and transmission in vehicle was introduced due to ICT and it has significantly contributed to digitalization of vehicle. The number of ECUs in recent vehicles is said to be up to 70, and the importance of ECU will continue to increase. Although controlling a vehicle became convenient due to each ECU, vehicle network protocol was needed to communicate with other ECUs. To solve this problem, vehicle network protocol like CAN, LIN, FlexRay were proposed. Among these protocols, CAN is the most widely used for vehicle network protocol, and is regarded as the worldwide standard.

Although the introduction of ECU made vehicle control more efficient and convenient, security issue in CAN is not properly addressed at its design stage and security challenge happens. Vehicle security research has gained the importance due to recent hacking incidents such as GM and Chrysler vehicles. Because CAN has certain vulnerable characteristics, it is easy to attack vehicles employing such a system. Some attacks are attempted such as spoofing and DoS attacks. These attempts are proved through many recent research[1][2][3] and countermeasures about security in vehicle are important. To solve this problem, various research [4][5] have been undertaken. However, because it is difficult to build an experimental environment due to the high cost involved and accessibility issues, some ideas remain unproven or have inherent issues such as the volume of traffic, the cost of utilization, *etc.*

Therefore, we propose a heuristics method that can solve the issues such as traffic and verification, *etc.* By introducing a security gateway that modifies the existing gateway that CAN currently uses, we can defend against spoofing and DoS attacks. Furthermore, by an experiment using OMNeT++, the proposed method can effectively defend against spoofing and DoS attacks and demonstrates its effectiveness by showing only a low increase of traffic.

The rest of this paper is organized as follows: Section 2 describes the background basic information about CAN, vulnerabilities in CAN, and related work about defense against attacks in CAN. In Section 3, some assumptions are defined and our method that can defend against spoofing and DoS attacks through a security gateway are proposed. OMNeT++, which is used in experiments and our implmentation is described in Section 4. Section 5 details the results and discusses its limitations. Finally, the conclusion and future work are discussed in Section 6.

2 Background

2.1 CAN

CAN is a standard vehicle network and was proposed by R. Bosch at the 1982 conference of the Society of Automotive Engineers (SAE). Bosch released CAN 2.0, which is a current specification, in 1991 and submitted it to the International Organization for Standardization (ISO). "ISO 11989" was released as a standard in 1991. In addition, when an amendment was submitted to ISO, ISO 11989 was extended and a CAN that had a 29 bit ID field was introduced called "CAN 2.0B."

2.2 Vulnerabilities in CAN

1) Bus architecture

* Graduate School of Information Security, KAIST, 291 Gwahak-ro, Yuseong-gu, Daejeon, 34141, Korea {ahn1015, taegm01, kkj}@kaist.ac.kr
† Computer Science Dep't, KAIST, 291 Gwahak-ro, Yuseong-gu, Daejeon, 305-701, Korea. ndemian@kaist.ac.kr

CAN uses bus architecture, so it has broadcast characteristic which means that all nodes in CAN receive messages without any restriction. Therefore, it is easy to execute a sniffing attack. A node can receive all messages from the bus, so, if the attacker maliciously installs an ECU, attacker can monitor all messages in the CAN bus. In addition, using the On-Board Diagnostics-II (OBD-II), which is used for diagnosis of a vehicle, it is easy to getting message from the CAN bus due to the broadcasting environment. An attacker can analyze the meaning of the collected messages and create new messages like the ones used in a spoofing attack.

2) Arbitration process

Next, CAN offers an autonomous arbitration process to avoid collisions between messages when sending a message. The arbitration process is performed using an arbitration field in the data frame. There are two levels (dominant and recessive) in CAN, and the dominant(0) level has a higher priority than the recessive(1) level. Due to this characteristic, a low ID has higher priority in the CAN. Thus, the arbitration field in data frame is compared with other data frames using two levels and the lower ID can send messages to CAN bus. Using this characteristic, the attacker is able to find which ID has the highest priority in the CAN and creates messages that have the highest priority in the CAN. When the attacker continuously sends such messages to the CAN bus, other nodes cannot send messages to the CAN bus. As a result, the vehicle cannot be operated, because messages cannot be sent.

3) No Authentication

Finally, the biggest vulnerability in CAN is that there is no message authentication. Basically, the ID in the CAN frame is used for the transmission and arbitration processes, but this ID does not represent the sender. Because this ID is used for whether messages are received at the receiver, nodes in the CAN cannot know where messages in the CAN originated. Therefore, an attacker can send fake messages that use message IDs obtained through sniffing. A receiver cannot verify the authenticity of any messages, because the attacker's message ID is valid. Thus, spoofing and DoS attacks can be easily executed.

2.3 Previous work

Many ideas have been suggested to defend against spoofing and DoS attacks that are security threats in the CAN. These ideas can be divided into two classes; methods using characteristics of CAN, methods using cryptographic algorithms. Below are representative ideas.

2.3.1 Class I: Methods using Characteristics of CAN

There are two defense methods using characteristics of CAN. First method uses intrusion detection system (IDS) that uses characteristics of CAN such as ID, periodicity of messages *etc.* Second method uses broadcast characteristic in CAN to detect attacks. The IDS method has been suggested to detect spoofing and DoS attacks in the CAN. In IDS, the transmission characteristics in CAN is deployed, unlike the existing method in the wired or wireless network environment that examines the content of messages. Because messages in CAN are transferred periodically, a method[11] using the periodicity of messages has been proposed. In addition, a detection technique using a white-list has been suggested. The white-list technique [8] makes a table, which stores the messages that are used in CAN, to detect attacks by checking each ID in the table. This method is used in each ECU or gateway in the CAN bus. Also, by combining the proposed methods mentioned above, new methods[9][13][14] that checks the periodicity of messages, the ID, *etc* has been suggested. Finally, there is a method[15] that has a monitor mode and a verification mode; messages are monitored using IDs in the monitor mode. If there are suspect messages, the mode is changed to the verification mode and verification is processed using MAC.

A method[10] utilizes the characteristic of broadcasting in CAN was suggested. It checks whether a message is made from itself using the own message ID. If messages have not originated from its own self, it notifies the other ECUs in the CAN by sending error frames. In other words, if the attack ECU creates messages using other ECU's ID, the ECU with the corresponding ID can know that messages are invalid by checking the ID of messages. These ideas can defend attacks, but, they have security issues such as modification of existing ECU, and its verification.

2.3.2 Class II: Methods using Cryptographic Primitives

There are two defense methods using cryptographic primitives. One method uses message authentication code (MAC) and another is to use existing cryptographic algorithms. Using MAC to check message integrity has been applied to CAN. Various ideas[7][16][18] that use hash-based MAC, CBC-MAC, or autonomously developed MAC algorithm have been proposed. However, they encounter similar problems like increasing traffic. The suggested ideas use any length larger than 64 bit, which is the maximum length of data in CAN. Thus, additional messages are created for the authentication of integrity. Due to the additional messages, the traffic in CAN will increase.

Another proposed method uses existing cryptographic algorithms. A method[12] using AES, which is a type of cryptographic algorithm, was suggested. In addition, there is a method[6] that shares a pair-wise secret key between ECUs to encrypt messages. In this situation, forgery prevention and the authentication of messages are both performed by the MAC. The ideas suggested above ensure confidentiality and integrity that CAN does not offer, but the ideas have overhead such as padding value and increasing traffic. If the padding value is always same, an attacker can easily learn the value to use in future attacks on CAN. Another problem is the increase in traffic. For example, when messages are encrypted by AES, the data length must be 128 bit. AES is a block cipher and requires that data length is a minimum of 128 bit. Because data length in

frames is up to 64 bit in CAN, the length must be extended to use AES. Therefore, additional message will be created.

3 Our Approach

3.1 Assumptions

Below are the required assumptions for the method proposed in this paper.

1) The security gateway and the ECUs in the CAN bus already share a specific derivation function and a secret key.

   - Basically, the manufacturer can insert anything in the ECU and gateway; therefore, they can insert a key into the ECU and gateway during the manufacturing process.

2) The IDs of the security gateway and the ECUs have already been determined.

   - It is also possible that the manufacturer can determine the ID of the ECU and gateway during the manufacturing process.

3) The security gateway stores the information about the ECU that exists in the domain.

   - During the manufacturing process, information about ECU can be inserted into the security gateway by the manufacturer.

3.2 Attack Model

In this paper, our attacking model is assumed as follow: The attacker performs attacks using the OBD-II port. An OBD-II port is installed in vehicles for diagnostic purpose, which is connected to the CAN bus. Thus, an attacker can attack the CAN bus by just connecting to the OBD-II port. Initially, this attack was complicated, but an attack using CANtact in Figure 1 is feasible now. Because CANtact is small size, it is difficult to be detected by a driver.

Figure 1: CANtact

In this situation, the attacker can know the ID of the ECUs in the CAN bus and the meaning of each ID. As mentioned above, because CAN is a broadcasting environment, it is easy to sniff messages. Therefore, the attacker can easily know the IDs of ECUs.

There is another method to attack a vehicle. The attacker can insert malicious ECU into the vehicle. The attacker can do it if attacker has the knowledge of vehicle or through mechanic. It is harder for the driver to detect attack ECU, because it is not visible to the driver. In this situation, the attacker connects to the malicious ECU using wireless device such as smart phone, tablet, and laptop. The attacker can easily know various information in CAN bus such as ID, messages. Based on this information, the attacker can do sniffing, spoofing, replay, and DoS attacks.

3.3 Defense Scheme

Before explaining our method in detail, we describe the basic background.

3.3.1 Defense using Driver's Behavior

Our method is based on driver's behavior and the methods using driver's behavior have been also suggested. There are some research[19][20][21] about analyzing driver's behavior through collected data in vehicle. The suggested method uses data based on driver's behaviors such as angle of handle, speed of vehicle revolution, and position of pedal, *etc.* These data can give clue of driver's behavior and would be used for detecting attack. This detection method can be used with machine learning or data mining algorithms. Kwak *et al.* [22] is one of the methods using driver's behavior and data mining. They suggested a framework for detecting attacks. In this framework, data from driver's behavior are collected and used for making or updating detection rule. However, because it is not implemented, effective and efficiency of their method are not verified.

3.3.2 Security Gateway

As the number of ECUs is increasing in the modern design of CAN bus, it is required for seperated domains to properly manage the enlarged traffic. Thus, security gateways are introduced in CAN like Figure 2.

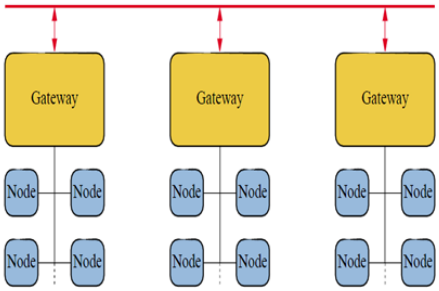

Figure 2: Gateway in CAN

The security gateway adds security functions to the gateways. There are two functions in the security gateway. One function is authenticating each ECU. In this situation, security gateway authenticates whether ECU in CAN is valid or not. Another function is detecting attacks in CAN. At this time, security gateway detects attacks such as spoofing or DoS with detection algorithms.

Because security gateway can do more things than ECU, it is useful to defend attacks. Therefore, the security gateway in this paper is used for defending attacks. Basically, the security gateway monitors its domain and defends against spoofing and DoS attacks. Figure 3 shows the structure of the security gateway,

which consists of four components. The role of each component is as follows:

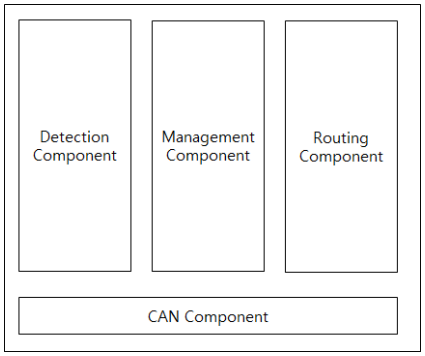

Figure 3: Structure of security gateway

1) Detection component

   - Component for detecting spoofing and DoS attacks.

2) Routing component

   - Component for arranging transfers to other domains.

3) Management component

   - Component for managing tables and creating messages and seeds

4) CAN component

   - Component for CAN communication

3.3.3 Spoofing Defense

The sequence of messages based on the driver's behavior is used to defend against spoofing attacks. There are various ECUs in CAN, and these ECUs are used for specific purposes. For example, different ECUs are in charge of the door, lights, brake, engine, *etc.*

Each ECU is used for a specific purpose. We think how each ECU is associated with the others and defines the sequence of messages determined by the driver's behavior. Thus, a spoofing attack can be detected by monitoring the sequence of messages. Because an attacker cannot know what ECUs exist in the CAN bus, the attacker cannot know the flow of messages. Thus, when the attacker tries to use a spoofing attack, this attack violates the sequence of messages, which can be detected by monitoring the sequence of messages. Using this information, the sequence of driver's behavior can be made according to what ECUs represent the basic ECUs used in the vehicle. Table 1 represents the basic ECUs that are used in vehicle.

Table 2 shows 10 driver's behaviors based on the basic ECU. For example, if the driver stops his vehicle, brake, gears, and engine will operate in that order. Thus, the Electronic Brake Control Module ECU, Transmission ECU, and Engine Control Module ECU will operate in this order and the related messages will flow in the CAN bus. Using this sequence, the detection component in the security gateway monitors the CAN bus. When the first message in the table is detected, the next message in table will be monitored. If

Table 1: Basic ECUs

	ECU
①	Electronic Stability Control ECU
②	Transmission ECU
③	Engine Control Module ECU
④	Electronic Break Control Module ECU
⑤	Electric Power Steering ECU
⑥	Throttle ECU
⑦	Instrument Cluster ECU
⑧	Electric Parking Break ECU
⑨	Light Control Module ECU
⑩	Adaptive Front Lighting ECU

the expected message is not witnessed and other message is discovered, the detector considers the possibility of a spoofing attack. However, because this message may be sent by a normal ECU, a verification process must be performed. At this time, the verification process will be performed through the management component and SipHash, which is a keyed hash algorithm that uses a simple operation such as Add-Rotate-Xor (ARX). SipHash is also optimized for a short message, resulting in a 64-bit tag value. Therefore, it is suitable for constrained devices such as an ECU.

The security gateway requests the hashed value of the suspect message. The ECU that sent the suspect message calculates the hashed value and sends it to the security gateway. Then, the management component in the security gateway calculates the hashed value of the suspect message and compares it to the received hashed value. If the two hashed values are same, verification succeeds, and if the hashed values are different, verification fails. In other words, a verification failure can be considered as an attack. At this time, the attacker does not know this secret key, so cannot create same hashed value. Thus, the attacker cannot send a verification message, and the attack can be detected.

Table 2: Sequence of messages based on driver's behavior

Driver's behavior	**Sequence of messages**
1) Usual	① → ② → ③
2) Left/Right turn, U-tern	④ → ② → ⑤ → ⑥ → ③
3) Ignition	⑥ → ③ → ② → ⑥ → ⑦
4) Stop	④ → ⑥ → ③ → ②
5) Acceleration	② → ⑥ → ③ → ⑦
6) Deacceleration	④ → ⑥ → ③ → ⑦
7) Parking/stop	④ → ② → ⑥ → ③ → ⑧ → ⑦
8) Light	⑨ → ⑩ → ⑦
9) Backward movement	④ → ② → ⑤ → ⑦ → ②
10) Change of line	⑨ → ⑦ → ⑤ → ⑥ → ③

3.3.4 DoS Defense

Finally, a temporary ID for defending against DoS attacks is proposed. First, the security gateway monitors frames in the CAN bus and detects a DoS attack. A DoS attack is detected using the existing method that analyzes the frequency of messages. At this time, messages that have high priorities are monitored. If a monitored message are transferred faster than defined

frequency, it may be a DoS attack. To defend against such DoS attack, the management component in the security gateway creates a seed for a temporary ID. Then, the temporary ID used in each ECU is created using the seed in the management component before it is sent and examined to check whether they have same ID. If the ID is same, the seed is created again to repeat the process. In addition, once the temporary ID used in each ECU is created, the table that corresponds to the existing ID will be created for a smooth communication with the other domains.

The created seed is transferred to each ECU. The ECUs that receive messages from the security gateway make a temporary ID using the transferred seed. At this time, the pre-shared derivation function and SipHash are used to create a temporary ID. First, the median value that is used in the SipHash input is made using the seed. Eq(1) is an equation of derived value for determining the median value.

$$\frac{ID + seed}{n} \tag{1}$$

n represents the total number of ECUs in a domain and *seed* is value from security gateway. *ID* is ECU's ID. Because an attacker does not have information about inside of vehicle, the attacker does not know n. Thus, this equation is appropriate. Next, below is the final temporary ID.

$$Lowest\ 11\ bit\ of\ SipHash(\frac{ID_1 + seed}{n})\ mod\ ID_h \tag{2}$$

Eq(2) is used to create temporary ID. ID_1 is ECU's ID and *seed* is value from security gateway and ID_h is ID that has highest priority in domain. The median value is inserted into SipHash as input. Then, a 64-bit tag value is obtained and the 11 bit that is the lowest bit in the 64-bit tag value is extracted. A modular operation is performed with this value and the highest ID. Through the modular operation, the value that is lower than the highest ID will be obtained, and it means that this value has a higher priority than the highest priority ID in the CAN bus. Therefore, if ECUs use the replaced temporary ID instead of the original ID, there is no communication problem because the temporary ID has a higher priority than the attacker's ID. This enables ECUs to communicate with each other regardless of any DoS attack. At this time, the attacker stops his DoS attack because he knows the attack is having no effect on the CAN bus. When the security gateway detects this event, the security gateway sends messages to notify other components that the DoS attack has ended.

4 Implementation

Our method is implemented using OMNET++, which is an open source program that can test networks and is used for testing various network environments such as Ethernet, wireless, mobile, and p2p. Because OMNeT++ is based on C++, anyone who uses C++ can easily organize his/her own experimental environment by modifying and testing code in OMNeT++.

4.1 Our Experiment

Although OMNeT++ offers various basic models for the network simulation, CAN models are not offered in OMNeT++. Therefore, Keigo *et al.* [24] developed a CAN model based on the basic model in OMNeT++.

However, the CAN basic model that is offered by [24] cannot simultaneously send and receive messages, because sender and receiver are separated. Thus, the model does not reflect actual behavior of ECUs that can send and receive messages simultaneously. In addition, because there is only the data frame, additional implementation of frames in CAN is required to test our method fully. We modify the CAN basic model. Furthermore, the security gateway proposed in this paper is added to this CAN basic model. Finally, some minor parts were added or modified.

Next, the basic topology is implemented based on the structure of the vehicle. Because the structure of the vehicle consists of three parts, engine part, break part, and assistance part, a basic topology also consists of three parts. Each part is made up of ECUs in Table 1, which is presented in Section 3. Figure 4 shows the basic topology, and each of the three parts is made of the ECUs mentioned in Table 1. In addition, there are three security gateways and each security gateway manages its domain.

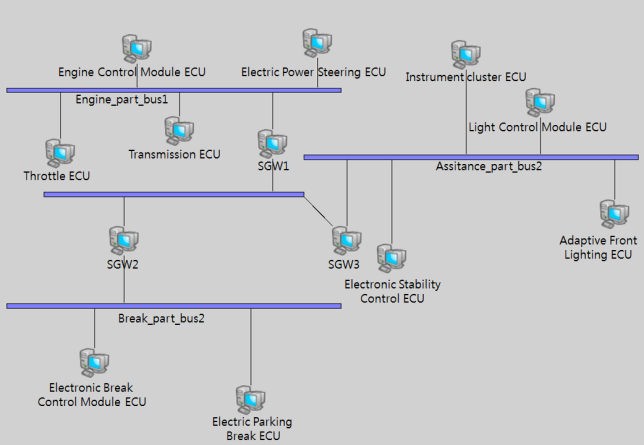

Figure 4: Basic topology

Below is ID of basic ECUs in each part that are used in experiment.

1) Engine part

- Throttle ECU: 0x61
- Engine Control Module ECU: 0x31
- Electric Power Steering ECU: 0x51
- Transmission ECU: 0x21

2) Break part

- Electronic Break Control Module ECU: 0x41
- Electric Parking Break ECU: 0x81

3) Assistance part

- Electronic Stability Control ECU: 0x11
- Light Control Module ECU: 0x91

- Adaptive Front Lighting ECU: 0x101
- Instrument Cluster ECU: 0x71

Additional ECUs are added to the basic topology in each experiment. 20 and 30 additional ECUs are added to the first and second experiment, because the average ECUs in vehicle are 30 from 40. The attack ECU is in one of parts in the basic topology for the spoofing attack and the attack frame is transferred to the CAN bus at an arbitrary time.

5 Result

5.1 Result I: Spoofing Defense

The detection rate and the increase in traffic are measured in the spoofing attack. The detection rate is measured by changing the number of attack frames, when there were 30 or 40 ECUs. Table 4 shows the detection rate for 30 and 40 ECUs. Examining Table 4, the detection rate is approximately 99%, 98.8%, and 98.5% when there were 10, 20, and 30 attack frames. Although the detection rates are generally high, the detection rate becomes low as the number of ECUs and attack frames increases. The lower detection rate is a false positive that detects normal frames as attack frames. In the proposed spoofing defense method, there is a verification process and false positives occur in this process. There are two frames, one is data frame and another is verification frame. If data frame arrives before arriving verification frame, this frame may be considered as a verification frame. Thus, the verification process will be failed and a false positive will occur despite it being a valid frame. Although false positives occur, the average detection rate is about 98.8%, demonstrating a high detection rate.

Table 4: Detection rate

		Number of ECUs	
		30 ECUs (%)	**40 ECUs (%)**
Number of attack frames	**10**	99.3	99.0
	20	99.1	98.6
	30	98.7	98.3

Next, the increase in traffic is measured in the our method. Table 3 shows how much traffic increases when the total number of ECUs is 30. The number of attack frames is 30 in this experiment. The **Rate** in Table 3 represents increased frames. From the table, the increase range is from 0.08% to 0.41%, when there are 30 ECUs. It shows an average increase of 0.11%. This is low considering the rate of total frames. When there are 40 ECUs, the increase range is from 0.11% ot 0.37%. The average increase is 0.19%, so it is slightly increased. As the number of ECUs is increased, the total number of frames is similarly increased. Therefore, the number of frames that requires the verification is relatively increased. Therefore, it is determined that the traffic increased slightly. However, the average increase is 0.11%, which is a relatively small portion of total frames.

5.2 Result II: DoS Defense

The result of the defense against DoS attacks is presented in this section. The attack ECU tries to attack the CAN bus at an arbitrary time. The DoS attack is maintained during regular time and then stops. This is one period of DoS attacks, and it occurs randomly. After this manner, DoS attack continuously happens and stops.

Figure 5 shows the frame drop rate at the ECU. Because the priority of the attack frames is higher than that of other frames during a DoS attack, other frames cannot be transferred. Therefore, the ECU's frame drop rate is very large. Figure 5 shows this situation. Initially, we can see that the frame drop rate gradually becomes large in Figure 5, because the ECU cannot send frames to the CAN bus. After that, a temporary ID is created through our method. The communication between ECUs becomes possible, because the temporary ID had a higher priority than the one used in the DoS attack. This means that the frame drop rate become lows, as shown in Figure 5. After that, the DoS attack stops and the original ECU ID is recovered. Therefore, ECU communication is possible and there is only a small frame drop rate. Finally, once the DoS attack starts again, defense against DoS attacks is initiated again. As a result, the frame drop rate increased and decreased in the similar pattern.

When DoS attack is tried at another time, the defense against DoS attack is well performed. There are two DoS attacks, and each attack triggers defenses. As in Figure 5, the defense against DoS attack seems effective.

Table 3: Increase of traffic (30 ECUs)

Driver's behavior	**Number of total frames (A)**	**Number of valid frames (B)**	**Rate (B/A * 100 (%))**
1) Usual	4,060	14	**0.34**
2) Left/Right turn, U-tern	13,522	20	**0.41**
3) Ignition	20,143	19	**0.09**
4) Stop	15,423	16	**0.10**
5) Acceleration	13,696	19	**0.13**
6) Deacceleration	11,776	13	**0.11**
7) Parking/stop	27,748	24	**0.08**
8) Light	5,611	17	**0.30**
9) Backward movement	22,949	18	**0.10**
10) Change of line	20,424	21	**0.10**

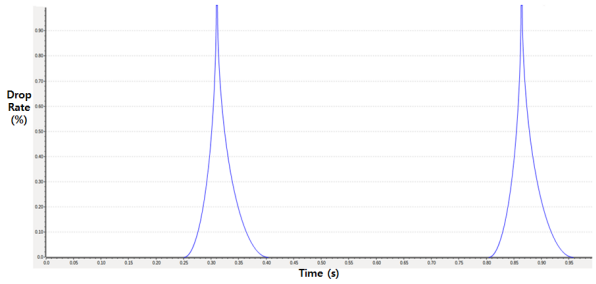

Figure 5: Frame drop rate

6 Conclusion and Future work

In this paper, the defense against spoofing and DoS attacks through a security gateway is proposed. In the case of the defense against a spoofing attack, the sequence of messages, stored in a table, is determined based on the driver's behavior. Whether a spoofing attack occurs in the CAN bus is monitored using this table. In addition, via a verification process, valid frames are verified and malicious frames can be detected. In the case of a DoS attack, a temporary ID is created based on a seed. At this time, SipHash is used for calculating a temporary ID that has higher priority than an attack frame's priority. This enables ECUs to smoothly communicate with each other regardless of any DoS attack.

Through OMNeT++, our experiments to defend against spoofing and DoS attacks are performed. Based on the basic topology, the detection rate in defending against spoofing attack is approximately 98.8%, and the traffic increase is at most 0.19%. This demonstrates that the proposed method is efficient. In defense against a DoS attack, the proposed method demonstrates that it could effectively defend against DoS attack by analyzing the frame drop rate.

The comparison is summarized in Table 5. From Table 5, our method shows low increase in traffic and can simultaneously defend spoofing and DoS attacks unlike other methods.

Some future work remains. First, the detection rate should be enhanced. It may be possible to extend the Table 2 that is used to defend against spoofing attacks efficiently by adding new driver's behaviors. Second, the verification process must be improved, to reduce the number of false positives. This can be solved by adding another verification process. Lastly, reducing the defense time during the initial period of a DoS attack is necessary. Our method takes time to begin defending against a DoS attack, causing the frame drop rate to reach approximately 100%. This can be reduced by applying a new algorithm that can detect a DoS attack more rapidly or by improving the process of generating a temporary ID.

Acknowledgements

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. NRF-2015R1A2A2A01006812).

References

[1] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, and S. Savage, "Experimental Security Analysis of A Modern Automobile," In Security and Privacy (SP), 2010 IEEE Symposium on. IEEE, 2010.

[2] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, and T. Kohno, "Comprehensive Experimental Analyses of Automotive Attack Surfaces," In USENIX Security Symposium. 2011.

[3] M. Nakano, T. Matsumoto, C. Vuillaume, and M. Kotani, "*Automotive information security: Threats and Countermeasures of ECU · automotive LAN · outside network*," Nikkei BP, Tokyo, *(in Japanese)*

[4] I. Studnia, V. Nicomette, E. Alata, Y. Deswarte, M. Kaâniche, and Y. Laarouchi, (2013, September). "Security of Embedded Automotive Networks: State of The Art and A Research Proposal," In SAFECOMP 2013 Workshop CARS (2nd Workshop on Critical Automotive applications: Robustness & Safety) of the 32nd International Conference on Computer Safety, Reliability and Security.

[5] K.J. Kim and D.S Lee, "A Trend Study on Vehicle Security in Conuntries using escar Conference," *Journal of The Korea Institute of Information Security and Cryptology,* Vol. 24, No. 2, pp.7 – 20, 2014. *(in Korean)*

[6] C. W. Lin, and A. Sangiovanni-Vincentelli. "Cyber Security for The Controller Area Network (CAN) Communication Protocol," Cyber Security (CyberSecurity), 2012 International Conference on. IEEE, 2012.

Table 5: Comparison existing methods with our method

	Increase of traffic volume	**Detection rate**	**Overhead**	**Modification of CAN protocol**	**Key management**	**Defending spoofing & DoS attacks**	**P.O.C (Proof of Concept)**
Proposed method	**≃ 0.11%**	**≃ 98.8%**	**Relatevely low**	**X**	**O**	**O**	**O**
Ujiie *et al* [9]	≃ 0%	≃ 99%	Relatevely low	X	X	X	O
Matumoto *et al* [10]	None	None	None	X	X	△	X
Hartkopp *et al* [18]	None	None	None	O	O	X	X
Kubota *et al* [12]	≃ 50%	None	High	X	O	X	O

[7] B. Groza, S. Murvay, A. van Herrewege, and I. Verbauwhede, (2012). "Libra-can: a lightweight broadcast authentication protocol for controller area networks," In Cryptology and Network Security (pp. 185-200). Springer Berlin Heidelberg.

[8] J. Yajima, M. Takenake, and T. Hasebe, "White-List CAN Hub with Disable Functionality against Attack Message," 2015 Symposium on Cryptography and Information Security (SCIS 2015), 2015, *(in Japanese)*

[9] Y. Ujiie, T. Kishikawa, T. Haga, and H. Matsushima, "Proposal of CAN Filtering Technology for In-Vehicle Network," 2015 Symposium on Cryptography and Information Security (SCIS 2015), 2015. *(in Japanese)*

[10] T. Matsumoto, M. Hata, M. Tanabe, K. Yoshioka, and K. Oishi "A Method of Preventing Unauthorized Data Transmission in Controller Area Network," Vehicular Technology Conference (VTC Spring), 2012 IEEE 75th. IEEE, 2012.

[11] O. Satoshi, and I. Tasuku, "Intrusion Detection for In-vehicle Networks without Modifying Legacy ECUs," IPSJ SIG Technical Report 2013. *(in Japanese)*

[12] T. Kubota, M. Nakano, R. Kurachi, S. Honda, M. Shiozaki, and T. Fujuno, "The Demonstration System of Encrypted CAN Communication and The Side-channel Attack Evaluation," 2015 Symposium on Cryptography and Information Security (SCIS 2015), 2015. *(in Japanese)*

[13] T. Kishikawa, Y. Ujiie, T. Haga, H. Matsushima, M. Tanabe, Y. Kitamuram and J. Anzai, "Proposal of Security CAN to Protect In-Vehicle Network: Updatable CAN Protection Method using HW/SW Cooperation and Evaluation of the Method," 2015 Symposium on Cryptography and Information Security (SCIS 2015), 2015. *(in Japanese)*

[14] T. Haga, Y. Ujiie, T. Kishikawa, H. Matsushima, M. Tanabe, Y. Kitamura, and J. Anzai, "Proposal of Security ECU to Protect In-Vehicle Network: Concept of CAN Protection Method that Suppresses the Introduction Impact," 2015 Symposium on Cryptography and Information Security (SCIS 2015), 2015. *(in Japanese)*

[15] M. Tanabe, Y. Kitamura, J. Anzai, T. Kishikawa, Y. Ujiie, T. Haga, and H. Matsushima, "A Secure Switching Method between Monitoring Mode and Verifying Mode for In-Vehicle Network," 2015 Symposium on Cryptography and Information Security (SCIS 2015), 2015. *(in Japanese)*

[16] N. Morita, K. Hakuta, T. Owada, and M. Kayashima, "A Proposal of a Message Authentication Method for Automotive Networks," 2015 Symposium on Cryptography and Information Security (SCIS 2015), 2015. *(in Japanese)*

[17] R. Kurachi, H. Takada, H. Ueda, and S. Horihata, "The Proposal of The Monitoring System for In-vehicle Networks," 2015 Symposium on Cryptography and Information Security (SCIS 2015), 2015. *(in Japanese)*

[18] O. Hartkopp, C. Reuber, and R. Schilling, (2012, November). "MaCAN-message authenticated CAN," In 10th Int. Conf. on Embedded Security in Cars (ESCAR 2012), 2012.

[19] P. Taylor, S. Anand, N. Griffiths, F. Adamu-Fika, A. Dunoyer, and T. Popham, "Road Type Classification through Data Mining." Proceedings of the 4th International Conference on Automotive User Interfaces and Interactive Vehicular Applications. ACM, 2012.

[20] H. Qichang, W. Li, and X. Fan. "Estimation of Driver's Fatigue based on Steering Wheel Angle." Engineering Psychology and Cognitive Ergonomics. Springer Berlin Heidelberg, 2011.

[21] J.C. McCall, and M. M. Trivedi. "Human Behavior based Predictive Brake Assistance." Intelligent Vehicles Symposium, 2006 IEEE. IEEE, 2006.

[22] B.I. Kwak, M.R. Han, A.R. Kang, H.K. Kim, "A Study on Detection Methodology of Threat on Cars from The Viewpoint of IoT," *Journal of The Korea Institute of Information Security and Cryptology,* Vol. 25, No. 2, pp.411 – 421, 2015. *(in Korean)*

[23] A. Jean-Philippe, and D. J. Bernstein. "SipHash: a fast short-input PRF," Progress in Cryptology-INDOCRYPT 2012 (pp. 489-508). Springer Berlin Heidelberg, 2012.

[24] K. Keigo, Y. Matsubara, and H. Takada. "A Simulation Environment and preliminary evaluation for Automotive CAN-Ethernet AVB Networks," arXiv preprint arXiv:1409.0998, 2014.