# A Proposal of the Device Disabler for Controller Area Network

Hiroshi UEDA*, Ryo KURACHI*, Shinya HONDA, Hiroaki TAKADA, Naoki ADACHI, and Yukihiro MIYASHITA

Recently, quite a number of security attacks against Controller Area Network (CAN) have been reported. Many automotive companies are planning to adopt security countermeasures to strengthen security of their in-vehicle systems while saving the costs. This paper proposes a method to block unauthorized CAN-bus access using our enhanced CAN controller that prevents the transmission of messages from a malicious electronic control unit. We demonstrate the effectiveness of our device disabler on a CAN with Flexible Date rate buses.

Keywords: in-vehicle network, security, Controller Area Network (CAN), CAN with Flexible Date rate (CAN FD)

## 1. Introduction

Each vehicle manufactured today is equipped with many small computers called electronic control units (ECUs).[1] These ECUs exchange information via an in-vehicle network to achieve vehicle control. Controller Area Network (CAN)[2] is a communication protocol that is widely used in these electronic control systems. CAN with Flexible Data rate (CAN FD), which is a new protocol, is also expected to come into widespread use. Recently, many cyberattacks via in-vehicle networks have been reported.

Koscher[3] et al. demonstrated that it is possible to take over vehicle control by transmitting forged CAN messages to the in-vehicle network. Valasek[4] et al. showed that unauthorized transmission is easily enabled by arranging an unauthorized device on a CAN bus. A recent study by Miller[5] et al. indicated that the program of an ECU connected to a CAN bus can be rewritten via the mobile phone network to achieve unauthorized transmission. Many attacks to rewrite ECU programs to transmit unauthorized CAN messages and take over vehicle control have been reported.

This paper proposes a hardware-based device disabler to prevent unauthorized transmission of CAN messages as a measure to cope with these attacks. Specifically, we propose a method of preventing transmission of unauthorized CAN messages from an ECU, whose program has been rewritten or into which malware*1 has been injected, by implementing this device disabler in an improved CAN controller.

## 2. Features of CAN

CAN is a communication protocol standardized by ISO 11898, and has the following features.

(a) Bus topology

CAN is widely used in a bus topology in which two or more nodes are connected to a single communications line.

(b) Arbitration of transmission right

Each node uses a multi-master system that immediately switches to transmission operation when there are messages to be transmitted. When two or more nodes transmit messages on a CAN bus at the same time, conflicts of messages occur. To resolve these conflicts, the transmission right is arbitrated based on CAN-ID. Based on the arbitration, CAN messages with the highest priority are transmitted preferentially. Meanwhile, the commencement of transmission of messages with lower priority is delayed until transmission of all the messages with higher priority is completed.

(c) Mailbox

In a CAN communication controller, a register group for transmitting and receiving messages is referred to as a mailbox. Most general CAN controllers have two or more transmission mailboxes and reception mailboxes to ensure application to various systems. Each node uses two or more mailboxes for transmission or reception prepared by the CAN controller, but it does not necessarily use all the mailboxes.

## 3. Proposed Method: The Device Disabler

This chapter explains the proposed method to prevent unauthorized transmission ("the device disabler") for CAN and illustrates the effectiveness of this solution against existing attacks.

### 3-1 Details of the device disabler

A possible attack threat is an unauthorized rewriting of the ECU program through an external network by the same means used in a reference document.[5] This proposed solution restricts the use of a CAN bus by an ECU whose program is rewritten ("stepping stone") and minimizes attacks via stepping stones on the in-vehicle control system. The protection functions offered by this device disabler are as follows.

**(a) Protection function 1: restriction of the use of unused mailboxes**

The design restricts the mailboxes used by each ECU for transmission and reception. Even if an ECU becomes a stepping stone, abuse of the unused mailboxes can be prevented.

**(b) Protection function 2: restriction of transmittable messages based on a whitelist***2

CAN messages that should be transmitted by each ECU are determined at the time of design. CAN messages that can be transmitted by each ECU are restricted in advance. Even if an ECU becomes a stepping stone, transmission of messages other than those determined at the time of design can be prevented.

**(c) Protection function 3: restriction of transmission at an abnormal transmission frequency**

The transmission frequency of CAN messages that should be transmitted by each ECU is determined at the time of design. Even if an ECU becomes a stepping stone, the device disabler prevents transmission of CAN messages beyond the predetermined transmission frequency.

Since these protection functions are not implemented in current ECUs, unauthorized rewriting of a program or injection of malware readily enables unauthorized transmission from a stepping stone. The specific threats are as follows.

If Protection function 1 is not implemented, malware masquerades as the legitimate application of the stepping stone and uses unused mailboxes to transmit unauthorized CAN messages. If Protection function 2 is not implemented, malware masquerades as a legitimate ECU other than the stepping stone and transmits unauthorized CAN messages. If Protection function 3 is not implemented, the stepping stone transmits messages at an abnormal frequency, enabling a denial-of-service (DoS) attack*3 on the CAN bus.

### 3-2 Method of achieving the proposed solution

The proposed solution is achieved by expanding the hardware of the existing CAN controller. The device disabler is a hardware that is implemented to determine whether a CAN message is transmittable when transmission of a message is requested. The necessary information about transmittable CAN messages (e.g., CAN-ID, data length code) and their transmission cycles must be set as a whitelist in this device disabler in the CAN controller.

Inspections 1 to 3 below are then conducted based on the inspection procedure shown in Fig. 1 to determine whether a message whose transmission is requested can be transmitted.

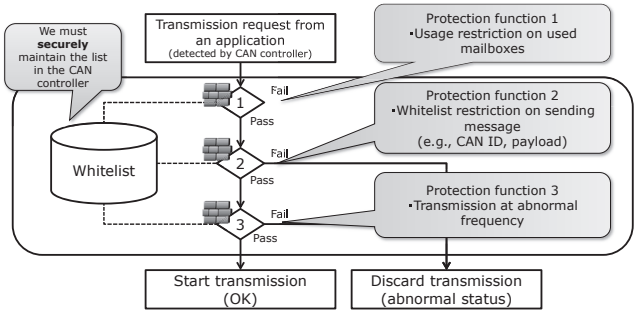



Fig. 1. Inspection procedure for processing a transmission request

**(a) Inspection 1:** The device disabler inspects the mailbox whether it is authorized to be used for transmission. If a mailbox that is not authorized to be used for transmission is intended to be used, the device disabler discards the transmission. Meanwhile it keeps the transmission operation only if the mailbox is an authorized one, and it performs Inspection 2.

**(b) Inspection 2:** The device disabler evaluates, based on the whitelist, whether the CAN message is authorized to be transmitted or not. If an unauthorized CAN message is to be transmitted, the device disabler discards the transmission. Meanwhile it keeps the transmission operation only if the message is an authorized one, and it performs Inspection 3.

**(c) Inspection 3:** The device disabler evaluates the minimum transmission cycle of the requested transmission. If the transmission frequency of the requested transmission is higher than the preset minimum transmission cycle, the device disabler permits the transmission. If the transmission frequency is less than the minimum transmission cycle, the device disabler discards the transmission.

### 3-3 Method of writing a whitelist

In this proposed solution, the device disabler is invalidated if the whitelist is tampered with. Thus, it is important to protect the whitelist. The whitelist should be written by either of the two procedures below. To write a whitelist on the runtime, the whitelist must be protected against rewriting after it is set. In the device disabler that is proposed in this paper, if Writing method 2 below is used, the setting of the whitelist must be disabled until the whitelist rewriting completion register is set after a reset is canceled.

**(a) Writing method 1:** Write the whitelist to a rewritable flash ROM from an authenticated diagnosis tool at the time of shipment. For some microcomputers that are commercially available at present, such means are used to set the clock for the microcomputer. A similar mechanism should be used.

**(b) Writing method 2:** Use Secure Boot to set the whitelist in the device disabler. In this case, Secure Boot must be executed using a secure element. The hardware cost may increase compared to that of existing ECUs.

Currently, not all ECUs execute Secure Boot. Thus, Writing method 1 is considered to be more compatible with existing in-vehicle control systems and to be more cost-effective.

### 3-4 Expected use cases

To use services such as metromile,[6] each vehicle must be equipped with communication devices for diagnosis called “OBD-II dongles.” These dongles are installed to discount the insurance premiums depending on the mileage. The information related to mileage transmitted on the CAN network is uploaded to the server of the service company. However, there have been attacks taking advantage of the vulnerability of these OBD-II dongles. Specifically, an unauthorized program is downloaded from the attacker’s server to transmit unauthorized messages on the CAN bus.

If the OBD-II dongles are equipped with the device disabler, transmission of unauthorized CAN messages can be prevented based on Inspection 2 mentioned above by setting the device disabler to disable transmission mailboxes on a CAN bus.

Regarding tampering with the ECU program as reported by Miller[5] et al., a stepping stone whose program

is rewritten can transmit messages that should be transmitted, but spoofing of messages transmitted by other ECUs can be prevented. Based on these results, it is highly useful to prevent transmission of unauthorized messages with hardware by using the device disabler.

## 4. Implementation

We implemented the device disabler on a field-programmable gate array (FPGA) board manufactured by Altera Corporation (Photo 1).

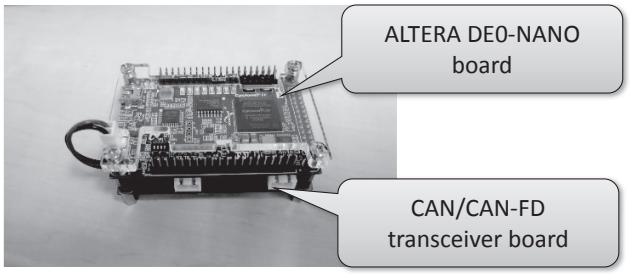



Photo 1. Real device evaluation environment

### 4-1 Implementation of a minimum transmission cyclic counter

We implemented a minimum transmission cyclic counter as a counter of 1 bit-time unit on the CAN network generated on the CAN controller. This counter starts to count up after the initial transmission request is made. This is followed by the commencement of monitoring. The minimum transmission cyclic counter does not provide protection when the initial transmission request is made. When the second or subsequent transmission request is made, transmission on a CAN bus is disabled and the transmission request is discarded on the CAN controller if the frequency does not match the minimum transmission cycle setting.

### 4-2 Implementation of the device disabler

To implement the device disabler, we embedded it as a submodule in the exis ting CAN controller IP, as shown in Fig. 2. The overall system consisted of a Nios II softcore, a modified CAN controller, a DRAM controller, and an on-chip RAM. The synthesis results of this system consumed 23,888 logic elements. The CAN controller used 5,374 logic elements. The number of logic elements required was 3,106 more than that required for synthesis using an existing CAN controller IP.

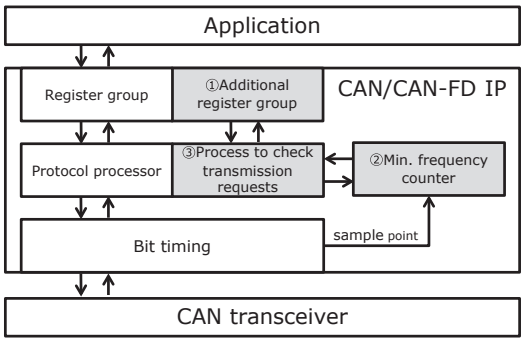



Fig. 2. Implementation of the device disabler

## 5. Evaluation

### 5-1 Evaluation method

The evaluation was conducted for the two aspects as follows. First, we evaluated the overhead attributed to the device disabler. Although the device disabler is implemented as hardware, the addition of the inspection process increases the stand-by time from the transmission request process to the commencement of transmission compared to the existing CAN controller. Thus, we verified that this increase in time is within the permissible scope.

Second, we evaluated the effectiveness of the device disabler in the event of an unauthorized rewriting of a program, in order to demonstrate the effectiveness of this proposed solution. We demonstrated that the device disabler works effectively in the event of program rewriting.

### 5-2 Evaluation of the processing overhead

The CAN controller IP equipped with the device disabler incurs an overhead compared to the existing CAN controller IP due to the time required for filtering at the time of transmission request. Thus, we used a hardware counter to measure the overhead process. After a transmission request is detected, the time required for the inspection process in Fig. 1 is 6.25 μsec at the maximum. We verified that the delay time is about 3 bits-time or 4 bits-time at a CAN transfer speed of 500 kbps. Based on this result, we verified that the time required for the inspection process is sufficiently small compared to the transmission cycle of CAN messages, despite an overhead compared to the existing CAN controller IP.

### 5-3 Evaluation of the device disabler

(1) Prevention of transmission of unauthorized CAN messages by a stepping stone

When a CAN controller with the device disabler embedded is used, only the CAN messages on the CAN-ID registered in the whitelist are transmitted even if the ECU program is rewritten. To verify this functionality, we created a program to request transmission of arbitrary CAN-IDs from the application and verified that only CAN messages registered in the whitelist are transmitted (Fig. 3 and Fig. 4).

(2) Prevention of transmission by the stepping stone to the CAN network

The device disabler is designed to maintain the minimum transmission frequency in the whitelist. This prevents DoS attacks from a stepping stone. In a specific example, when the minimum transmission cycle of a CAN message is set to 5 msec, only one CAN message can be transmitted during the 5 msec period. However, if there are two or more messages transmitted from this ECU, the function depends on the number of messages transmitted and the set value of the minimum transmission cycles. Thus, it

is not necessarily effective as a measure against DoS attacks. Caution is required when setting the minimum transmission cycle.

In this evaluation, we created an application to request transmission at an interval of about 10 μsec and set a whitelist. If the device disabler is not provided on a CAN network, transmission is repeated constantly depending on the frequency of the transmission request and transfer time interval on a CAN bus, enabling DoS attacks. The device disabler provides an interval of the minimum transmission cycle time. We verified that the device disabler is effective against DoS attacks (Fig. 5).

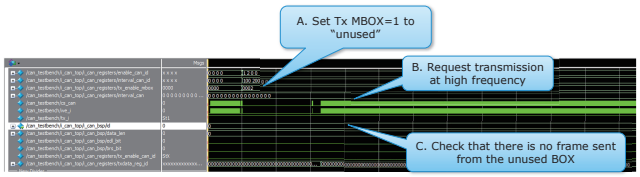



Fig. 3. Verification of Protection function 1

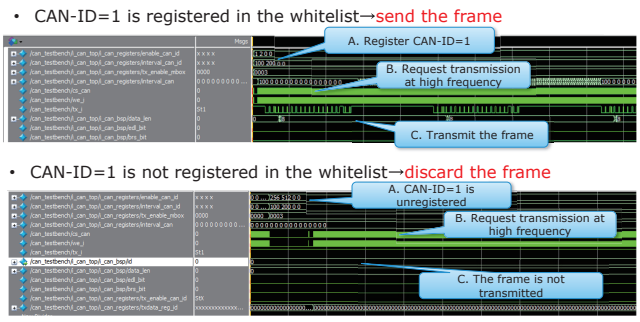



Fig. 4. Verification of Protection function 2

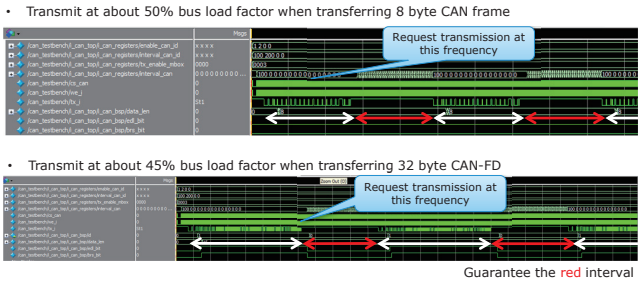



Fig. 5. Verification of Protection function 3

## 6. Discussion

### 6-1 Effective scope of the device disabler

The proposed device disabler cannot prevent tampering of the program of a legitimate ECU. However, it can prevent transmission of spoofing messages by a stepping stone under certain conditions. Thus, the best possible solution is to implement Secure Boot and a CAN controller equipped with the device disabler. To implement Secure Boot, secure elements must be implemented in existing ECUs. Implementation of Secure Boot in all the ECUs of an in-vehicle control system could increase the cost and incur an overhead in the starting time. As shown in Table 1, ECUs that are particularly related to control or used for external connection should be equipped with both functions. To ensure safety, either Secure Boot or the device disabler should be selected and applied to part of the control and body systems.

Table 1. Effective range of the device disabler

| ECU Lv | Secure boot | The device disabler | Application example |
|---|---|---|---|
| Level 1 | Required | Required | Gateway, Engine ECU, HU |
| Level 2 | Not Required | Required | Part of the control and body systems |
| Level 3 | Not Required | Not Required | Part of subsystems |

### 6-2 Comparison with conventional studies

There have been researches on firewalls and intrusion detection systems (IDSs) that filter communications from outside in the past. Otsuka[7] et al. proposed a system that can be easily implemented by software as an IDS for in-vehicle networks. This method uses a software application to monitor the arrival interval of each message in an in-vehicle system by focusing on the fact that each CAN message is transmitted cyclically. Meanwhile, Muter[8] et al. proposed an entropy-based IDS system. However, ECUs that make up an in-vehicle control system are equipped with low-speed but cost-effective CPUs. Thus, it is considered extremely difficult to achieve advanced calculations. Miller[4] et al. also discussed a method of monitoring the transmission frequency.

Sekiguchi[9] et al. proposed the operation of hubs using whitelists, but did not specify the method of updating the whitelists and hubs. The proposed solution targets the hub configuration, and is not designed for ECUs that have one port for CAN communication. Ujiie[10] et al. proposed software-based filtering of the incoming communication (firewall function) by setting a static filter. They mentioned Secure Boot but did not describe the effectiveness for in-vehicle infotainment systems that are likely to be infected with malware.

Herber[11],[12] et al. proposed a time division method for access to a CAN bus to minimize the impact of DoS attacks. It should be noted that this method can localize the impact of DoS attacks caused by infection with malware, but unauthorized transmission of CAN messages was not mentioned. For these reasons, our method is considered to be effective for head units (HUs) and navigation systems equipped with two or more applications.

## 7. Conclusion

This paper proposes a hardware-based device disabler to prevent unauthorized transmission to protect the CAN network when an ECU program is tampered with or rewritten by an unauthorized program. This device disabler

is achieved by expanding the hardware of the CAN controller on an ECU that is used for in-vehicle control systems. We implemented the device disabler on an FPGA and conducted an evaluation. We verified that the device disabler is effective against existing attacks. We also demonstrated that the device disabler can minimize the impact of a stepping stone on the CAN network.



**Technical Terms**

∗1 Malware: Malware is a generic term for malicious code that is created for unauthorized and malicious operation. Malware includes computer viruses and worms.

∗2 Whitelist: A whitelist refers to a list of items that do not require special caution or alert among lists that indicate whether caution or alert is required.

∗3 DoS attack: A DoS attack refers to an attack to disrupt services by applying an excessive communication load intentionally.

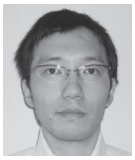
**References**

(1) J. Leohold, Communication Requirements for Automotive Systems, 5th IEEE Workshop Factory Communication Systems (2004)
(2) International Organization for Standardization, Road vehicles - Controller area network (CAN) - Part 1: Data link layer and physical signaling, ISO11898-1 (2003)
(3) K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, Experimental Security Analysis of a Modern Automobile, IEEE Symposium on Security and Privacy (2010)
(4) C. Valasek, C. Miller, "Adventures in Automotive Networks and Control Unit," http://www.ioactive.com/pdfs/IOActive_Adventures_in_Automotive_Networks_and_Control_Units.pdf (2014)
(5) C. Miller, C. Valasek, "Remote Exploitation of an Unaltered Passenger Vehicle," http://illmatics.com/Remote\%20Car\%20Hacking.pdf (2015)
(6) Metromile, https://www.metromile.com/insurance/ (2015)
(7) S. Otsuka, T. Ishigooka, Y. Oishi, and K. Sasazawa, "CAN Security: Cost-Effective Intrusion Detection for Real-Time Control Systems," SAE Technical Paper 2014-01-0340, 2014, doi: 10.4271/2014-01-0340
(8) M. Muter and N. Asaj, "Entropy-based anomaly detection for in-vehicle networks," In Intelligent Vehicles Symposium (IV), Baden Baden, Germany (2011)
(9) D. Sekiguchi, M. Tanabe, K. Yoshioka, T. Matsumoto, "Preventing Unauthorized CAN Transmission by Surveillance Mechanism Built in Electronic Control Unit" (in Japanese), IEICE Technical Report (2013)
(10) Y. Ujiie, T. Kishikawa, T. Haga, H. Matsushima, T. Wakabayashi, M. Tanabe, Y. Kitamura, J. Anzai, "A Method for Disabling Malicious CAN Messages by Using a Centralized Monitoring and Interceptor ECU," Embedded Security in Cars 2015
(11) C. Herber, A. Richter, H. Rauchfuss, and A. Herkersdorf, "Spatial and Temporal Isolation of Virtual CAN Controllers," In Workshop on Virtualization for Real-Time Embedded Systems (VtRES 2013), pp. 7-13 (2013)
(12) C. Herber, D. Reinhardt, A. Richter, and A. Herkersdorf, "HW/SW Trade-offs in I/O Virtualization for Controller Area Network," presentation at DAC'15 (June 2015)

**Contributors** The lead author is indicated by an asterisk (*).

**H. UEDA***
• Manager, AutoNetworks Technologies, Ltd.

**R. KURACHI***
• Ph.D.
Designated Associate Professor, Nagoya University



**S. HONDA**
• Ph.D.
Associate Professor, Nagoya University

**H. TAKADA**
• Ph.D.
Executive Director, Professor, Nagoya University

**N. ADACHI**
• AutoNetworks Technologies, Ltd.

**Y. MIYASHITA**
• Senior Manager, AutoNetworks Technologies, Ltd.