A Survey of Attacks on Controller Area Networks and Corresponding Countermeasures

Hyo Jin Jo and Wonsuk Choi

***Abstract*— The development of vehicle technologies such as connected and autonomous vehicle environments provide drivers with functions for convenience and safety that are highly capable of remote vehicle diagnosis or lane-keeping assistance. Unfortunately, despite impressive advantages for drivers, these functions also have various vulnerabilities that could lead to cyber-physical attacks on automotive Controller Area Networks (i.e., automotive CAN). To deal with these security issues, a multitude of issue-specific countermeasures have already been proposed. In this paper, we introduce existing research on automotive CAN attacks and evaluate several state-of-the-art countermeasures. Particularly, we provide a comprehensive adversary model for automotive CAN and classify existing countermeasures into four system categories: (1) preventative protection, (2) intrusion detection, (3) authentication, and (4) post-protection. From the extensive literature review, we attempt to summarize the security research regarding automotive CAN and identify open research directions for in-vehicle networks of autonomous vehicle.**

***Index Terms*— Controller area network, vulnerabilities, intrusion detection system, authentication.**

I. Introduction

IT does not seem long ago when groundbreaking safety functions like the anti-lock braking system (ABS) or the lane keeping assist system (LKAS) were introduced to the modern automobile. These and other convenience-adding functions, such as cruise control or auto-parking systems, have now become integral parts of the contemporary driving experience. Of course, these components all require massive support inside the vehicle to function properly. To this end, electronic control units (ECUs) have been designed to control all components in modern vehicles, from ventilation and window control systems to the engine control system. For communication among ECUs, the Controller Area Network (CAN) as a de-facto standard has been widely used. In addition, this internal network (i.e., CAN) can be connected to external networks through a connected and/or autonomous vehicle technology. For example, an adaptive cruise control (ACC) system could be turned into a cooperative ACC (CACC) system by adding vehicle-to-vehicle (V2V) communications [1].

Manuscript received 4 October 2019; revised 5 August 2020, 10 January 2021, and 16 March 2021; accepted 17 April 2021. Date of publication 24 May 2021; date of current version 8 July 2022. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIT) under Grant 2018R1C1B5086261 and Grant NRF-2020R1C1C1007446. The Associate Editor for this article was S. Olariu. *(Corresponding author: Wonsuk Choi.)*

Hyo Jin Jo is with the Department of Software, Soongsil University, Seoul 06978, South Korea (e-mail: hyojin.jo@ssu.ac.kr).

Wonsuk Choi is with the Division of IT Convergence Engineering, Hansung University, Seoul 02876, South Korea (e-mail: wonsuk@hansung.ac.kr).

Digital Object Identifier 10.1109/TITS.2021.3078740

Despite the significant advances these emerging functions have achieved in improving both efficiency and safety of vehicle controls, they have also ushered in a range of new and complicated security threats, such as vehicle hacking [2]. Once an adversary accesses the automotive CAN of a target vehicle, they can control the vehicle by injecting malicious packets (e.g., engine shut-down) to automotive CAN as explained in [2]–[15]. The results of these studies have demonstrated that vulnerabilities of modern vehicles are critical—and very real—issues that must be addressed with haste.

A basic method to protect automotive CAN from hacking is network isolation and/or firewalls. According to [16], 8 out of 14 (57%) of the 2014 model vehicles the authors examined have a separate control network that is isolated from external network interfaces. However, the creation of network isolation and firewalls such as [17]–[19] does not make the automotive CAN impenetrable by unauthorized agents. Indeed, some auto manufacturers have developed an open access port that can bridge between internal and external networks to collect CAN data from customer vehicles in order to provide telematics services like remote diagnosis or to support future automotive research in now-emerging fields such as autonomous vehicles [20].

Considering the grave ramifications of cyber attacks on automotive CAN, we have conducted a survey on state-of-the-art of attack surfaces and corresponding protection mechanisms as shown in Fig 1. During our research, the necessity of a comprehensive survey such as this came to our attention when we discovered that the major surveys on security for in-vehicle networks published in [21]–[29] offer insufficient information about the classification of security mechanisms for automotive CAN as shown in Table I. Although these descriptive surveys are considerably noteworthy in value, we noted that there lacks a comprehensive view of CAN attacks and corresponding countermeasures, which has become the basis of this present research. In short, the goal of this paper is to not only provide a better understanding of current security-related research and challenges of next-generation in-vehicle networks (e.g., automotive Ethernet) for readers in different areas of expertise but also act as a comprehensive practical guide for constructing a secure automotive CAN by describing CAN attack surfaces and applicable corresponding protection mechanisms.

The contributions of this paper are as follows.

- Classification of CAN attack surfaces: By analyzing attack methods presented in [2]–[15], CAN attack surfaces can be classified into three parts: physical access-based attacks, wireless access-based attacks requiring

Attacks on Automotive CAN	Countermeasures					
	Preventative Protection Methods		Intrusion Detection	Message Authentication	Post-Protection Methods	
	Fuzzing	Anti-analysis			Attack Identification	Secure Patch
Koscher et al. [2] (2010) Valasek et al. [3] (2013) Checkoway et al. [4] (2011) Eric et al. [5] (2015) Woo et al. [6] (2015) Lee et al. [7] (2019) Mandal et al. [8] (2019) Wen et al. [9] (2020) Haohuang Wen et al. [10] (2020) Foster et al. [11] (2015) Jo et al. [12] (2016) Mazloom et al. [13] (2016) Nie et al. [14] (2017) Miller et al. [15] (2015)	Bayer et al. [30] (2015) Oka et al. [31] (2015) Fowler et al. [32] (2018)	Yu et al. [33] (2015) Hely et al. [34] (2007) Rosenfeld et al. [35] (2010) Novak et al. [36] (2006) Pierce et al. [37] (2013) Mundhenk et al. [38] (2017) Han et al. [39] (2015) Humayed et al. [40] (2017) Wu et al. [41] (2018) Woo et al. [42] (2019) Cheng et al. [43] (2020) Groza et al. [44] (2020)	Taylor et al. [45] (2015) Song et al. [46] (2016) Tomlinson et al. [47] (2018) Olufowobi et al. [48] (2020) Young et al. [49] (2019) Marchetti et al. [50] (2017) Katragadda et al. [51] (2020) Stabili et al. [52] (2017) Markovitz et al. [53] (2017) Kang et al. [54] (2016) Song et al. [55] (2020) Taylor et al. [56] (2016) Longari et al. [57] (2020) Hossain et al. [58] (2020) Tariq et al. [59] (2020) Wasicek et al. [60] (2017) Ansari et al. [61] (2017) Müter et al. [62] (2010) Müter et al. [63] (2011) Marchetii et al. [64] (2016) Cho et al. [65] (2016) Lee et al. [66] (2017) Murvay et al. [67] (2020) Murvay et al. [68] (2014) Choi et al. [69] [70] (2018a, 2018b) Kneib et al. [71] (2020)	Woo et al. [6] (2015) Wang et al. [72] (2014) Nürnberger et al. [73] (2016) Radu et al. [74] (2016) Kang et al. [75] (2017) Bella et al. [76] (2019) Mun et al. [77] (2020) Youn et al. [78] (2020) Palaniswamy et al. [79] (2020) Schmandt et al. [80] (2017) Groza et al. [81] (2013) Kurachi et al. [82] (2014) Wang et al. [83] (2017) Jo et al. [84] (2020)	Nilsson et al. [85] (2008) Cho et al. [86] (2017) Lee et al. [87] (2019) Van Bulck et al. [88] (2017)	Nilsson et al. [89] (2008) Mahmud et al. [90] (2005) Mansour et al. [91] (2012) Steger et al. [92] (2018) Petri et al. [93] (2016) Kuppusamy et al. [94] (2018)

Fig. 1. Attack and defense of automotive CAN.

initial physical access, and wireless access-based attacks without physical access. In addition, we define two types of adversaries: an adversary who can compromise an ECU and an adversary who can only access automotive CAN.

- Categorization of countermeasures against CAN attacks: Protection methods for automotive CAN [6], [30]–[94] fall into four system categories: preventative protection (fuzzing methods and anti-analysis methods), intrusion detection, authentication, and post-protection (attack identification methods and secure patch methods). We also analyze the advantages and disadvantages of the categorized defense techniques and evaluate whether or not these techniques are secure against the two types of adversaries we defined.

The rest of the paper is organized as follows: Section II gives background information on CAN. Existing attack results and an adversary model on automotive CAN are described in Section III. Sections IV, V, VI, and VII explain a defense system for secure automotive CAN. In Section VIII, we give suggestions on open security research directions for autonomous vehicle in-vehicle networks. Section IX presents our conclusion.

II. Background of Controller Area Networks

In this section, we provide a background on automotive CAN for better understating of automotive CAN security.

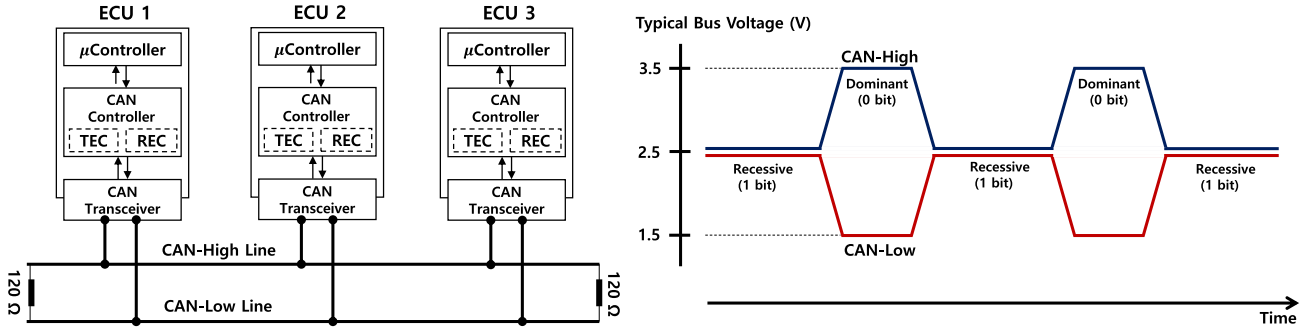
A. Controller Area Network and Electronic Control Unit

CAN is based on a bus topology (i.e., CAN bus) and provides a reliable communication channel for ECUs.

TABLE I

COMPARISON BETWEEN THE EXISTING SURVEY PAPERS AND OURS (O: DISCUSSED, -: NOT DISCUSSED)

	Attack Classification	Security mechanisms for automotive CAN			
		Preventative Protection	Intrusion Detection	Authentication	Post-Protection
Zhang et al. [21]	O	O	-	-	-
Luo et al. [22]	O	O	-	-	-
Loukas et al. [23]	O	-	O	-	-
Al-Jarrah et al. [24]	O	-	O	-	-
Wu et al. [25]	O	-	O	-	-
Young et al. [26]	O	-	O	-	-
Khan et al. [27]	O	-	O	-	-
Halder et al. [28]	O	-	-	-	O
Sun et al. [29]	O	-	-	-	-
Ours	O	O	O	O	O



(a) Three components of ECUs

(b) Voltage levels of the CAN physical layer

Fig. 2. Three components of ECUs and the CAN physical layer.

In general, modern vehicles have a few dozen ECUs that control various subsystems including the engine, brakes, and steering. A typical ECU is composed of three parts: a host processor, a CAN controller, and a CAN transceiver as shown in Fig. 2(a).

CAN transceivers are used to connect the CAN controller to the physical transmission medium (i.e., CAN bus), and these transceivers always have two CAN bus pins: CAN high line (CANH) and CAN low line (CANL). By using these two lines, a CAN transceiver can output certain voltage levels to send either a dominant bit (0 bit) or recessive bit (1 bit). As shown in Fig. 2(b), to transmit a dominant bit (0 bit) on the CAN bus, the CAN transceiver outputs approximately 3.5V on CANH and 1.5V on CANL so that the voltage difference is approximately 2.0 V. When the CAN transceiver transmits a recessive bit (1 bit), it outputs approximately 2.5V on both CANH and CANL, which creates a minor voltage difference. Thus, receiver ECUs can read CAN packets transmitted on CAN bus by measuring the differential voltage of CANH and CANL.

B. CAN Frames and Arbitration

The CAN standard defines four frame types: a data frame, a remote frame, an error frame, and an overload frame.[1] Every data frame includes an identifier called CAN ID, which is 11 bits (CAN specification 2.0 A) in length, as shown in Fig 3. In general, almost all ECUs transmit their own data frames periodically to share status or send a specific command to other ECUs.[2]

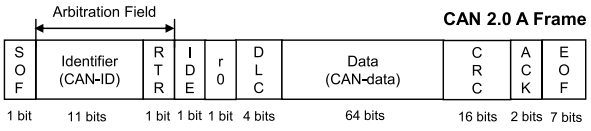
The CAN protocol as a synchronous protocol is based on a bit-time slot, which, for example, could be illustrated as a one bit-time slot equalling two microseconds when using CAN bus with 500 kbit/s. If more than two ECUs start to transmit remote frames or data frames on CAN bus at the same bit-time slot, the priority of access is determined by bit-wise arbitration using their CAN IDs. In arbitration, a dominant bit (0) is considered dominant over a recessive bit (1).

III. ATTACKS ON AUTOMOTIVE CAN

In order to perform a cyber attack on automotive CAN, it is necessary to locate the vulnerabilities of an access port. In this section, we introduce well-known attack surfaces of automotive CAN, corresponding adversary models, and attack scenarios. Fig. 4 shows three access points of automotive CAN: OBD-II-based access, audio device-based access, and telematics device-based access.

[1]Remote frames, error frames, and overload frames are not discussed in detail in this paper

[2]In this paper, a data frame will be interchangeably denoted as a CAN packet or CAN message if there is no specific description for other frames.



SOF: Start of frame, RTR: Remote transmission request, IDE: Identifier extension bit, r0: Reserved bits, DLC: Data Length Code, CRC: Cyclic Redundancy Check, EOF: End of frame

Fig. 3. The formats of CAN data frames.

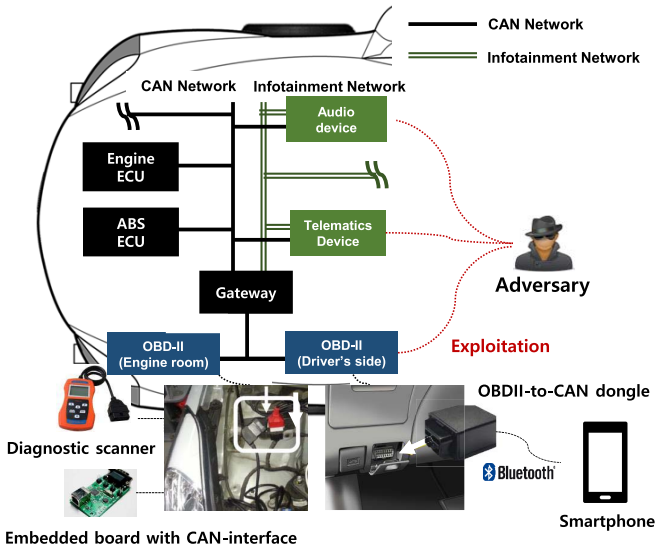


Fig. 4. Access points to automotive CAN (In general, modern vehicles have either an audio system or a telematics system).

A. Attack Surfaces

In this subsection, we categorize these attack surfaces into three types: (1) physical access-based attacks, (2) Wireless Access-based Attacks requiring initial Physical Access, and (3) wireless access-based attacks without physical access.

1) Physical Access-Based Attacks: Modern vehicles have physical access points to automotive CAN such as the OBD-II port and USB (or Disc) ports that can be used for ECU diagnosis and ECU firmware updates. However, these access points may be exploited by CAN bus attacks as follows.

a) OBD-II: In [2], [3], the OBD-II port was directly accessed by the adversary's computer to inject CAN packets to automotive CAN.

In [4], a PassThru-based attack was proposed. PassThru is the SAE J2534 standard that provides reprogramming/diagnostic functions to ECUs [95]. Checkoway *et al.* chose one commonly used PassThru device, which runs a variant of Linux, to find vulnerabilities. The target device is equipped with USB, WiFi and a connector that plugs into the OBD-II port. In their result, the authors discovered that the target PassThru device could be accessed quite easily by adversaries who are connected to the same WiFi network as the target device because there is no authentication process to access the PassThru device. The adversary was then able to install a malicious binary on the target device to compromise the ECU through firmware updates.

Another OBD-II-based attack using an attack tool called CANtact was additionally introduced in [5]. CANtact provides easy CAN bus attacks, such as denial of service (DoS) and CAN packet injections.

b) USB (or Disc) of an Audio System: Checkoway *et al.* found that their target audio system automatically re-flashes its own firmware upon recognizing a specific file name on a USB flash drive (or a disc) [4]. They demonstrated that an adversary can transmit malicious CAN packets on automotive CAN through modified firmware if the adversary can install the modified one on the audio system of a target vehicle.

In addition, the authors reverse-engineered the firmware of their target audio system to find vulnerabilities related to MP3 and WMA file parsers. In these parsers, there was one file read function that did not check the size of an input file. Testing this vulnerability, the authors produced a WMA audio file that plays perfectly on a PC, but transmits malicious CAN packets when played on their target audio system. Since these malicious "song" files have the potential to spread rapidly through file sharing systems such as BitTorrent, the scale of attack could be greater than the firmware update attack described above.

2) Wireless Access-Based Attacks Requiring Initial Physical Access: Attacks through physical access have a limited range, so longer range attacks such as physical access-based attacks relying on wireless channels have been studied in [6]–[13]. In general, two types of wireless channels are connected to automotive CAN, and both can be deployed to modern vehicles: (1) an OBD-II dongle equipped with a Bluetooth interface (or an OBD-II dongle equipped with a cellular modem),[3] and (2) a telematics device. In this regard, an adversary can exploit these wireless channels either to send malicious CAN packets remotely, if a victim (e.g., a driver) is successfully duped into connect these OBD-II dongles to their vehicle [6]–[11], or to compromise a telematics device via a USB port or an SD card slot [12], [13]. The following are examples of these attacks.

a) OBD-II Dongle With a Wireless Channels: Woo *et al.* demonstrated an attack based on an OBD-II dongle with a self-diagnostic application[4] [6].

In [7]–[10], the most popular vehicle diagnostic applications paired with OBD-II dongles were analyzed to show that commercial diagnostic applications could be exploited to manipulate control of a target vehicle.

Foster *et al.* analyzed one OBD-II dongle that is known to be used for insurance purposes (e.g., Metromile: Pay-per-mile insurance[5]) [11]. Unlike regular OBD-II dongles equipped with a Bluetooth interface, this OBD-II dongle is equipped with a cellular modem to provide direct communication between the dongle and a remote entity (i.e., an insurance company). By exploiting the vulnerabilities of the communication channel provided by this cellular modem, Foster *et al.* was able to compromise the target OBD-II dongle program and transmit malicious CAN packets to automotive CAN.

[3]OBD-II dongles have been used for ECU self-diagnostics or car insurance policies (e.g., total mileage per year). A popular OBD-II dongle is the ELM 327 dongle (https://www.elmelectronics.com/products/ics/obd/)

[4]In smartphone application markets like Google Play Store and Apple Appstore, there are hundreds of self-diagnostic applications available to help drivers diagnose car troubles without assistance from an auto-repair shop.

[5]https://www.metromile.com/

b) USB (or Disc) of a Telematics Device: Jo *et al.* analyzed a vehicle equipped with an Android-based telematics device [12]. Since the target device provides a navigation service, regular updates are recommended for routine maintenance, and these updates are carried out via an SD card. Jo *et al.* found that the target device also updates other telematics programs during the map update process, and the manufacturer of the target telematics device used a Google test signing key, which is publicly available. Based on these vulnerabilities, the authors demonstrated that an adversary could cryptographically sign the modified update file by using the test signing key and install a modified file onto the target device.

In the work [13], a smartphone connected to a target telematics device via USB was used to inject malicious CAN packets to automotive CAN.

3) Wireless Access-Based Attacks Without Physical Access: Even though the attacks based on wireless access-based attacks requiring initial physical access provide remote access points to an adversary, these attacks are dependent on the success of physical access. However, physical access to a vehicle is limited to some authorized individuals (e.g., auto mechanics) and might be easily recognized by drivers.

To overcome these limitations of physical access, remote attacks on automotive CAN without any physical access have been proposed in [4], [14], [15]. These attacks exploit vulnerabilities of wireless channels (i.e., Bluetooth or cellular channels) provided by telematics devices to remotely transmit malicious CAN packets to automotive CAN. A detailed explanation of these attacks follows.

a) Bluetooth Channel: Checkoway *et al.* demonstrated that a Bluetooth channel could become an access point to automotive CAN [4]. In their experiment with a target telematics device supporting a Bluetooth interface, the authors reverse-engineered a program related to a Bluetooth protocol stack and found vulnerabilities of `strcpy` functions that do not check the length of input. Using a buffer overflow attack based on these vulnerabilities, they showed that a compromised device (e.g., an infected smartphone) that has been Bluetooth-paired with the target device can execute an arbitrary code on the target telematics device.

b) WiFi Channel: In the work [14], the parking and driving modules of a Tesla Model S were compromised by an attacker masquerading as the Service Set Identifier (SSID) running on Tesla's Auto Shop. The spoofed SSID allows the attacker to connect the telematics unit by using the auto WiFi connection feature provided by the Tesla Model S, and this led to the remote exploits on the web browser of the telmatics unit.

c) Cellular Channel: Checkoway *et al.* demonstrated an attack on the GM telematics service called OnStar [4]. First, the authors reverse-engineered the aqLink protocol, which is a proprietary protocol used for communication between the GM telematics device and the telematics server, to analyze protocol specifications such as packet size and protocol flows. Through the analysis of the aqLink protocol, they were able to infer that the maximum packet size is 1024 bytes. Then, they analyzed the authentication process between their target telematics device and the GM telematics server and discovered that the random number generator used in this authentication process was re-initialized whenever the telematics unit started. Consequently, the authors were able to show that an adversary could remotely transmit an exploitable binary to the target telematics device by using the vulnerabilities they found.

Likewise, Miller *et al.* found vulnerabilities in Jeep's Uconnect, which resulted in a recall of 1.4 million vehicles [15]. The attacks on Uconnect are based on a femtocell device (i.e., Sprint Airave 2.0 [6]), which acts as a small cell tower for customers who live in areas with poor signal reception. The authors were able to obtain a shell of the femtocell device by using public exploits[7] establishing Telnet or HTTPS services to the device. By using this femtocell and the corresponding exploits, the authors were able to communicate with all Uconnect devices in Jeep vehicles without undergoing an authentication process. They demonstrated that an adversary can access a Uconnect device through the femtocell device and send arbitrary CAN packets remotely to automotive CAN.

Table II illustrates the summary of verified attack surfaces and the explanations about adversary types are described in the following Section III-B.

B. Adversary Model

This section aims to focus on the actual actions of adversaries and their role in performing different types of attacks that ultimately impact operations of compromised vehicles. We generalize these bad actors into two types.

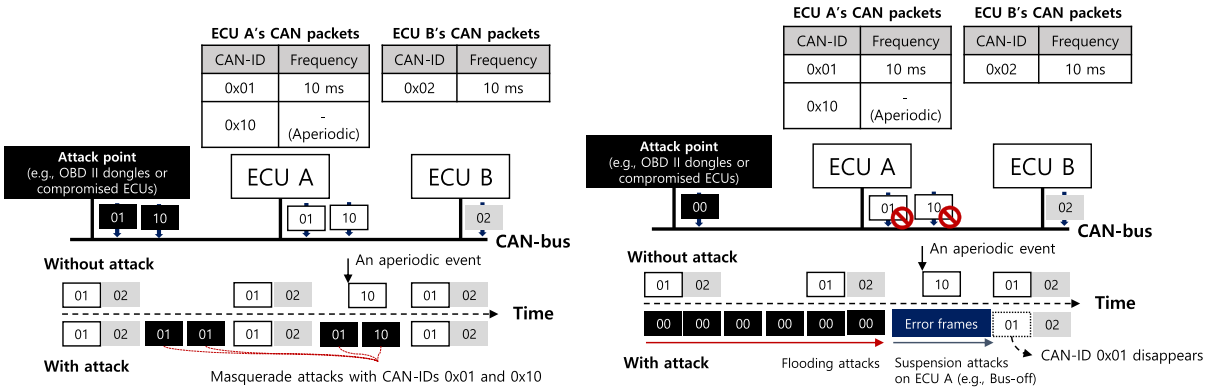
1) Type a Adversary: The first type of adversary, Type A, attempts to access automotive CAN though the OBD-II port. As previously discussed, some drivers make use of a dongle that supports a CAN-to-external networks interface through the OBD-II port for self-diagnostic services (e.g. smartphone apps) or car insurance services (e.g., pay-per-mile insurance schemes).

In light of this, a Type A adversary can exploit these OBD-II-based services to control safety-critical ECUs like the engine ECU, or to cause a network failure of automotive CAN by transmitting a number of CAN packets to the vehicle.

2) Type B Adversary: The second type of adversary can compromise ECUs through vulnerable physical or remote access points. In general, modern vehicles have an ECU (i.e., telematics or multimedia ECU) that keeps remote access ports open for external networks (e.g., cellular, WiFi, or Bluetooth communications) as well as physical access ports (e.g., USB, Disk, and SD card slots). Upon gaining access to one access point, the Type B adversary can compromise an ECU and inject malicious CAN packets to automotive CAN in order either to control safety-critical ECUs like the engine ECU or to cause a network failure even if the driver is not using an additional service through a dongle connected to the OBD-II port.

[6]Uconnect services operate on the Sprint network to provide vehicles with external network access.

[7]https://files.persona.cc/zefie/files/airvana/telnet.html



(a) Masquerade attacks

(b) DoS attacks (Flooding and suspension attacks)

Fig. 5. Attack scenarios on automotive CAN: (a) While CAN messages with CAN IDs (0×01 and 0×02) are transmitted every 10ms, the transmission of a CAN message with CAN ID (0×10) is triggered by an aperiodic event. An adversary can impersonate ECU A by sending CAN messages with 0×01 or 0×10. (b) In a flooding attack, the adversary can preempt CAN bus by sending a large number of CAN messages with 0×00. In a suspension attack, the adversary can disable CAN communications from ECU A by using a bus-off attack.

TABLE II
KNOWN ATTACK SURFACES OF AUTOMOTIVE CAN

	Attack Surface	Attack Examples	Attack Range	Adversary Type
Physical Access-based Attacks	OBD-II	Embedded boards (e.g., CANtact) [2], [3], [5]	Short	Type A
		Diagnostic Tools (e.g., PassThru) [4]	Short	Type A
	USB ports (or Disk) of audio systems	Firmware update of ECUs [4]	Short	Type B
		Modification of song files (e.g., Modification of MP3 (or WMA) files) [4]	Medium - Long	Type B
Wireless Access-based Attacks requiring initial Physical Access	OBD-II with additional dongles	OBD-II-to-Bluetooth dongle (e.g., Self-diagnostic service) [6], [7], [8], [9], [10]	Medium - Long	Type A
		OBD-II-to-Cellular dongle (e.g., Car insurance services) [11]	Medium - Long	Type A
	USB ports (or SD card) of telematics systems	Firmware update of a telematics device [12], [13]	Medium - Long	Type B
Wireless Access-based Attacks without Physical Access	Bluetooth	Bluetooth services through smart phones [4]	Medium - Long	Type B
	WiFi	WiFi services [14]	Medium - Long	Type B
	Cellular	Telematics services [4], [15]	Long	Type B

The 5th column of Table II shows which adversary types are associated with known attack vectors described in the previous section.

C. Attack Scenarios

1) Masquerade Attack: A Type A or Type B adversary can send malicious commands with fake CAN IDs through automotive CAN to gain unauthorized control of a vehicle by impersonating uncompromised ECUs, known as a masquerade attack. During this type of attack, adversaries typically employ message replay or message fabrication to alter vehicle controls. In attacks using message replay, CAN packets observed during automotive CAN monitoring are replayed without any modification. In attacks using message fabrication, the adversary fabricates a CAN packet by forging the CAN ID field and the CAN data field. Masquerade attacks that are based on message replay or message fabrication are achieved through either masquerade with periodic CAN packets or masquerade with aperiodic CAN packets, as shown in Fig. 5(a).

2) DoS Attack: An adversary can perform message flooding attacks on automotive CAN to deplete network bandwidth or suspension attacks on a specific ECU to interfere with message transmission, as shown in Fig. 5(b).

- Message flooding attack: In this attack, a Type A or Type B adversary continuously sends a large number of high priority CAN packets (i.e., the highest CAN ID of 0×00). While this attack is in progress, other lower priority CAN packets from all other ECUs are unable to be transmitted over CAN bus, which could result in unexpected events during vehicle driving.
- Suspension attack: A specific ECU cannot join CAN communications if a Type A or Type B adversary performs a suspension attack on it. One possible method for this attack is to perform the bus-off attack[8] [96].

3) Combined Attack: To gain greater control of safety-critical ECUs, an adversary can perform both masquerade attacks and DoS attacks simultaneously. For example, a message fabrication attack was not enough to control the Jeep

[8]In a bus-off attack, an adversary deliberately generates a CAN bus error whenever a target ECU transmits a CAN packet. If the adversary can inflict successive CAN bus errors, the target ECU will eventually switch to the bus-off state, which means that it is temporarily unable to participate in CAN communication.

Cherokee's brake because its ABS collision prevention system (i.e., ABS ECU) does not allow controlling the brake by periodically sending CAN packets indicating ABS's switch-off [15]. In order to control Jeep's brake, the transmission of ABS's switch-off message was blocked by suspension attacks on Jeep Cherokee's ABS ECU. Then, CAN packets indicating ABS's switch-on were transmitted at an allowable frequency as if these packets had been generated by the ABS ECU to deceive other ECUs.

a) Difference Between Type A and Type B Adversaries: From the viewpoint of attack scenarios, there are no significant differences between the characteristics of Type A and Type B adversaries. Both adversaries can perform masquerade, DoS, and combined attacks on automotive CAN, but whether or not the adversary can gain access to the internal storage of an ECU depends on the adversary type. In short, a Type A adversary chooses the attack method based on their limited ability, whereas a Type B adversary has a more sophisticated skill set that enables them to access to the internal storage of an ECU where secure information may be stored.

IV. Preventative Protection System

Fuzzing studies that can identify vehicle's weaknesses and anti-analysis studies that make reverse analysis difficult have been studied as preventative systems as follows.

A. Fuzzing Methods

In general, fuzzing can be used to find new vulnerabilities of a target system by sending systematically malformed input values to the target system.

According to the existing fuzzing methods [30]–[32], a fuzzer consists of a message generator and a target monitor. The message generator is concerned with the generation of malformed CAN messages that are sent to the fuzzing target ECU in order to provoke unexpected failures, and the target monitor is used to judge whether the target ECU being tested has been affected by the malformed input from the message generator.

To design an efficient and accurate fuzzing method, a model-driven message generator is needed. In a model-driven message generator, CAN messages are systematically generated by considering the meaning of each byte of the CAN data field defined in a CAN database[9] that describes information about each byte that the data field transfers for all CAN packets. However, the message generators presented in the fuzzing methods [30]–[32] either do not take into account the meaning of the CAN packet data fields, or do not describe the message generation model in detail. Thus, there is a need for a systematical automotive CAN fuzzing method that can infer the meaning of every byte in the CAN data field [97]–[99] and network configurations [100].

B. Anti-Analysis Methods

To make an attack step identifying attack surfaces and attack packets difficult, there have been several anti-analysis methods, including ECU firmware protection and anti-monitoring of automotive CAN.

1) Firmware Protection Methods: According to existing attacks on automotive CAN that exploit vulnerabilities of ECU firmware [2], [3], [12], a diagnostics tool (e.g., PassThru), a debugging port (e.g., Joint Test Action Group (JTAG)), and an official website (e.g., telematics service website) have been used to obtain specific ECU firmware. To prevent easy access to ECU firmware, encryption and obfuscation can be applied [33]. In addition, protection methods for debugging ports (e.g., JTAG) can be used for access control to the ECU firmware. For example, anti-fuse methods meant to disable the JTAG port [34], [35] and secret key-based access control methods that control JTAG access [36], [37] could be introduced to prevent unauthorized access to debugging ports of a specific ECU.

2) Anti-Monitoring Methods: In general, a monitoring step of automotive CAN is commonly used to find attack packets that are targeting safety-critical ECUs such as the engine ECU or the brake ECU. Since CAN packets are not encrypted, an adversary is left relatively unchallenged and unhindered to easily find meaningful CAN packets by simply monitoring network activity. In response, the work [38] presented an encryption method for automotive CAN to create an obstacle between the adversary and the CAN packet information. In this work, a centralized node authenticates all ECUs. Then, an encryption key for the CAN data field is distributed to all ECUs that have been authenticated by the centralized node. Since an unauthorized node cannot decrypt the encrypted CAN packets, it becomes difficult for an adversary to locate and therefore analyze meaningful CAN packets related to controlling a safety-critical ECU.

While encryption methods for the CAN data field can hide meaningful information in the CAN data field, the CAN ID field itself cannot be hidden from an adversary. Since finding the CAN IDs of safety-critical ECUs is also required to control safety-critical ECUs, several reallocation methods for the CAN ID field were proposed to make it difficult to identify any linked relationship between a safety-critical ECU and a CAN ID [39]–[44]. However, in these works, even though every CAN ID is reallocated, reallocation methods must still preserve the pre-defined priority of CAN IDs configured by car manufacturers due to the CAN arbitration process. Thus, this fact limits the degree of CAN ID reallocation methods. Also, these anti-monitoring systems could still be circumvented by a compromised ECU attacked by a Type B adversary because the compromised ECU would be able to decrypt/infer the encrypted CAN data/reallocated CAN ID.

Table III shows the summary of existing preventative protection methods we reviewed.

V. Intrusion Detection

Intrusion detection systems (IDS) for automotive CAN have been a large focus of much research due to the proven difficulty in blocking all unknown attacks before an attempt is underway. To detect unknown attacks as soon as possible and respond appropriately, intrusion detection is absolutely necessary for automotive CAN. Current intrusion detection

[9]Every vehicle model has its own CAN database to define all messages transmitted on CAN bus. (e.g., https://github.com/commaai/opendbc)

TABLE III
SUMMARY OF EXISTING PREVENTATIVE PROTECTION METHODS

	Research work	Main Technology	Contribution	Drawback
Fuzzing	Bayer et al. [30] Oka et al. [31]	Model-driven CAN packet generation	- Generation of CAN packets considering a CAN-Database	- There is no explicit explanation about the model-driven message generation.
	Fowler et al. [32]	Random CAN packet generation	- Easy generation of CAN packets	- It does not consider a CAN-database.
Anti-analysis	Yu et al. [33]	Firmware encryption and obfuscation	- Encryption and obfuscation firmware	- Firmware can be accessed by other methods such as ROM dump. - It is not secure against a Type B adversary.
	Hely et al. [34] Rosenfeld et al. [35]	Access control to Firmware	- Disabling a debugging port	- Firmware can be accessed by other methods such as ROM dump. - It is not backward compatible. (The debugging port all legacy ECUs should be disabled.)
	Novak et al. [36] Pierce et al. [37]		- A symmetric key-based access control	- Firmware can be accessed by other methods such as ROM dump. - It is not secure against a Type B adversary. (A symmetric key for access control could be compromised)
	Mundhenk et al. [38]	Encryption of the CAN-data field	- Encryption of CAN-data by a symmetric key	- Encryption/Decryption overhead - It is not secure against a Type B adversary. (A symmetric key for encryption could be compromised)
	Han et al. [39] Humayed et al. [40] Wu et al. [41] Woo et al. [42] Cheng et al. [42] Groza et al. [44]	Reallocation of the CAN-ID field	- Reallocation of CAN-ID by a secret key	- Reallocation overhead - It is not secure against a Type B adversary. (A secret key for reallocation of CAN-ID could be compromised)

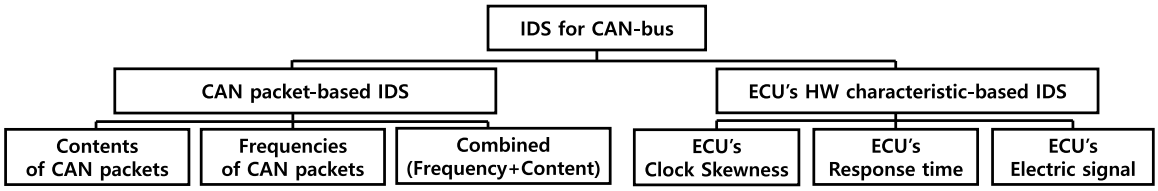


Fig. 6. Classification tree for CAN bus intrusion detection techniques.

techniques fall into two categories: (1) a CAN packet-based intrusion detection approach that focuses on the features of CAN packets (e.g., content or frequencies of CAN packets) and (2) an ECU hardware characteristic-based intrusion detection approach that focuses on the physical features of ECUs (e.g., characteristics of the CAN low and CAN high signals generated by an ECU CAN transceiver). Fig. 6 shows our classification tree based on the two categories.

This section discusses the merits and the drawbacks of both categories.

A. CAN Packet-Based Intrusion Detection Approach

1) Frequency-Based IDS: In [45]–[49], the transmission frequencies of CAN packets are used to detect intrusion. Since almost every CAN packet is periodically transmitted on CAN bus according to their own frequencies set by the manufacturer, these transmission frequencies can be used as one parameter for the detection of attacks on automotive CAN. For example, if an adversary performs a DoS attack (i.e., a message flooding attack or suspension attack) on automotive CAN, the transmission frequencies of some CAN packets will inevitably differ radically from normal frequencies. Thus, an IDS based on the frequencies of CAN packets can easily root out DoS attacks on automotive CAN by observing abnormal frequencies of CAN packets. However, frequency-based intrusion detection systems have difficultly determining whether or not aperiodic CAN packets, which do not have transmission cycles, are being used for masquerade attacks. For example, if cruise control was activated while driving on the highway, and aperiodic CAN packets were transmitted on automotive CAN by this mode, the frequency-based intrusion detection system would not be able to distinguish whether these aperiodic packets originated from the valid cruise control mode or an externally sourced masquerade attack.

Marchetti *et al.* makes use of the sequence of CAN packets that are transmitted on automotive CAN to detect anomalies of CAN communications [50]. They observed that particular patterns on automotive CAN (i.e., the sequences of CAN IDs) are repeated according to the transmission frequencies of CAN packets. To uncover the CAN ID sequences of a vehicle's CAN bus, the authors proposed a transition matrix (m by m), where m is the number of CAN IDs monitored on

automotive CAN, and all elements of the transition matrix are initialized to false. Then, if the CAN ID of 0×01 is followed by the CAN ID of 0×02, true is set to the corresponding row and column of the transition matrix. Once the transition matrix is built, an injected malicious CAN packet that does not conform to the sequence of CAN IDs can be detected when the corresponding position of the transition matrix checked returns false. Likewise, the work of [51] proposed an IDS based on the sequence of CAN packets.

However, these methods are still inadequate in that it is unable to distinguish between valid aperiodic CAN packets from normal operations and invalid aperiodic CAN packets from masquerade attacks.

2) Content-Based IDS: In [52], Stabili *et al*. proposed a Hamming distance-based IDS that measures the Hamming distance between the CAN data fields of two consecutive CAN packets having the same CAN ID in order to detect anomalies by comparing the measured Hamming distance value with the previously created and validated Hamming distance value. However, even though this method requires little computational overhead, the experiment results in [52] prove that the Hamming distance-based IDS cannot detect reply attacks on automotive CAN.

Markovitz *et al.* defined four data types that are normally represented in the CAN data field: a constant value type, a multi-value type, a counter value type and a sensor value type [53]. Since CAN packet specifications for all commercial vehicles (i.e., CAN database) are not publicly available, the researchers developed a classification algorithm that splits 64 bits of the CAN data field into several parts. Then, the divided parts are separately assigned to one of the four data types. As an initial step, a classification step is applied to every CAN ID to identify which data type is used in the associated CAN data. After the classification step, a specific detection rule for each data type is defined to detect attacks on automotive CAN. However, this specific detection rule cannot be used for the detection of a masquerade attack using replayed CAN packets because the characteristics of the replayed packets does not violate the detection rule.

As another content-based approach, a deep neural network (DNN)-based IDS was proposed in [54]–[60]. In [54], Kang *et al.* found that malicious CAN packets can be classified by a deep neural network (DNN) based on a rectified linear unit (ReLU) [101]. As an initialization step, all CAN packets are classified by CAN ID. Then, in a learning step, the 64-bit value of the CAN data field becomes the 64 dimensional input values for DNN. This learning step is performed for all CAN IDs. To evaluate the performance of the DNN-based IDS, CAN packets were generated by a CAN packet simulator named Open Car Testbed and Network Experiments, or OCTANE [102].

In [55], a CAN ID sequence of 29 CAN messages is converted into a 2D grid data, and then this grid data is used as input for Inception-ResNet, which is a proven deep convolution neural network (DCNN).

However, it is unclear whether these DNN-based IDSs can prevent masquerade attacks with aperiodic CAN packets because the aforementioned experiment is based on either simulated CAN traffic from the TPMS ECU, the engine ECU, and the body ECU or specific attack scenarios like DoS, spoofing RPM and spoofing gear attacks. Thus, the detection performance of a DNN-based IDS need to be verified in attack scenarios based on masquerade attacks with aperiodic CAN packets.

Taylor *et al.* proposed a recurrent neural network (RNN)-based IDS using long short-term memory (LSTM) [56]. In this RNN model, a 64-bit value of the CAN data field is used for the input value of the RNN learning phase. While the experiment of [54] is based on simulated CAN traffic, the RNN-based IDS was evaluated by the CAN packet generated by a 2012 Subaru Impreza. Likewise, the work of [57]–[59] proposed LSTM-based IDSs. However, it is also uncertain whether these LSTM-based IDSs can detect masquerade attacks with aperiodic CAN packets.

Wasicek, *et al.* [60], proposed a context-aware IDS that uses sensor information to create reference models of the target's physical operations, and then it evaluates the correctness of the measured sensor data by comparing it against the reference models. The proposed context-aware IDS is based on a bottleneck artificial neural network (ANN), a special variant of ANNs known to be suitable for anomaly detection [103]. However, it should be verified that the context-aware IDS are also able to distinguish between the aperiodic CAN packets from valid vehicle operations and the aperiodic CAN packets from masquerade attack attempts.

Another IDS using CAN ID filtering was proposed in [61]. In this CAN ID filtering-based IDS, if an ECU receives a CAN packet from another ECU with the same CAN ID that it has, the ECU filter outs this packet by identifying it as a masquerade or replay attack attempt. After the detection of an attack attempt, an alert signal is broadcasted to other ECUs. However, it needs a special CAN signal for the alert broadcast that violates the CAN standard.

3) Combined IDS (Frequency + Content): Müter *et al.* introduced three intrusion detection parameters for the detection of attacks on automotive CAN: Content of CAN packets, frequencies, and sources [62]. Content of CAN packets (i.e., values of the CAN data field) are used to see if there are inappropriate values (e.g., unacceptable vehicle speeds) or abnormal correlations between consecutive CAN packets. In addition, the frequencies and sources of CAN packets are checked to identify abnormal frequencies (e.g., message flooding attacks or suspension attacks) and inappropriate CAN packets (e.g., engine control packets originating from the infotainment domain),[10] respectively. However, Müter *et al.* did not perform any experiments for the evaluation of their IDS.

The authors also proposed an entropy-based IDS for the detection of CAN bus attacks [63]. In general, the entropy of CAN packets is lower than that of the packets from a standard computer network (e.g., TCP/IP) because CAN packet transmission frequencies and allowable CAN data values are predetermined in the CAN database. If a number of malicious

[10]In order to identify the CAN bus domain from which a CAN packet originates, an IDS module must be installed in every domain.

CAN packets that are not defined in the CAN database are injected into automotive CAN, the entropy of CAN bus traffic is sure to be affected. Thus, an abnormal entropy of CAN bus traffic could be used as an indicator of attacks on automotive CAN. However, the entropy-based IDS has not been verified to deal with all of the attack scenarios because Müter *et al.* performed a constrained experiment based on limited attack scenarios, including the increased transmission frequency of a CAN packet, message flooding, and injection of abnormal CAN packets. Subsequently, Marchetti *et al.* evaluated the performance of the entropy-based IDS by using the CAN packets from a real vehicle [64]. The experimental results show that an entropy-based approach is the only possible way to detect CAN bus attacks with a high volume of forged CAN packets, especially in cases where the adversary injects hundreds of forged CAN packets in a matter of seconds. Thus, the entropy-based approach could be considered inappropriate to detect an attack with a low volume of forged CAN packets.

B. ECU Hardware Characteristic-Based Intrusion Detection Approach

1) Clock Skew-Based IDS: ECUs operate based on their local clocks with distinct skews. Since these distinct skews can be used for device fingerprinting, a clock skew-based IDS was proposed in [65]. In the clock skew-based IDS, the transmission interval of a periodic CAN packet is monitored to estimate an ECU clock skews because there are no available timestamps in CAN packets, as opposed to the Ethernet packet-based works where timestamps exists [104]. According to their evaluation based on real vehicles—a 2013 Honda Accord, a 2010 Toyota Camry, and 2010 Dodge Ram Pickup—it was found that the clock skew-based IDS was able to detect masquerade attacks with periodic CAN packets. During the experiment, the false positive rate was 0.055% when the time required for attack detection was set to 200 ms. However, the clock skew-based IDS can only be applied to periodical CAN packets. In other words, if aperiodic CAN packets such as diagnostic packets are used for masquerade attacks, the clock skew-based IDS cannot distinguish whether these packets are originating from diagnostic services or masquerade attacks. Furthermore, ECU clock skews can be emulated to circumvent the clock skew-based IDS as proposed in [105], [106].

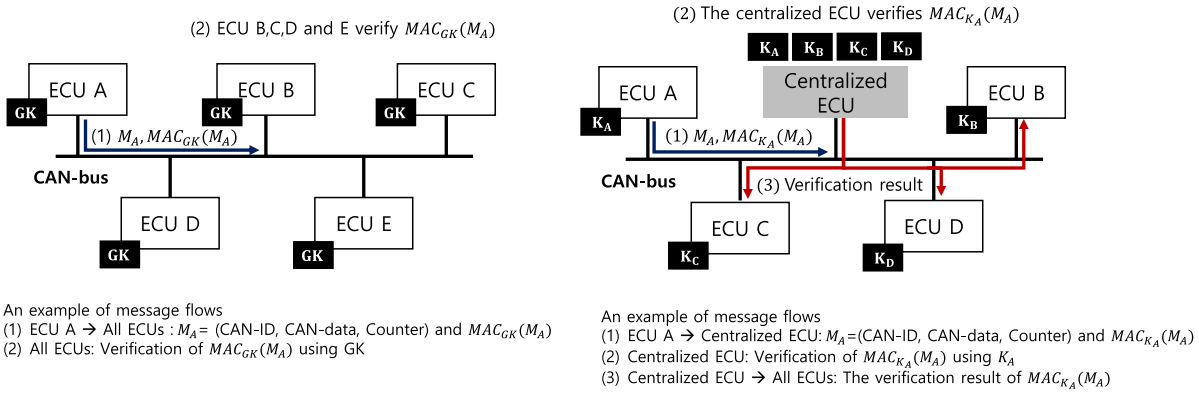
2) Response Time-Based IDS: In the CAN standard, every ECU is designed to send a response message (i.e., data frame) when it receives a request message (i.e., remote frame). For example, if a remote frame with the CAN ID of 0×01 is transmitted on CAN bus, then the ECU with the CAN ID of 0×01 will respond to this remote frame using a data frame with the CAN ID of 0×01. Lee *et al.* observed that each ECU has a unique response time distribution regarding request messages it receives [66]. Even though some ECUs have the same clock frequency, response time distributions are different due to environmental factors, such as the ECU temperature or location [107]. They also found that response time distributions for all ECUs are influenced by malicious CAN packets on CAN bus.

Based on this observation, the authors proposed a response time-based IDS based on the time intervals between request and response messages in ECUs. To detect attacks on automotive CAN, the response time-based IDS is designed to periodically transmit request messages (i.e., remote frames) to all ECUs. Whenever it finds that the response time of an ECU deviates from the corresponding response time distribution which was measured in automotive CAN without an attack, it determines that there are malicious CAN packet injections on automotive CAN. However, the response-based IDS cannot detect some attacks on automotive CAN that do not affect the response time of an ECU. For example, while the response time-based IDS is checking the response time of an ECU with the CAN ID of 0×01 after sending a request message with CAN ID of 0×01, injection of a few CAN packets with CAN IDs that are set to lower priority than 0×01 (e.g., 0×02) cannot affect the response time of the ECU with CAN ID of 0×01 because the response packet from the ECU can preempt automotive CAN even if there is an arbitration between an attack packet (e.g., 0×02) and the response packet (e.g., 0×01).

As another response time-based IDS, [67] introduced a new IDS type using the propagation time of the physical signals. This work found out that the propagation delays from ECUs to each of the bus ends depend on the position of the CAN transmitters, and the differences between signal arrival times (i.e., propagation time) at the CAN bus ends are used to detect illegal message transmissions. However, the method proposed in [67] cannot distinguish between normal messages and attack messages that originate from the same location (i.e. same domain), which could affect the attack detection rate.

3) Electric Can Signals-Based IDS: All ECUs (i.e., CAN-transceivers) generate CAN high and CAN low signals whenever they transmit their own CAN packets. Murvay *et al.* found that these electric signals can be used to identify ECUs because manufacturing variations and imperfections cause the shape of electric signals from all ECUs to vary slightly [68]. In other words, even if two ECUs generate an identical CAN packet, the electric waveforms generated from the two ECUs will produce different shapes. To measure the unique shapes of electric signals generated by one ECU, the authors focused on the CAN ID field of CAN packets because a CAN ID is the pre-fixed constant value that is mainly used by one ECU. Thus, in a learning phase of electric CAN signals-based IDS, electric signals corresponding to the CAN ID field of CAN packets (i.e., CAN high and CAN low signals in the CAN ID field) are only measured to create digital fingerprints for all ECUs. These fingerprints are then used as a reference point during the detection of masquerade attacks on automotive CAN. Murvay *et al.* evaluated the electric CAN signal-based IDS in a simulated environment that was composed of ten USB-to-CAN devices and five CAN development boards.

However, their experiments were performed in a low-speed CAN bus (10 kbit/s) environment, which is not commonly used by automotive CAN. In general, 500 kbit/s is used for safety-critical CAN bus domains. In addition, this work does not take arbitration of automotive CAN into consideration, which is noteworthy because electric signals in the CAN ID



(a) Group key-based approach (b) Pairwise key-based approach (with a centralized ECU)

Fig. 7. Two representative approaches for message authentication (GK: a group key, Counter: a counter value, K_X: a unique key of ECU$_X$, MAC$_k$(): message authentication code using a key k).

field can be generated by multiple ECUs. This may occur because a few bits in the CAN ID field can be transmitted by more than one ECU during the arbitration process. Lastly, since this work is based on a pre-fixed reference set acquired from batch learning, it cannot handle unpredictable environmental changes (e.g., temperature or battery levels) of the automotive CAN. According to [86], the status of a vehicle's battery charge is one environmental factor that influences the shape of electric signals generated by all ECUs. In other words, a context-aware update mechanism for ECU fingerprints is needed to deal with unpredictable environmental changes in automotive CAN.

In [69], a new and electric CAN signal-based IDS was proposed. This work used the Extended ID field to deal with the above described arbitration issue. It is guaranteed that the electric signals in the Extended ID field will be generated from only one ECU because the CAN ID field can be configured to complete the arbitration process in the CAN ID field. However, this work requires all ECUs to update firmware for the use of the Extended ID field because it is not commonly used in automotive CAN. In addition, this method has not been verified in a real CAN-bus environment and does not provide a context-aware update of ECU fingerprints.

To deal with the limitations of [69], another way to measure electric signals of CAN bus was proposed in [70]. In this work, Choi *et al.* focused on the signal change that occurs when a dominant bit is changed to a recessive bit, or vice versa. This transient signal is considered to have a unique characteristic caused by passive (e.g., resistance) and active (e.g., capacitance) transmitter components. The authors also evaluated their work in two real CAN bus environments—with a 2010 Hyundai YF Sonata and a 2014 Kia All New Soul. However, the method also fails to consider a context-aware update of fingerprints. In this context, the work of [71] was proposed, which was designed to handle varying conditions like temperature and voltage level of the power supply.

Table IV shows the summary of existing intrusion detection methods we reviewed.

VI. Message Authentication

Existing message authentication methods for automotive CAN were designed to deal with masquerade attacks. In general, they are based on the use of authentication tags, such as message authentication code (MAC). When a message authentication is applied to automotive CAN, two issues need to be considered. First, the way authentication keys are shared should be defined for the generation of authentication tags. Second, a secure way to transmit authentication tags must be designed because the CAN data field of a CAN packet is 64 bits, which is not enough to include an authentication tag that is usually longer than 64 bits. Fortunately, there has been a wealth of research on message authentication for automotive CAN to deal with these two issues as follows.

A. Authentication Key Sharing Methods

Keys for message authentication are either shared via a group key-based approach or a centralized node-based approach. The merits and the drawbacks of these two key sharing approaches are explained as follows.

1) Group Key-Based Approach: In this approach, all ECUs share one authentication key to generate authentication tags for their corresponding CAN packets. A number of works have indeed adopted a group key-based authentication method [6], [72]–[80]. Fig. 7(a) shows the architecture of the group key-based approach and an example of message authentication protocol flows. Since all ECUs share one key (i.e., a group key), there is no need to generate and then also transmit many authentication tags that increase proportionately with the number of receivers. As one might guess, only one authentication tag is generated per one CAN packet.

Unfortunately, this approach still cannot block masquerade attacks from Type B adversaries. The group key used to generate authentication tags could be exploited by Type B adversaries, who would then be able to corrupt an ECU [108]. Even though a group key is assigned to one group domain as proposed in [80], there is a possibility that some ECUs, such as

TABLE IV
SUMMARY OF EXISTING INTRUSION DETECTION METHODS

	Research work	Main Technology	Contribution	Drawback
CAN packet -based IDS	Taylor et al. [45] Song et al. [46] Tomlinson et al. [47] Olufowobi et al. [48] Young et al. [49]	Frequency-based IDS	- Measurement of the frequencies of CAN packets	- Aperiodic CAN packets cannot be handled.
	Marchetti et al. [50] Katragadda et al. [51]		- Measurement of the sequence of CAN packets	
	Stabili et al. [52]	Content-based IDS	- Measurement of the Hamming distance between two consecutive CAN packets	- Replayed CAN packets cannot be handled.
	Markovitz et al. [53]		- Classification of CAN-data (e.g., constant values, multi-values, and sensor values)	
	Kang et al. [54] Song et al. [55]		- DNN-based approach	- It is uncertain whether these IDSs can prevent masquerade attacks with aperiodic CAN packets.
	Taylor et al. [56] Longari et al. [57] Hossain et al. [58] Tariq et al. [59]		- RNN-based approach	
	Wasicek et al. [60]		- Bottleneck ANN-based approach	
	Ansari et al. [61]		- CAN-ID filtering-based approach	- It needs a special type of CAN signal that violates the CAN standard.
	Müter et al. [62]	Combined IDS	- Use three detection parameters: CAN-data , frequencies and sources of CAN packets	- It need to be verified in the experiment with real data of automotive CAN.
	Müter et al. [63] Marchetii et al. [64]		- Measurement of entropy of CAN packets	- Aperiodic CAN packets cannot be handled. - It is insufficient to detect a CAN-bus attack with a low volume of forged CAN packets.
ECU's HW Characteristic -based IDS	Cho et al. [65]	Clock skew-based IDS	- Measurement of clock skew of ECU's micro-controller	- Aperiodic CAN packets cannot be handled. - It does not consider an adversary that can emulate an ECU's clock skew.
	Lee et al. [66]	Response time-based IDS	- Measurement of response time using remote frames	- It cannot detect some attacks that affect ECU's response time.
	Murvay et al. [67]		- Measurement of propagation delay of CAN packets	- It cannot distinguish between normal messages and attack message that are transmitted by the same location (i.e. same domain)
	Murvay et al. [68] Choi et al. [69] [70]	Electric CAN signal-based IDS	- Measurement of electric CAN signals	- It does not consider the context-aware fingerprint update (e.g., electric CAN signals are influenced by vehicle's battery level)
	Kneib et al. [71]		- Measurement of electric CAN signals - It supports the context-aware fingerprint update	- It should be verified in various vehicle driving environments.

a telematics ECU, might be involved in more than two group domains as shown in the attack on a 2014 Jeep Cherokee in [15].

2) Centralized Node-Based Approach: Alternatively, some works have adopted centralized node-based message authentication [81]–[84]. In this approach, every ECU establishes their own authentication key with a centralized ECU to protect automotive CAN from masquerade attacks carried out via compromised ECUs (i.e., Type B adversary). The centralized ECU is in charge of message authentication.

Fig. 7(b) shows the architecture of the centralized node-based approach and an example of message authentication protocol flow. In general, the centralized node-based message authentication protocol is composed of four steps. First, all ECUs share a distinct key with the centralized ECU. Second, a sender ECU computes only one authentication tag for its own CAN packet by using the key shared with the centralized ECU, and then transmits it along with the CAN packet. Third, the centralized ECU verifies the authentication tag using the key shared by the sender ECU. Lastly, the centralized ECU has an alert step informing other ECUs of the verification results. In this survey paper, the last step is referred to as the informing process of verification results. There are two main methods behind the informing process: the hash chain-based method [109] and the electronic signal-based method, which generates an error frame to invalidate CAN messages that have been used for attacks.

B. Transmission of Authentication Tags

To provide message authentication, a CAN packet should be transmitted along with the corresponding authentication tag. Since the size of the CAN data field (64 bits) is constrained, three methods for transmission of an authentication tag have been proposed. For explanatory purposes, assume that the size of an authentication tag is l bits ($l \leq 64$).

1) Basic Approach: In this approach, the CAN data field is divided into two parts to include both CAN data (64 bits - l bits) and an authentication tag (l bits). Since almost all messages in a CAN database are designed to use all 64 bits of the CAN data field, they should be adjusted from 64 bits to (64 - l) bits before applying this approach. The works proposed in [74], [76], [78], [80], [82], [84] employ this approach.

2) Extended ID-Based Approach: The Extended ID field defined in the CAN 2.0 B specification is usually used for industrial CAN bus in which many nodes exist. However, this field can also be used to transfer authentication tags. If the Extended ID field (18 bits) is set for transmission of authentication tags, a l-bit authentication tag can be divided into 18 bits and $l - 18$ bits in the Extended ID field and the CAN data field, respectively (l is assumed to be greater than 18). Like the basic approach, almost all messages defined in a CAN database should be adjusted from 64 bits to (64 - $l+18$) bits before applying this approach. This approach is detailed in [6], [75], [79].

3) Additional Can Packet-Based Approach: In this approach, an additional CAN packet is used for the transmission of an authentication tag. [72], [73], [81], [83] presented the main discussions regarding this approach. While this approach does not require any adjustments of CAN messages defined a CAN database, it causes additional authentication delay and network overhead because every CAN packet needs another CAN packet for message authentication. To minimize communication overhead on automotive CAN, [73], [77] proposed a selective message authentication method to be used exclusively on safety-critical CAN packets, such as engine control-related packets, and not on any other CAN packets. However, even though the selective message authentication can reduce communication overhead on automotive CAN, the delay originating from the transmission of an additional packet is an inevitable drawback.

Table V shows the summary of existing authentication methods we reviewed.

VII. Post-Protection Methods

Attack identification and secure patch countermeasures as post-protection methods can actually effectively bolster security of automotive CAN. This can be achieved, for example, via a data logging system or an attestation service that is considered as an important part of a digital investigation method. Once the compromised ECU is discovered, secure patches are initiated that will update the compromised ECU's firmware back to its uninfected state. This section discusses attack identification and secure patch for automotive CAN as follows.

A. Attack Identification

Even though significant effort has been made to protect automotive CAN from malicious actors, it is impossible to block all types of unknown attacks. As such, it is pertinent to identify a new attack and a compromised ECU, which could be accomplished via data logging of CAN traffic. Since data logging alone is not sufficient to identify a compromised node perfectly in cases when an attack exploits a stealth feature like Stuxnet [110], attestation could also be employed to identify a compromised node.

1) Data Logging: In [85], an information set—CAN ID, packet arrival time, and domain ID (e.g., powertrain, comfort, and infotainment)—was defined for a data logging system. However, this information is not sufficient to identify the source ECU from which an attack was actually performed on automotive CAN because this data set could easily be contaminated by an attack attempt. In other words, this set does not contain any inimitable data (i.e., digital fingerprints) like information about electric CAN signals. For example, if message flooding attacks using the CAN ID of 0×00 was performed by a compromised ECU, information about the CAN ID and packet arrival time stored in the logging system could be used to analyze the features (e.g., frequency of attack packets) of the message flooding attacks. However, it is still difficult for the logging system to identify which ECU has been exploited for message flooding attacks.

Identification of a compromised ECU is one important and necessary requirement in a fast and efficient response to an attack, such as an isolation method or dispatching of a security patch to the compromised ECU. To meet this requirement, a voltage-based ECU identification that would fingerprint the CAN transceiver of ECUs via voltage measurements was proposed in [86]. Through experiments on two real vehicles, Cho *et al.* [86] demonstrated that their logging system can identify a compromised ECU with a low false identification rate of 0.2%.

Recently, trusted execution environment (TEE)-based logging system was proposed in [87] to protect the logging information from forgery attacks.

2) Attestation: Attestation as an interactive protocol between a verifier and a prover provides a way for the verifier to validate the prover's software. This attestation protocol can be used to identify compromised nodes, and it is based on a challenge and response flow [88]. For attestation of all ECUs on automotive CAN, the role of the verifier is assigned to an ECU or an additional special device to validate the integrity of firmware installed on all ECUs. Thus when a vehicle is manufactured, an attestation code with a unique attestation key should be stored in all ECUs along with a designated verifier ECU. The attestation process for ECU firmware could be initialized when the ignition is started, or it could be performed dynamically whenever verification of ECU firmware is required.

B. Secure Patch

1) Firmware Over the Air (FOTA): The final defense system that we review in this paper is Firmware Over The Air, or FOTA. FOTA is used to add new features or to repair

TABLE V
SUMMARY OF EXISTING MESSAGE AUTHENTICATION METHODS

	Research work	Main Technology	Contribution	Drawback
Group key-based message authentication	Woo et al. [6] Wang et al. [72] Nürnberger et al. [73] Radu et al. [74] Kang et al. [75] Bella et al. [76] Mun et al. [77] Palaniswamy et al. [78] Youn et al. [79]	Group-based key sharing	- All ECUs share one group key for message authentication.	- It is not secure against a Type B adversary. (A group key could be compromised by a Type B adversary for masquerade attacks.)
	Schmandt et al. [80]	Domain-based key sharing	- All ECUs existing on same domain share one group key for message authentication.	- It is not secure against a Type B adversary. (A group key could be compromised by a Type B adversary for masquerade attacks.)
Centralized node-based message authentication	Groza et al. [81]	Hash chain-based authentication	- A centralized node verifies all authentication tags. - If an authentication tag is verified successfully, a centralized node transmits the verification result via a hash-chain method. - It is secure against masquerade attacks from a Type B adversary.	- There is a potential risk regarding a single point of failure on the centralized node. - An authentication delay results from the hash-chain method
	Kurachi et al. [82]	Error frame generation	- A centralized node verifies all authentication tags. - If verification of an authentication tag fails, the centralized node generates an error-frame. - It is secure against masquerade attacks by a Type B adversary.	- There is a potential risk regarding a single point of failure on the centralized node. - To generate an error-frame, modification of a CAN-controller is needed, though the use of a modified CAN-controller does not follow the CAN standard
	Wang et al. [83]	Dominant-bit generation	- A centralized node verifies all authentication tags. - If verification of an authentication tag fails, the centralized node generates a dominant-bit. - It is secure against masquerade attacks by a Type B adversary.	- There is a potential risk regarding a single point of failure on the centralized node. - To generate a dominant-bit, modification of a CAN-controller is needed, though the use of a modified CAN-controller does not follow the CAN standard
	Jo et al. [84]	Hash chain-based authentication	- A centralized node verifies all authentication tags. - If verification of an authentication tag fails, a centralized node informs the verification failure via a hash-chain method. - It is secure against masquerade attacks from a Type B adversary.	- There is a potential risk regarding a single point of failure on the centralized node. - An authentication delay results from the hash-chain method and the pre-defined waiting time.

vulnerabilities of ECUs related to attacks on automotive CAN by installing new firmware remotely. In [89], the authors define four security requirements for secure firmware update: data integrity, data authentication, data confidentiality, and data freshness. To meet theses requirements, the existing FOTA systems adopt a symmetric key-based approach [90], [91] or a hybrid approach [89], [92], in which a symmetric key cryptosystem and an asymmetric key cryptosystem are combined.

The symmetric key-based approach proposed in [90], [91] manages a number of symmetric keys shared with all vehicles and thus needs a key management process, which causes overhead. In [89], a digital signature algorithm is used to verify new firmware transferred by FOTA systems. However, it is a challenge for ECUs to verify the digital signature of firmware due to limited computing power and capacity. Owing to this limitation, the work of [92] was designed to verify all ECU firmware with a special node called a diagnostic tester (DT) that is equipped with high computational power. Once verification of new firmware is successfully performed by the special node, the verification information of the firmware will be reprocessed by a symmetric key-based approach, such as message authentication code (MAC). It should be noted that the symmetric key is to be shared between the special node and all ECUs in advance.

To provide secure management of the symmetric key used for firmware updates, TPM-based key management is considered one of the most promising approaches because TPM makes it difficult for adversaries to access the symmetric key (e.g., firmware encryption and authentication). For example, in [93], all ECUs contain trusted hardware to verify new firmware, and secret values stored in the trusted hardware are used for firmware decryption and authentication. Still, it is

TABLE VI
SUMMARY OF EXISTING POST-PROTECTION METHODS

	Research work	Main Technology	Contribution	Drawback
Data logging	Nilsson et al. [85]	CAN packets logging	- It defines a set of logging information: CAN-ID, packet arrival time and domain-ID.	- It is difficult for the logging system to be used to identify which ECU has been exploited for attacks. - It is not secure against a Type B adversary. (Logging data could be compromised a Type B adversary)
	Cho et al. [86]	Electronic CAN signals logging	- It measures electronic signals from the CAN-transceiver of all ECUs for ECU identification.	- If there are multiple ECUs that are assigned to send messages with the same ID, it is difficult to identify ECU well. - It is not secure against a Type B adversary. (Logging data could be compromised a Type B adversary)
	Lee et al. [87]	TPM-based CAN packets logging	- A trusted automotive data recording system that ensures data integrity, continuity, and non-repudiation	- Overhead for the integrity-ensured data generation
Attestation	Van Bulck et al. [88]	Symmetric key-based attestation	- Attestation using a challenge and responses protocol - TPM is used to protect the symmetric key used for attestation.	- It is not backward compatible. (TPM should be integrated with all legacy ECUs.)
Secure patch	Nilsson et al. [89]	Hybrid key-based FOTA	- It define four security requirements for secure firmware update: data integrity, data authentication, data confidentiality, and data freshness.	- it is a challenge for ECUs to verify the digital signature of firmware due to limited computing power and capacity.
	Mahmud et al. [90]	Symmetric key-based FOTA	- It introduces a key exchange mechanism and a software upload process.	- It is not secure against a Type B adversary. (A symmetric key for FOTA could be compromised.)
	Mansour et al. [91]	Symmetric key-based FOTA	- It introduces the two secure protocols between a vehicle and a manufacturer for vehicle diagnosis and software update.	
	Steger et al. [92]	Hybrid key-based FOTA	- Parallel and partial SW updates is designed to increase the efficiency. - TPM is used to protect the symmetric key used for FOTA.	- It is not backward compatible. (TPM should be integrated with all legacy ECUs.)
	Petri et al. [93]	TPM-based FOTA	- It adopts different levels of TPM implementations; full TPMs and lightweight TPMs	
	Kuppusamy et al. [94]	Secret key sharing-based FOTA	- Six design principles of FOTA are introduced to ensure resilience in the event of an attack on FOTA services.	- It needs a key management process for compromise-resilience.

difficult to apply the TPM-based key management for firmware updates to all legacy ECUs.

Recently, in the work [94], six design principles of secure FOTA were introduced to ensure resilience in the event of an attack on FOTA services. These are the use of multiple keys for digital signatures, setting a threshold number for performing a digital signature algorithm, key revocation, the use of offline keys that are not transmitted over the air, resilience against key compromise attacks, the use of a diversity of signing, and hashing algorithms.

Table VI shows the summary of existing post-protection methods we reviewed.

VIII. RESEARCH DIRECTIONS FOR SECURITY OF AUTONOMOUS VEHICLES

With increasing consumer demand for active-safety functions (e.g., advanced driver assistance systems (ADAS)) and multimedia functions in autonomous vehicles, next-generation in-vehicle networks require much broader network bandwidth. For example, there are new technologies for in-vehicle networks such as CAN with a flexible data rate (CAN FD) [111] and automotive Ethernet [112]. According to [113], it is predicted that automotive Ethernet will be present in up to 40% of new vehicle models in the near future.

In this regard, based on our knowledge of automotive CAN protection methods as reviewed in Section V, VI, and VII, we introduce open security research directions for next-generation in-vehicle networks.

A. Preventative Protection System

1) Automated in-Vehicle Network Fuzzer: To automatically find ECU vulnerabilities, a systematic and automated in-vehicle network fuzzing technique for next-generation

in-vehicle networks is needed, like the CAN-based fuzzing works that consider the meaning of data payload [97]–[99] and network configurations [100].

2) Dynamic Configuration of Access Control: A number of wireless connections will also soon be applied into autonomous vehicles to support communications like vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) technologies. In order to prevent cyber attacks on these communication systems, it is necessary to implement an access control that can be dynamically configured according to vehicle's environmental conditions, such as location, time, or vehicle states.

For constructing dynamic access control, software-defined networking (SDN) could be a useful building block for next-generation in-vehicle networks as it offers greater flexibility and in-vehicle network resource management by providing programmable network functions. For example, SDN controllers that enable a dynamic access control as presented in [114] could be used to protect safety-critical domains of in-vehicle networks.

B. Intrusion Detection System

Recently, there are increasing reports of attacks on vehicle sensors [115], [116], light imaging detection and ranging (LiDAR) [117], GPS [118], and advanced driving assistance systems (ADAS) [119]. Since these sensor-based attacks could cause incorrect information to be shared over next-generation in-vehicle networks or inaccurate vehicle control, these attacks must be detectable by advanced intrusion detection systems that are robust against adversarial examples [120]–[122] and leverages on sensor fusion [123].

C. Message Authentication Protocol

Scalable service-oriented middleware over IP (SOME/IP) is an emerging automotive middleware protocol standardized by AUTOSAR. It is designed to support next-generation in-vehicle communication needs, like high transmission data rates and low transportation overhead, and offers two communication types, i.e., request/response and publish/subscribe.

However, SOME/IP communications lack security functionality, which can cause messages transmitted across the network to be modified, blocked, or destroyed by malicious attacks. Even though there is extensive literature on authentication works, the existing methods appear not to fit the peculiarities of next-generation in-vehicle networks, especially for SOME/IP middleware. For example, transport layer security (TLS) does not support multicast communications, which is the main communication type used in next-generation in-vehicle networks. Thus, a novel security protocol must be designed to protect new types of in-vehicle communication systems, like SOME/IP communications [124]–[126].

D. Post-Protection System

1) Identification of Compromised ECUs: Existing security countermeasures for in-vehicle networks cannot locate which ECU is compromised because current in-vehicle networks are based on broadcast/multicast communications, and message sender identities (e.g, CAN ID) can be modified by an adversary. As such, identification of compromised ECUs is one of the security requirements for next-generation in-vehicle networks as this can provide isolation, a secure patch, and so on.

Even though the work [86] was designed to identify which ECU the attack is actually mounted by using the electrical signal characteristics of ECUs, it does not consider environmental factors that can affect electrical signal patternsand could be circumvented by [127]. This means that it cannot be applied to emerging in-vehicle networks like automotive Ethernet. Therefore, in order to identify a compromised ECU, it is necessary to improve on the work [86] and to design a new identification approach that can be used for next-generation in-vehicle networks.

2) Mitigator for DoS Attacks: There is no perfect prevention method for dealing with DoS attacks on in-vehicle networks. While message authentication can be a solution for handling masquerade attacks attempted by Type B adversaries, they can neither prevent nor mitigate DoS attacks. Even though a well-constructed firewall system and network isolation could serve as formidable prevention mechanisms against DoS attacks, these could still be circumvented by an unknown type of attack due to the development of connected cars and autonomous car environments that could open up new attack paths for adversaries to exploit.

Hence, it is crucial to design mitigation methods like [128] to withstand DoS attacks for the robustness of next-generation in-vehicle networks.

IX. Conclusion

In this paper, we offered an overview of the literature on automotive CAN security protocol and methods with a focus on attacks on automotive CAN and corresponding countermeasures. We also presented a comprehensive attack model (i.e., Type A and Type B adversaries) and three attack scenarios (i.e., masquerade attack, DoS attack, and combined attack). Through the attack models and scenarios, we evaluated existing countermeasures and considered the limitations of each mechanism and possible alternatives to bolster these.

Based on our findings, the strongest attack on automotive CAN is a combined attack by a Type B adversary because this type of adversary can perform both a masquerade attack and a DoS attack simultaneously after successfully compromising one ECU. However, even now, there is no perfect solution that can effectively block all potential damage of a combined attack by a Type B adversary in terms of security or practicability. Thus, we carefully envision open security research directions for the next generation in-vehicle networks of autonomous vehicles. It is our hope this work will contribute to the understanding of state-of-the-art security research on automotive CAN and open up new research opportunities for the emerging in-vehicle network research.

References

[1] B. van Arem, C. J. G. van Driel, and R. Visser, "The impact of cooperative adaptive cruise control on traffic-flow characteristics," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 4, pp. 429–436, Dec. 2006.

[2] K. Koscher *et al.*, "Experimental security analysis of a modern automobile," in *Proc. IEEE Symp. Secur. Privacy*, May 2010, pp. 447–462.

[3] C. Valasek and C. Miller, "Adventures in automotive networks and control units," *Def Con*, vol. 21, nos. 260–264, pp. 15–31, 2013.

[4] S. Checkoway *et al.*, "Comprehensive experimental analyses of automotive attack surfaces," in *Proc. 20th USENIX Secur. Symp.* San Francisco, CA, USA: USENIX Association, 2011, pp. 447–462.

[5] E. Eric, "An introduction to the CANard toolkit," in *Proc. Black Hat Asia*, 2015, pp. 1–8.

[6] S. Woo, H. J. Jo, and D. H. Lee, "A practical wireless attack on the connected car and security protocol for in-vehicle CAN," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 993–1006, Apr. 2015.

[7] Y. Lee, S. Woo, J. Lee, Y. Song, H. Moon, and D. H. Lee, "Enhanced Android app-repackaging attack on in-vehicle network," *Wireless Commun. Mobile Comput.*, vol. 2019, pp. 1–13, Feb. 2019.

[8] A. K. Mandal, F. Panarotto, A. Cortesi, P. Ferrara, and F. Spoto, "Static analysis of Android auto infotainment and on-board diagnostics II apps," *Softw., Pract. Exper.*, vol. 49, no. 7, pp. 1131–1161, May 2019.

[9] H. Wen, Q. Zhao, Q. A. Chen, and Z. Lin, "Automated cross-platform reverse engineering of CAN bus commands from mobile apps," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2020, pp. 1–17.

[10] H. Wen, Z. Lin, and Q. A. Chen, "Plug-N-pwned: Comprehensive vulnerability analysis of OBD-II dongles as a new over-the-air attack surface in automotive IoT," in *Proc. 29th USENIX Secur. Symp.* Boston, MA, USA: USENIX Association, Aug. 2020, pp. 949–965.

[11] I. Foster, A. Prudhomme, K. Koscher, and S. Savage, "Fast and vulnerable: A story of telematic failures," in *Proc. 9th USENIX Workshop Offensive Technol. (WOOT)*. Washington, DC, USA: USENIX Association, 2015, pp. 1–9.

[12] H. J. Jo, W. Choi, S. Y. Na, S. Woo, and D. H. Lee, "Vulnerabilities of Android OS-based telematics system," *Wireless Pers. Commun.*, vol. 92, pp. 1–20, Feb. 2016.

[13] S. Mazloom, M. Rezaeirad, A. Hunter, and D. McCoy, "A security analysis of an in-vehicle infotainment and app platform," in *Proc. 10th USENIX Workshop Offensive Technol. (WOOT)*. Austin, TX, USA: USENIX Association, 2016, pp. 1–12.

[14] S. Nie, L. Liu, and Y. Du, "Free-fall: Hacking tesla from wireless to CAN bus," *Blackhat USA*, vol. 25, pp. 1–16, Jul. 2017.

[15] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, vol. 2015, p. 9, Aug. 2015.

[16] C. Miller and C. Valasek, "A survey of remote automotive attack surfaces," *Black Hat USA*, vol. 2014, p. 94, Aug. 2014.

[17] S. K. M. Pesé and H. Zweck, "Hardware/software co-design of an automotive embedded firewall," SAE Tech. Paper 2017-01-1659, 2017.

[18] L. Wang and X. Liu, "NOTSA: Novel OBU with three-level security architecture for Internet of vehicles," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3548–3558, Oct. 2018.

[19] A. Humayed, F. Li, J. Lin, and B. Luo, "CANSentry: Securing CAN-based cyber-physical systems against denial and spoofing attacks," in *Computer Security—ESORICS 2020*. Guildford, U.K.: Springer, 2020, pp. 153–173.

[20] M. A. Taie, E. M. Moawad, M. Diab, and M. ElHelw, "Remote diagnosis, maintenance and prognosis for advanced driver assistance systems using machine learning algorithms," *SAE Int. J. Passenger Cars-Electron. Electr. Syst.*, vol. 9, no. 1, pp. 114–122, Apr. 2016.

[21] T. Zhang, H. Antunes, and S. Aggarwal, "Defending connected vehicles against malware: Challenges and a solution framework," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 10–21, Feb. 2014.

[22] Q. Luo and J. Liu, "Wireless telematics systems in emerging intelligent and connected vehicles: Threats and solutions," *IEEE Wireless Commun.*, vol. 25, no. 6, pp. 113–119, Dec. 2018.

[23] G. Loukas, E. Karapistoli, E. Panaousis, P. Sarigiannidis, A. Bezemskij, and T. Vuong, "A taxonomy and survey of cyber-physical intrusion detection approaches for vehicles," *Ad Hoc Netw.*, vol. 84, pp. 124–147, Mar. 2019.

[24] O. Y. Al-Jarrah, C. Maple, M. Dianati, D. Oxtoby, and A. Mouzakitis, "Intrusion detection systems for intra-vehicle networks: A review," *IEEE Access*, vol. 7, pp. 21266–21289, 2019.

[25] W. Wu *et al.*, "A survey of intrusion detection for in-vehicle networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 919–933, Mar. 2020.

[26] C. Young, J. Zambreno, H. Olufowobi, and G. Bloom, "Survey of automotive controller area network intrusion detection systems," *IEEE Des. Test*, vol. 36, no. 6, pp. 48–55, Dec. 2019.

[27] S. K. Khan, N. Shiwakoti, P. Stasinopoulos, and Y. Chen, "Cyber-attacks in the next-generation cars, mitigation techniques, anticipated readiness and future directions," *Accident Anal. Prevention*, vol. 148, Dec. 2020, Art. no. 105837.

[28] S. Halder, A. Ghosal, and M. Conti, "Secure over-the-air software updates in connected vehicles: A survey," *Comput. Netw.*, vol. 178, Sep. 2020, Art. no. 107343.

[29] J. Sun, S. Iqbal, N. S. Arabi, and M. Zulkernine, "A classification of attacks to in-vehicle components (IVCs)," *Veh. Commun.*, vol. 25, Oct. 2020, Art. no. 100253.

[30] S. Bayer and A. Ptok, "Don't fuss about fuzzing: Fuzzing controllers in vehicular networks," in *Proc. 13th Int. Conf. Embedded Secur. Cars (ESCAR Eur.)*, Nov. 2015, p. 88.

[31] D. K. Oka, A. Yvard, S. Bayer, and T. Kreuzinger, "Enabling cyber security testing of automotive ECUs by adding monitoring capabilities," in *Proc. 13th Int. Conf. Embedded Secur. Cars (ESCAR Eur.)*, Nov. 2015, pp. 1–13.

[32] D. S. Fowler, J. Bryans, S. A. Shaikh, and P. Wooderson, "Fuzz testing for automotive cyber-security," in *Proc. 48th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. Workshops (DSN-W)*, Jun. 2018, pp. 239–246.

[33] L. Yu, J. Deng, R. R. Brooks, and S. B. Yun, "Automobile ECU design to avoid data tampering," in *Proc. 10th Annu. Cyber Inf. Secur. Res. Conf.*, New York, NY, USA, Apr. 2015, pp. 10:1–10:4.

[34] D. Hély, F. Bancel, M.-L. Flottes, and B. Rouzeyre, "Securing scan control in crypto chips," *J. Electron. Test.*, vol. 23, no. 5, pp. 457–464, Nov. 2007.

[35] K. Rosenfeld and R. Karri, "Attacks and defenses for JTAG," *IEEE Design Test Comput.*, vol. 27, pp. 36–47, Jan. 2010.

[36] F. Novak and A. Biasizzo, "Security extension for IEEE Std 1149.1," *J. Electron. Test.*, vol. 22, no. 3, pp. 301–303, Jun. 2006.

[37] L. Pierce and S. Tragoudas, "Enhanced secure architecture for joint action test group systems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 21, no. 7, pp. 1342–1345, Jul. 2013.

[38] P. Mundhenk *et al.*, "Security in automotive networks: Lightweight authentication and authorization," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 22, no. 2, pp. 25:1–25:27, Mar. 2017.

[39] K. Han, A. Weimerskirch, and K. G. Shin, "A practical solution to achieve real-time performance in the automotive network by randomizing frame identifier," in *Proc. 13th Int. Conf. Embedded Secur. Cars (ESCAR)*, Nov. 2015, pp. 13–29.

[40] A. Humayed and B. Luo, "Using ID-hopping to defend against targeted DoS on CAN," in *Proc. 1st Int. Workshop Safe Control Connected Auto. Vehicles*, New York, NY, USA, Apr. 2017, pp. 19–26.

[41] W. Wu *et al.*, "IDH-CAN: A hardware-based ID hopping CAN mechanism with enhanced security for automotive real-time applications," *IEEE Access*, vol. 6, pp. 54607–54623, 2018.

[42] S. Woo, D. Moon, T.-Y. Youn, Y. Lee, and Y. Kim, "CAN ID shuffling technique (CIST): Moving target defense strategy for protecting in-vehicle CAN," *IEEE Access*, vol. 7, pp. 15521–15536, 2019.

[43] K. Cheng, Y. Bai, Y. Zhou, Y. Tang, D. Sanan, and Y. Liu, "CANeleon: Protecting CAN bus with frame ID chameleon," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7116–7130, Jul. 2020.

[44] B. Groza, L. Popa, and P.-S. Murvay, "Highly efficient authentication for CAN by identifier reallocation with ordered CMACs," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 6129–6140, Jun. 2020.

[45] A. Taylor, N. Japkowicz, and S. Leblanc, "Frequency-based anomaly detection for the automotive CAN bus," in *Proc. World Congr. Ind. Control Syst. Secur. (WCICSS)*, Dec. 2015, pp. 45–49.

[46] H. M. Song, H. R. Kim, and H. K. Kim, "Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2016, pp. 63–68.

[47] A. Tomlinson, J. Bryans, S. A. Shaikh, and H. K. Kalutarage, "Detection of automotive CAN cyber-attacks by identifying packet timing anomalies in time windows," in *Proc. 48th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. Workshops (DSN-W)*, Jun. 2018, pp. 231–238.

[48] H. Olufowobi, C. Young, J. Zambreno, and G. Bloom, "SAIDuCANT: Specification-based automotive intrusion detection using controller area network (CAN) timing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 1484–1494, Feb. 2020.

[49] C. Young, H. Olufowobi, G. Bloom, and J. Zambreno, "Automotive intrusion detection based on constant CAN message frequencies across vehicle driving modes," in *Proc. ACM Workshop Automot. Cybersecur.* New York, NY, USA: Association Computing Machinery, Mar. 2019, pp. 9–14.

[50] M. Marchetti and D. Stabili, "Anomaly detection of CAN bus messages through analysis of ID sequences," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 1577–1583.

[51] S. Katragadda, P. J. Darby, A. Roche, and R. Gottumukkala, "Detecting low-rate replay-based injection attacks on in-vehicle networks," *IEEE Access*, vol. 8, pp. 54979–54993, 2020.

[52] D. Stabili, M. Marchetti, and M. Colajanni, "Detecting attacks to internal vehicle networks through Hamming distance," in *Proc. AEIT Int. Annu. Conf.*, Sep. 2017, pp. 1–6.

[53] M. Markovitz and A. Wool, "Field classification, modeling and anomaly detection in unknown CAN bus networks," *Veh. Commun.*, vol. 9, pp. 43–52, Jul. 2017.

[54] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PLoS ONE*, vol. 11, no. 6, pp. 1–17, Jun. 2016.

[55] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Veh. Commun.*, vol. 21, pp. 1–13, Jan. 2020.

[56] A. Taylor, S. Leblanc, and N. Japkowicz, "Anomaly detection in automobile control network data with long short-term memory networks," in *Proc. IEEE Int. Conf. Data Sci. Adv. Anal. (DSAA)*, Oct. 2016, pp. 130–139.

[57] S. Longari, D. H. N. Valcarcel, M. Zago, M. Carminati, and S. Zanero, "CANnolo: An anomaly detection system based on LSTM autoencoders for controller area network," *IEEE Trans. Netw. Service Manage.*, early access, Nov. 18, 2020, doi: 10.1109/TNSM.2020.3038991.

[58] M. D. Hossain, H. Inoue, H. Ochiai, D. Fall, and Y. Kadobayashi, "LSTM-based intrusion detection system for in-vehicle CAN bus communications," *IEEE Access*, vol. 8, pp. 185489–185502, 2020.

[59] S. Tariq, S. Lee, H. K. Kim, and S. S. Woo, "CAN-ADF: The controller area network attack detection framework," *Comput. Secur.*, vol. 94, Jul. 2020, Art. no. 101857.

[60] A. R. Wasicek, M. D. Pese, A. Weimerskirch, Y. Burakova, and K. Singh, "Context-aware intrusion detection in automotive control systems," in *Proc. 5th Int. Conf. Embedded Secur. Cars (ESCAR USA)*, Jun. 2017, pp. 21–22.

[61] M. R. Ansari, W. T. Miller, C. She, and Q. Yu, "A low-cost masquerade and replay attack detection method for CAN in automobiles," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4.

[62] M. Muter, A. Groll, and F. C. Freiling, "A structured approach to anomaly detection for in-vehicle networks," in *Proc. 6th Int. Conf. Inf. Assurance Secur.*, Aug. 2010, pp. 92–98.

[63] M. Muter and N. Asaj, "Entropy-based anomaly detection for in-vehicle networks," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2011, pp. 1110–1115.

[64] M. Marchetti, D. Stabili, A. Guido, and M. Colajanni, "Evaluation of anomaly detection for in-vehicle networks through information-theoretic algorithms," in *Proc. IEEE 2nd Int. Forum Res. Technol. Soc. Ind. Leveraging Better Tomorrow (RTSI)*, Sep. 2016, pp. 1–6.

[65] K.-T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection," in *Proc. 25th USENIX Secur. Symp.* Austin, TX, USA: USENIX Association, 2016, pp. 911–927.

[66] H. Lee, S. H. Jeong, and H. K. Kim, "OTIDS: A novel intrusion detection system for in-vehicle network by using remote frame," in *Proc. 15th Annu. Conf. Privacy, Secur. Trust (PST)*, Aug. 2017, pp. 5709–5757.

[67] P.-S. Murvay and B. Groza, "TIDAL-CAN: Differential timing based intrusion detection and localization for controller area network," *IEEE Access*, vol. 8, pp. 68895–68912, 2020.

[68] P.-S. Murvay and B. Groza, "Source identification using signal characteristics in controller area networks," *IEEE Signal Process. Lett.*, vol. 21, no. 4, pp. 395–399, Apr. 2014.

[69] W. Choi, H. J. Jo, S. Woo, J. Y. Chun, J. Park, and D. H. Lee, "Identifying ECUs using inimitable characteristics of signals in controller area networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 4757–4770, Jun. 2018.

[70] W. Choi, K. Joo, H. J. Jo, M. C. Park, and D. H. Lee, "VoltageIDS: Low-level communication characteristics for automotive intrusion detection system," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 8, pp. 2114–2129, Aug. 2018.

[71] M. Kneib, O. Schell, and C. Huth, "EASI: Edge-based sender identification on resource-constrained platforms for automotive networks," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2020, pp. 1–16.

[72] Q. Wang and S. Sawhney, "VeCure: A practical security framework to protect the CAN bus of vehicles," in *Proc. Int. Conf. Internet Things (IOT)*, Oct. 2014, pp. 13–18.

[73] S. Nürnberger and C. Rossow, "–vatiCAN–vetted, authenticated CAN bus," in *Proc. 18th Int. Conf. Cryptograph. Hardw. Embedded Syst. (CHES)*. Berlin, Germany: Springer, 2016, pp. 106–124.

[74] A.-I. Radu and F. D. Garcia, "LeiA: A lightweight authentication protocol for CAN," in *Proc. 21st Eur. Symp. Res. Comput. Secur. (ESORICS)*. New York, NY, USA: Springer-Verlag, Sep. 2016, pp. 283–300.

[75] K.-D. Kang, Y. Baek, S. Lee, and S. H. Son, "An attack-resilient source authentication protocol in controller area network," in *Proc. ACM/IEEE Symp. Archit. Netw. Commun. Syst. (ANCS)*, May 2017, pp. 109–118.

[76] G. Bella, P. Biondi, G. Costantino, and I. Matteucci, "TOUCAN: a protocol to secure controller area network," in *Proc. ACM Workshop Automot. Cybersecur.*, New York, NY, USA, Mar. 2019, pp. 3–8.

[77] H. Mun, K. Han, and D. H. Lee, "Ensuring safety and security in CAN-based automotive embedded systems: A combination of design optimization and secure communication," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7078–7091, Jul. 2020.

[78] T.-Y. Youn, Y. Lee, and S. Woo, "Practical sender authentication scheme for in-vehicle CAN with efficient key management," *IEEE Access*, vol. 8, pp. 86836–86849, 2020.

[79] B. Palaniswamy, S. Camtepe, E. Foo, and J. Pieprzyk, "An efficient authentication scheme for intra-vehicular controller area network," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3107–3122, 2020.

[80] J. Schmandt, A. T. Sherman, and N. Banerjee, "Mini-MAC: Raising the bar for vehicular security with a lightweight message authentication protocol," *Veh. Commun.*, vol. 9, pp. 188–196, Jul. 2017.

[81] B. Groza and S. Murvay, "Efficient protocols for secure broadcast in controller area networks," *IEEE Trans. Ind. Informat.*, vol. 9, no. 4, pp. 2034–2042, Nov. 2013.

[82] R. Kurachi, Y. Matsubara, H. Takada, N. Adachi, Y. Miyashita, and S. Horihata, "CaCAN-centralized authentication system in CAN (controller area network)," in *Proc. 12th Int. Conf. Embedded Secur. Cars (ESCAR Eur.)*, Nov. 2014, pp. 1–9.

[83] E. Wang, W. Xu, S. Sastry, S. Liu, and K. Zeng, "Hardware module-based message authentication in intra-vehicle networks," in *Proc. 8th Int. Conf. Cyber-Phys. Syst.*, New York, NY, USA, Apr. 2017, pp. 207–216.

[84] H. J. Jo, J. H. Kim, H.-Y. Choi, W. Choi, D. H. Lee, and I. Lee, "MAuth-CAN: Masquerade-attack-proof authentication for in-vehicle networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 2204–2218, Feb. 2020.

[85] D. K. Nilsson and U. E. Larson, "Forensic investigations of cyber attacks on automobile in-vehicle networks," in *Proc. 1st Int. Conf. Forensic Appl. Techn. Telecommun., Inf., Multimedia Workshop*, 2008, pp. 8:1–8:6.

[86] K.-T. Cho and K. G. Shin, "Viden: Attacker identification on in-vehicle networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, Oct. 2017, pp. 911–927.

[87] S. Lee, W. Choi, H. J. Jo, and D. H. Lee, "T-box: A forensics-enabled trusted automotive data recording method," *IEEE Access*, vol. 7, pp. 49738–49755, 2019.

[88] J. Van Bulck, J. T. Mühlberg, and F. Piessens, "VulCAN: Efficient component authentication and software isolation for automotive control networks," in *Proc. 33rd Annu. Comput. Secur. Appl. Conf.*, New York, NY, USA, Dec. 2017, pp. 225–237.

[89] D. K. Nilsson and U. E. Larson, "Secure firmware updates over the air in intelligent vehicles," in *Proc. ICC Workshops-IEEE Int. Conf. Commun. Workshops*, May 2008, pp. 380–384.

[90] S. M. Mahmud, S. Shanker, and I. Hossain, "Secure software upload in an intelligent vehicle via wireless communication links," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2005, pp. 588–593.

[91] K. Mansour, W. Farag, and M. ElHelw, "AiroDiag: A sophisticated tool that diagnoses and updates vehicles software over air," in *Proc. IEEE Int. Electr. Vehicle Conf.*, Mar. 2012, pp. 1–7.

[92] M. Steger *et al.*, "An efficient and secure automotive wireless software update framework," *IEEE Trans. Ind. Informat.*, vol. 14, no. 5, pp. 2181–2193, May 2018.

[93] R. Petri *et al.*, "Evaluation of lightweight TPMs for automotive software updates over the air," in *Proc. 14th Int. Conf. Embedded Secur. Cars (ESCAR Eur.)*, Nov. 2016, pp. 1–15.

[94] T. K. Kuppusamy, L. A. DeLong, and J. Cappos, "Uptane: Security and customizability of software updates for vehicles," *IEEE Veh. Technol. Mag.*, vol. 13, no. 1, pp. 66–73, Mar. 2018.

[95] *Recommended Practice for Pass-Thru Vehicle Programming* SAE Standard, Society of Automotive Engineers, 2004.

[96] K.-T. Cho and K. G. Shin, "Error handling of in-vehicle networks makes them vulnerable," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, Oct. 2016, pp. 1044–1055.
[97] M. Marchetti and D. Stabili, "READ: Reverse engineering of automotive data frames," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 4, pp. 1083–1097, Apr. 2019.
[98] M. D. Pesé, T. Stacer, C. A. Campos, E. Newberry, D. Chen, and K. G. Shin, "LibreCAN: Automated CAN message translator," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 2283–2300.
[99] D. Frassinelli, S. Park, and S. Nurnberger, "Automated reverse engineering and privacy analysis of modern cars," in *Proc. IEEE Symp. Secur. Privacy (IEEE S P)*. Los Alamitos, CA, USA: IEEE Computer Society, May 2020, pp. 1184–1198.
[100] S. Kulandaivel, T. Goyal, A. K. Agrawal, and V. Sekar, "CANvas: Fast and inexpensive automotive network mapping," in *Proc. 28th USENIX Secur. Symp.* Santa Clara, CA, USA: USENIX Association, Aug. 2019, pp. 389–405.
[101] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.
[102] C. E. Everett and D. McCoy, "OCTANE (open car testbed and network experiments): Bringing cyber-physical security research to researchers and students," in *Proc. 6th Workshop Cyber Secur. Exp. Test*. Washington, DC, USA: USENIX Association, 2013, pp. 1–8.
[103] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 15:1–15:58, 2009.
[104] S. Jana and S. K. Kasera, "On fast and accurate detection of unauthorized wireless access points using clock skews," *IEEE Trans. Mobile Comput.*, vol. 9, no. 3, pp. 449–462, Mar. 2010.
[105] S. U. Sagong, X. Ying, A. Clark, L. Bushnell, and R. Poovendran, "Cloaking the clock: Emulating clock skew in controller area networks," in *Proc. ACM/IEEE 9th Int. Conf. Cyber-Phys. Syst. (ICCPS)*. Piscataway, NJ, USA: IEEE Press, Apr. 2018, pp. 32–42.
[106] X. Ying, S. U. Sagong, A. Clark, L. Bushnell, and R. Poovendran, "Shape of the cloak: Formal analysis of clock skew-based intrusion detection system in controller area networks," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 9, pp. 2300–2314, Sep. 2019.
[107] D. Paret and R. Riesco, *Multiplexed Networks for Embedded Systems*. Hoboken, NJ, USA: Wiley, 2007.
[108] B. Groza and P.-S. Murvay, "Security solutions for the controller area network: Bringing authentication to in-vehicle networks," *IEEE Veh. Technol. Mag.*, vol. 13, no. 1, pp. 40–47, Mar. 2018.
[109] A. Perrig, R. Canetti, J. D. Tygar, and D. Song, "The TESLA broadcast authentication protocol," *RSA CryptoBytes*, vol. 5, no. 2, pp. 2–13, Nov. 2002.
[110] R. Langner, "StuxNet: Dissecting a cyberwarfare weapon," *IEEE Secur. Privacy Mag.*, vol. 9, no. 3, pp. 49–51, May 2011.
[111] *CAN With Flexible Data-Rate Specification, Version 1.0*, CAN-FD Standard, Robert Bosch GmbH, 2012.
[112] P. Hank, T. Suermann, and S. Müller, "Automotive Ethernet, a holistic approach for a next generation in-vehicle networking standard," in *Advanced Microsystems for Automotive Applications 2012*. Berlin, Germany: Springer, 2012, pp. 79–89.
[113] *Ethernet In-Vehicle Networking to Feature in 40% of Vehicles Shipping Globally by 2020*, ABI Res., New York, NY, USA, 2014.
[114] M. Rumez, A. Duda, P. Grunder, R. Kriesten, and E. Sax, "Integration of attribute-based access control into automotive architectures," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 1916–1922.
[115] I. Rouf *et al.*, "Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study," in *Proc. 19th USENIX Secur. Symp.* Berkeley, CA, USA: USENIX Association, 2010, pp. 323–338.
[116] X. W. C. Yan and J. Liu, "Can you trust autonomous vehicles: Contactless attacks against sensors of self-driving vehicle," presented at the DEFCON, Boston, MA, USA, Aug. 2016.
[117] J. Petit, B. Stottelaar, M. Feiri, and F. Kargl, "Remote attacks on automated vehicles sensors: Experiments on camera and LiDAR," *Black Hat Eur.*, vol. 11, p. 995, Nov. 2015.
[118] J. Su, J. He, P. Cheng, and J. Chen, "A stealthy GPS spoofing strategy for manipulating the trajectory of an unmanned aerial vehicle," *IFAC-PapersOnLine*, vol. 49, no. 22, pp. 291–296, Sep. 2016.
[119] B. Nassi, Y. Mirsky, D. Nassi, R. Ben-Netanel, O. Drokin, and Y. Elovici, "Phantom of the ADAS: Securing advanced driver-assistance systems from split-second phantom attacks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.* New York, NY, USA: Association Computing Machinery, 2020, pp. 293–308.
[120] J. Lu, T. Issaranon, and D. Forsyth, "SafetyNet: Detecting and rejecting adversarial examples robustly," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 446–454.
[121] S. Ma, Y. Liu, G. Tao, W.-C. Lee, and X. Zhang, "NIC: Detecting adversarial samples with neural network invariant checking," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2019, pp. 1–15.
[122] S. Jeong, B. Jeon, B. Chung, and H. K. Kim, "Convolutional neural network-based intrusion detection system for AVTP streams in automotive Ethernet-based networks," *Veh. Commun.*, vol. 29, Jun. 2021, Art. no. 100338.
[123] B. Ao, Y. Wang, L. Yu, R. R. Brooks, and S. S. Iyengar, "On precision bound of distributed fault-tolerant sensor fusion algorithms," *ACM Comput. Surv.*, vol. 49, no. 1, pp. 1–23, Jul. 2016.
[124] M. Iorio, A. Buttiglieri, M. Reineri, F. Risso, R. Sisto, and F. Valenza, "Protecting in-vehicle services: Security-enabled SOME/IP middleware," *IEEE Veh. Technol. Mag.*, vol. 15, no. 3, pp. 77–85, Sep. 2020.
[125] M. Iorio, M. Reineri, F. Risso, R. Sisto, and F. Valenza, "Securing SOME/IP for in-vehicle service protection," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 13450–13466, Nov. 2020.
[126] A. Bhat, S. Samii, and R. R. Rajkumar, "Fault-tolerance support for adaptive AUTOSAR platforms using SOME/IP," in *Proc. IEEE 26th Int. Conf. Embedded Real-Time Comput. Syst. Appl. (RTCSA)*, Aug. 2020, pp. 1–6.
[127] R. Bhatia, V. Kumar, K. Serag, Z. B. Celik, M. Payer, and D. Xu, "Evading voltage-based intrusion detection on automotive CAN," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, San Diego, CA, USA, 2021, pp. 1–17.
[128] H.-Y. Choi, A. L. King, and I. Lee, "Making DDS really real-time with openflow," in *Proc. 13th Int. Conf. Embedded Softw.* New York, NY, USA: Association Computing Machinery, Oct. 2016, pp. 1–10.

Hyo Jin Jo received the B.S. degree in industrial engineering and the Ph.D. degree in information security from Korea University, Seoul, South Korea, in 2009 and 2016, respectively. From 2016 to 2018, he was a Post-Doctoral Researcher with the Department of Computer and Information Systems, University of Pennsylvania, Philadelphia, PA, USA. He is currently an Assistant Professor with the School of Software, Soongsil University, Seoul. His research interests include applied cryptography, security and privacy for ad-hoc networks, the and IoT/CPS security.

Wonsuk Choi received the B.S. degree in mathematics from the University of Seoul, Seoul, South Korea, in 2008, and the M.S. and Ph.D. degrees in information security from Korea University, Seoul, in 2013 and 2018, respectively. He was a Post-Doctoral Researcher with the Graduate School of Information Security, Korea University. In 2020, he joined the Division of IT Convergence Engineering, Hansung University, Seoul, as an Assistant Professor. His research interests include security for body area networks, usable security, applied cryptography, and smart car security.