Evading Voltage-Based Intrusion Detection on Automotive CAN

Rohit Bhatia*, Vireshwar Kumar†, Khaled Serag*, Z. Berkay Celik*, Mathias Payer‡ and Dongyan Xu*
*Purdue University, {bhatia13, kserag, zcelik, dxu}@purdue.edu
†Indian Institute of Technology Delhi, viresh@cse.iitd.ac.in
‡EPFL, mathias.payer@nebelwelt.net

***Abstract*—The controller area network (CAN) is widely adopted in modern automobiles to enable communications among in-vehicle electronic control units (ECUs). Lacking mainstream network security capabilities due to resource constraints, the CAN is susceptible to the ECU masquerade attack in which a compromised (attacker) ECU impersonates an uncompromised (victim) ECU and spoofs the latter's CAN messages. A cost-effective state-of-the-art defense against such attacks is the CAN bus voltage-based intrusion detection system (VIDS), which identifies the source of each message using its voltage fingerprint on the bus. Since the voltage fingerprint emanates from an ECU's hardware characteristics, an attacker ECU by itself cannot controllably modify it. As such, VIDS has been proved effective in detecting masquerade attacks that each involve a single attacker.**

In this paper, we discover a novel voltage corruption tactic that leverages the capabilities of two compromised ECUs (i.e., an attacker ECU working in tandem with an accomplice ECU) to corrupt the bus voltages recorded by the VIDS. By exploiting this tactic along with the fundamental deficiencies of the CAN protocol, we propose a novel masquerade attack called DUET, which evades all existing VIDS irrespective of the features and classification algorithms employed in them. DUET follows a two-stage attack strategy to first manipulate a victim ECU's voltage fingerprint during VIDS retraining mode, and then impersonate the manipulated fingerprint during VIDS operation mode. Our evaluation of DUET on real CAN buses (including three in two real cars) demonstrates an impersonation success rate of at least 90% in evading two state-of-the-art VIDS.

Finally, to mitigate ECU masquerade attacks, we advocate the development of cost-effective defenses that break away from the "attack vs. IDS" arms race. We propose a lightweight defense called RAID, which enables each ECU to make protocol-compatible modifications in its frame format generating a unique *dialect* (spoken by ECUs) during VIDS retraining mode. RAID prevents corruption of ECUs' voltage fingerprints, and re-enables VIDS to detect all ECU masquerade attacks including DUET.

I. INTRODUCTION

Controller area network (CAN) is a wired broadcast network widely utilized in modern automobiles to enable real-time communication among electronic control units (ECUs). For providing in-vehicle infotainment, an increasing number of ECUs are being devised to connect to other IoT networks/nodes through USB, cellular, Bluetooth, and WiFi connections [48]. Unfortunately, these interfaces provide various attack surfaces making these ECUs vulnerable to a remote adversary [6], [31], [32], [35], [36]. After infiltrating CAN by compromising any of these ECUs, the adversary can carry out a variety of attacks [24], [32], [35], [36], [49] against other ECUs since the CAN lacks mainstream network security capabilities due to resource constraints. In particular, a compromised ECU (e.g., Telematics Control Unit) may impersonate a benign ECU (e.g., Engine Control Unit) that cannot be remotely compromised, and forge the latter's CAN messages to disrupt safety-critical automotive functions (e.g., engine speed). We call such attacks *ECU masquerade attacks*.

To defend against ECU masquerade attacks, various CAN intrusion detection systems (IDS) have been proposed to detect malicious/compromised ECUs on an automobile CAN [8]. The IDS approach is the dominant CAN defense because an IDS runs as an independent entity on the CAN bus and does not burden the resource-constrained ECUs/network with non-trivial computation/communication overhead. Traditional message (anomaly)-based IDS (MIDS) [18], [34], [44] utilize features such as payload values to identify anomalies on CAN. Modern state-of-the-art CAN IDS leverage physical characteristics of ECUs, such as clock-skew (CIDS [8]) and voltage (VIDS [9], [10], [14], [22], [23]), to determine if a message is generated by a legitimate ECU. While MIDS and CIDS have been shown vulnerable to persistent attackers [8], [43], the VIDS defenses have been proven highly effective in detecting ECU masquerade attacks. Hence, the VIDS is being actively researched by the academia and industry (including Bosch [22], [23], the developer of the CAN standard [5]).

The VIDS measures the bus voltages during the transmission of each CAN message at a certain sampling frequency. The VIDS then computes the *voltage fingerprint*, which is defined as a feature vector of the measured voltage samples. Since the voltage fingerprint of an ECU and its messages fluctuate over time due to environmental factors [9], the VIDS often transitions to its *retraining mode* and learns a supervised model by mapping the voltage fingerprints of messages to their source ECUs. Then, in its *operation mode*, the VIDS employs the model to infer the source of each transmitted message on the bus. As such, the VIDS equipped with high-frequency voltage sampling virtually puts the CAN traffic under a "microscope" for high-resolution monitoring and attacker detection.

We note that due to the above design (common to all existing VIDS), an attacker faces two critical challenges in evading the VIDS: (1) Although the software of an in-vehicle ECU

Network and Distributed Systems Security (NDSS) Symposium 2021
21-25 February 2021, Virtual
ISBN 1-891562-66-5
https://dx.doi.org/10.14722/ndss.2021.23013
www.ndss-symposium.org

can be remotely compromised, it is difficult to controllably alter the hardware characteristics and hence the corresponding voltage fingerprint of the compromised ECU. (2) Since the VIDS defenses are vulnerable to training set poisoning, they employ the state-of-the-art countermeasures, e.g., message authentication codes (MACs), to "securely" record the voltage fingerprints of ECUs during the brief retraining duration [22].

In this paper, to address the first challenge, we propose a novel attack tactic called *voltage corruption* which exploits the capabilities of not one, but two compromised ECUs, i.e., a duo of *attacker* ECU (or attacker for short) and *accomplice* ECU (accomplice). In this tactic, the attacker, with the help of the accomplice, performs simultaneous transmission with a *victim* ECU (victim), superimposes its voltage samples over those of the victim, and corrupts the voltage fingerprint of the targeted victim as measured by the VIDS. Further, by exploiting the voltage corruption tactic, we propose a novel masquerade attack called DUET, which evades all existing VIDS including those employing the secure retraining mode. Different from the "lone wolf" style (i.e., by a single attacker) of existing masquerade attacks, DUET is launched against a victim ECU by the duo of attacker and accomplice that follow a two-stage training set poisoning-based attack strategy. We refer to these two stages as *voltage fingerprint manipulation* (Stage 1) and *voltage fingerprint-based impersonation* (Stage 2).

DUET first performs voltage fingerprint manipulation in which the attacker, with the help of the accomplice, carries out the voltage corruption tactic on the victim. In other words, DUET attempts to manipulate the voltage fingerprints of all victim's messages. During the retraining mode, the VIDS employs some of the manipulated messages as part of the training set and learns a *distorted* fingerprint of "victim + attacker" as the victim's fingerprint. As such, DUET successfully poisons the training set and resolves the second aforementioned challenge. We point out that the fingerprint of all ECUs on the CAN bus, except that of the victim, remain undistorted in this stage.

Thereafter, whenever the DUET duo intend to spoof a victim's message, they perform voltage fingerprint-based impersonation in which the accomplice sends the spoofed message while the attacker corrupts its voltage fingerprint using the voltage corruption tactic with the accomplice as the target. This way, the VIDS observes the distorted voltage fingerprint of "accomplice + attacker" in the spoofed message. Since the VIDS has been tricked into learning that the victim has a distorted fingerprint (generated by "victim + attacker") during its retraining mode, the VIDS classifies all distorted fingerprints including that of "accomplice + attacker" as the victim's fingerprint during its operation mode. This way, the execution of the two DUET stages leads to a stealthy, advanced persistent threat (APT)-style ECU masquerade attack.

DUET exploits deficiencies in the fundamental mechanisms of the CAN protocol (i.e., the bus arbitration and error handling), capabilities of the real-world CAN controller (i.e., "one-shot" transmission mode), and *common* characteristics of the CAN traffic (i.e., the periodicity of messages and the predictability of message contents). Hence DUET can be launched against any automobile employing the CAN. We highlight that DUET is the first practical data poisoning attack that successfully evades all state-of-the-art (training-based) VIDS, *irrespective* of the features and classification algorithms used in them as illustrated in Table I. DUET can also evade the state-of-the-art defenses employed during the secure retraining mode [20], [29], including those utilizing MACs [22].

TABLE I: Effectiveness of DUET against existing VIDS.

VIDS	Sampling Frequency	Number of Features	Anomaly Detection	Vulnerable to DUET
Viden (CCS'17) [9]	50 KS/s	6	per 8 messages	✓
VoltageIDS (TIFS'18) [10]	2.5 GS/s	64	per message	✓
Scission (CCS'18) [22]	20 MS/s	48	per message	✓
SIMPLE[1](ACSAC'19) [14]	500 KS/s	32	per message	✓
EASI (NDSS'20) [23]	2 MS/s	12	per message	✓

[1] In practice, the VIDS defenses require retraining to handle environmental factors, aging, or reconfiguration of ECUs. However, as to be discussed in Section X, in scenarios where SIMPLE can be utilized without retraining, it will not be vulnerable to DUET.

We have evaluated DUET against two representative VIDS: *Scission* [22], which analyzes the voltage fingerprint of each message and *Viden* [9], which analyzes the cumulative voltage fingerprint of eight messages. Our evaluation in real CAN environments, which include three CAN buses in two cars, shows that after corrupting voltage samples corresponding to only two bytes of the victim's payload, DUET demonstrates an impersonation *success rate* of 90% against Scission and 100% against Viden. Further, through an illustrative experiment on a real car, we demonstrate that the DUET duo successfully impersonate the Engine Control Unit of the car and exhibit anomalous revolutions per minute (RPM) values on the vehicle dashboard without raising a VIDS alarm.

To safeguard CAN, it might be possible to modify the VIDS such that it detects DUET by uncovering the distortion in the voltage fingerprint of the victim. However, the modified VIDS must find a tedious balance between detecting DUET and avoiding false alarms caused by non-malicious electromagnetic interference. In this paper, breaking away from such a trade-off and the resulting "attack vs. IDS" arms race, we propose an efficient defense, called *Ra*ndomized *I*dentifier *D*efense (RAID) which prevents (not just detects) DUET. RAID takes an orthogonal (to VIDS) approach of randomizing a part of the victim's CAN message identifier. Such randomization generates a unique *dialect* that is spoken by ECUs only during VIDS retraining mode. This dialecting ensures that the attacker will either win or lose the arbitration on the bus while attempting to transmit simultaneously with the victim, and will not be able to carry out the voltage corruption tactic. Our evaluation of RAID shows that it prevents DUET in its first stage while incurring negligible computation overhead, and no more than an increase of 13% in busload and 50 μs in message delays during the retraining mode of the VIDS.

We summarize our main contributions as follows.

- We discover the voltage corruption tactic, which enables a duo of attacker and accomplice ECUs to controllably corrupt the voltage samples measured by the VIDS.
- By exploiting the voltage corruption tactic, we propose a novel masquerade attack strategy, DUET, which is among the first attacks to evade the VIDS successfully.
- Our analytical and evaluation results illustrate that DUET is effective against all state-of-the-art VIDS irrespective of features and classification algorithms utilized in them.
- We present an efficient and effective defense, RAID, that equips CAN ECUs with the necessary tool to prevent APT-style attacks such as DUET.

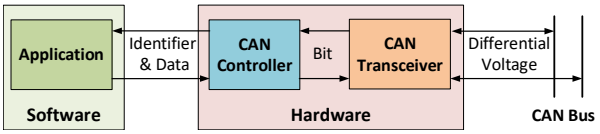


Fig. 1: Architecture of an ECU.

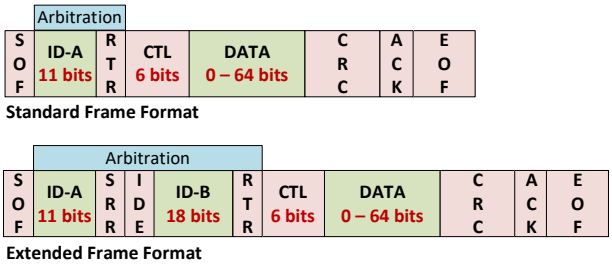


Fig. 2: Standard and extended frame formats in CAN.

II. Background and Key Observations

A. CAN Protocol

Figure 1 presents an overview of the architecture of a CAN ECU. We note that the application software can modify only the message identifier (ID) and the message data/payload. Most of the other CAN functionalities are handled in the hardware, including the framing, arbitration, and error handling performed by the CAN controller, and the physical-layer bus communication performed by the CAN transceiver [5].

Bus Communication. In CAN, the bits 1 and 0 are called recessive and dominant bits, respectively. CAN employs a twisted wire pair called CAN high (CAN-H) and CAN low (CAN-L) so that the recessive (1) and dominant (0) bits can be represented by the differential (between CAN-H and CAN-L) voltages of 0 V and 2 V, respectively. On the one hand, the ECU's CAN transceiver employs an "open" electrical circuit to realize the actual voltage value (≈ 0 V) for the transmission of a recessive bit which effectively remains *transparent* to other ECUs and the CAN bus. On the other hand, the actual voltage value ($\approx$ 2 V) induced on the bus by the ECU during the transmission of a dominant bit is dependent on the "closed" circuit characteristics of the ECU's CAN transceiver [9]. Due to this physical-layer protocol, when multiple ECUs transmit their bits concurrently on the CAN bus, the resultant voltage corresponds to the superposition of the voltages induced by the ECUs, and the resultant bit is equal to the logical AND of the transmitted bits. We note that since a malicious ECU with a compromised application *software* cannot alter the voltage characteristics demonstrated by its transceiver's hardware, VIDS exploits these characteristics to discreetly fingerprint the ECU and detect any masquerade attack carried out by it.

Framing. CAN messages are transmitted in either the standard or extended frame format (Figure 2). While a frame in the standard format is identified by an 11-bit ID-A field, a frame in the extended format is identified by a 29-bit ID, which is composed of two fields: an 11-bit ID-A field and an 18-bit ID-B field. In both formats, the length of the data field can be from zero to eight bytes, as indicated in the data length code in the control (CTL) field. We note that an ECU can transmit messages with different formats on the same bus.

Arbitration. The CAN employs a naive arbitration procedure among ECUs, which prevents any two ECUs to concurrently transmit their message payloads (the data fields) on the bus. Each ECU starts transmitting on the bus by synchronizing with the start-of-frame (SOF) field. For the ID fields, each ECU sends an ID bit on the bus and then reads it back from the bus. If an ECU reads a dominant bit while it transmitted a recessive bit, it loses the arbitration and stops transmitting; otherwise, it continues to transmit the message. We note that an attacker can impersonate the victim's ID, and then exploit this arbitration procedure to enable a deliberate overlap of the data field of its own message with that of the victim's message [7].

Error Handling. Communications on CAN may be corrupted due to many factors such as software/hardware faults and electromagnetic interference from other ECUs or nearby devices. During any transmission other than the arbitration field, a *bit-error* occurs when the transmitted bit is not equal to the resultant bit on the bus. To handle such errors, an ECU is defined to be in one of the three states: *error-active*, *error-passive*, and *bus-off*. By default, the ECU is in error-active state. Based on the number of encountered errors, the ECU transitions first to the error-passive state and then to the bus-off state. The ECU stops communication on the bus in the bus-off state and must be reset to the error-active state to restart communication. Prior studies have established that an attacker can deliberately cause bit-errors in victim's messages to enforce the victim to make these state transitions [7].

We note that there are two important differences in the behavior of an error-passive ECU when compared with an error-active ECU: (1) While transmitting two successive messages, the error-active ECU follows a regular 3-bit inter-frame spacing (IFS) consisting of a 3-bit intermission, but the error-passive ECU is required to have an *extended* 11-bit IFS consisting of an 8-bit suspend transmission field in addition to the 3-bit intermission. (2) When the error-active ECU encounters a bit-error, it immediately starts to transmit an error frame consisting of six dominant bits and eight recessive bits. On the contrary, after observing a bit-error, the error-passive ECU waits for the bus to become idle and then transmits an error frame consisting of 14 recessive bits. The proposed voltage corruption tactic exploits these distinct characteristics of the error-passive ECU (Section IV-A).

B. Vulnerable ECUs on Automotive CAN

Previous studies have demonstrated that automotive ECUs can be compromised through either physical [18], [24] or remote [6], [32], [36] attack surfaces. Specifically, the threats that are of grave concern include the compromise of:

1) the Telematics Control Unit through USB, Bluetooth, WiFi, vehicle-to-vehicle, or cellular connections [6], [32],
2) the long-range communication unit used for the road-side assistance and crash reporting [32],
3) the short-range communication unit for enabling keyless entry and tire pressure monitoring [6],
4) the radio and entertainment unit through a corrupted physical or over-the-air media [6],

5) the third-party devices connected to the on-board diagnostic (OBD)-II port [47], [49], and
6) the Battery Control Module through the wired connection utilized at battery-charging stations [3].

In many cases, a critical vulnerability affects multiple ECUs and leads to their compromise: Recent work has demonstrated the concurrent compromise of the instrument cluster ECU, central information display ECU, and gateway ECU of a Tesla car through a malicious WiFi/cellular connection [35]. This work was followed by another study that showed that the gateway ECU, body control unit, and autopilot ECU of a Tesla car could be remotely compromised at the same time [36]. Furthermore, an APT-style adversary can exploit a chain of vulnerabilities to compromise multiple ECUs. For instance, a malware-infected smartphone can result in the compromise of the Vehicle Communication Interface Module (connected to the smartphone through a Bluetooth connection) as well as the Telematics Control Unit (connected to the smartphone through a cellular connection) [6]. In another example, an adversary can first compromise a third-party device connected to the OBD-II port, and then exploit this device to send diagnostic messages to compromise the Telematics Control Unit [31].

Unfortunately, the above vulnerabilities also result in APTs, which can stealthily transcend over reboots and ECU firmware flashing/updates [31], [32], [35], [36]. As such, after compromising vulnerable ECUs, an adversary can launch a variety of attacks, including an ECU masquerade attack. We note that *not all* automotive ECUs (e.g., Engine Control Unit and Brake Control Unit) can be compromised directly, making the masquerade attack valuable to an adversary that aims to perform safety-critical maneuvers. In this paper, we build upon the capability of an adversary to compromise at least two of the vulnerable ECUs for launching the DUET attack.

C. Voltage-based Intrusion Detection System (VIDS)

The VIDS [9], [10], [14], [22], [23] provides an advanced technique for detecting compromised, unmonitored, and newly added ECUs on a CAN. It extracts the voltage fingerprint (which consists of features like mean and variance) from the measured voltage characteristics of each CAN message. It then learns a supervised model by inferring the source ECU using the message ID. In its operation mode, VIDS raises an alarm if it detects an anomaly between the observed fingerprint of a CAN message and the learned fingerprint of the source ECU indicated by the message ID. As such, the VIDS has been highly effective against ECU masquerade attacks.

However, the message voltage characteristics and hence the ECU fingerprints vary with time due to changing environmental/workload factors, firmware updates, and aging effects [9]. For instance, the VIDS presented in [14] is robust to fingerprint variations up to only 6 °C fluctuations in ambient temperature, while another VIDS [10] yields poor predictions beyond 10 °C temperature differences. Since such variations may happen within a few hours/days, the VIDS model must be updated accordingly through online learning [9], incremental learning [10], [14], or periodic model retraining [22], [23]. Such frequent (e.g., daily) retraining makes all existing VIDS vulnerable to training set poisoning attacks [4], [45]. Therefore, the VIDS must update its model only during the secure retraining mode while employing the established countermeasures against such attacks [20], [29]. As the VIDS stays in this mode for only a brief duration, the ECUs may also be equipped to employ MACs to authenticate their messages while the VIDS records their voltage fingerprints [22], [23].

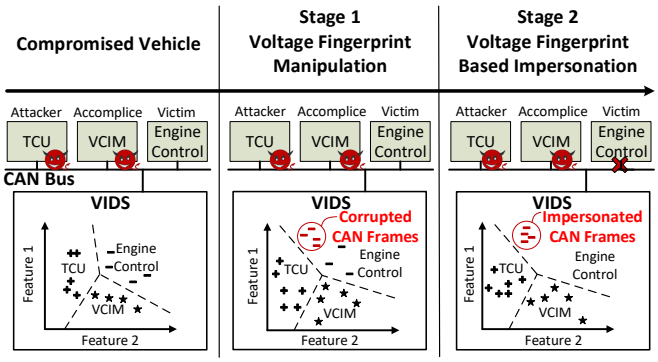


Fig. 3: Overview of DUET where the Telematics Control Unit (TCU) and Vehicle Communication Interface Module (VCIM) compromised using their wireless interfaces, collaborate to masquerade the Engine Control Unit and evade VIDS.

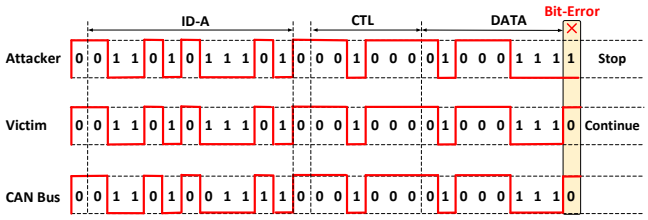
III. DUET OVERVIEW

A. Attack Tactic and Strategy

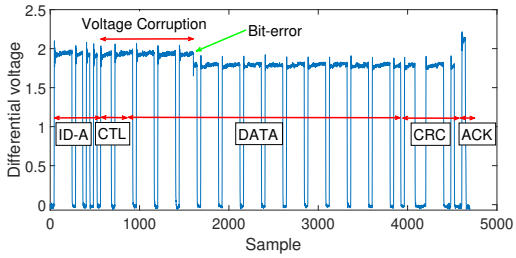
Voltage Corruption. In this paper, we notice the subtle fact that although the voltage characteristics of an ECU are immutable, those measured by the VIDS on the bus can be corruptible. To this end, we discover a novel voltage corruption tactic (Section IV-A) leveraged by two compromised ECUs – *attacker* and *accomplice* – to modify the voltage samples as measured by the VIDS. We find that when the attacker is in the error-passive state and performs simultaneous transmission with an ECU with the help of the accomplice, it will successfully corrupt the ECU's voltage samples, without leaving a trace on the bus. This finding leads to a critical vulnerability targeting at how the voltage data is physically collected at the bus-level by a VIDS. Hence, no voltage data-based learning mechanism is immune from it. As such, this tactic opens the window for data poisoning attacks against the VIDS.

DUET. Exploiting the voltage corruption tactic, we propose DUET, a novel ECU masquerade attack which evades detection from all existing VIDS as illustrated in Table I. DUET involves the duo of attacker and accomplice that carry out a training set poisoning-based attack strategy in a stealthy, two-stage fashion. DUET includes voltage fingerprint manipulation (Stage 1) and voltage fingerprint-based impersonation (Stage 2) as illustrated in Figure 3. Fundamentally, the attacker employs the voltage corruption tactic to corrupt the voltage fingerprint of the victim in Stage 1 (Section IV-B1) and that of the accomplice in Stage 2 (Section IV-B2) of DUET. This way, the DUET duo work together to first trick VIDS into learning the corrupted voltage fingerprint of the victim as its true fingerprint in Stage 1, and then classify the corrupted fingerprint of the accomplice as the fingerprint of the victim in Stage 2.

Exploited Characteristics of CAN Traffic. DUET mainly exploits three common characteristics of CAN traffic: (1) static



(a) Bit-error caused by the attacker after one byte of data.



(b) Voltage samples on the bus as observed by a VIDS.

Fig. 4: Illustration of the impact of the voltage corruption tactic on a victim ECU's message.

TABLE II: Common characteristics of CAN traffic (each number is a message count).

	2011 ExpCar-1[1]	2013 ExpCar-2[1] (Bus-1)	2013 ExpCar-2[1] (Bus-2)	2012 Toyota Camry	2012 Honda Civic	2010 Dodge Ram
Total messages	50	88	27	42	45	55
Standard frame	50	88	27	42	45	55
Periodic	50	88	27	42	45	51
PREP $\geq$ 1 byte	49	83	25	42	42	50

IDs, (2) message periodicity, and (3) *predictable payload-prefix* (PREP), which is the predictable set of bits representing constants, counters, or multi-valued numbers after the arbitration field in CAN messages (Figure 2). We confirm these characteristics using the CAN traffic from five vehicles (of different brands) which include two of our experimental[1] and three non-experimental vehicles. Table II summarizes our findings, with detailed results in Appendix A. We have also validated the prevalence of these characteristics in other modern vehicles using the reverse-engineered data available at [11].

B. Adversary Model

We follow the attack model in prior art [7]–[9], [22] and let the adversary make a *one-time* reverse engineering effort [27], [39] to infer a basic understanding of the payload and periodicity of CAN messages in the target vehicle or in a vehicle with the same make and model. We assume that the adversary behind DUET is capable of achieving arbitrary code execution on at least two ECUs of an automotive CAN. Such compromised ECUs – attacker and accomplice – can read and inject messages on the bus through their CAN controllers and transceivers, and use the message payloads to coordinate their actions during an attack. DUET's strategy to evade the VIDS is orthogonal to the strategies utilized to evade the MIDS and CIDS. Integrating these strategies to design a masquerade attack that evades all IDS is out of scope for this paper.

IV. Detailed Design of DUET

A. Voltage Corruption Tactic

To evade a VIDS, a compromised ECU, i.e., the attacker, should be able to *controllably* modify its own voltage fingerprint and/or the voltage fingerprint of the uncompromised ECU, i.e., the victim. We highlight that the attacker faces two critical CAN-specific challenges in achieving these objectives: (1) Since the attacker cannot directly alter either the software or the hardware of the victim, it cannot control the voltage samples generated by the victim. (2) The attacker cannot modify its own voltage fingerprint because the voltage samples are generated by its CAN transceiver, which is implemented in the hardware (Figure 1). To address these challenges, we discover a novel attack tactic called *voltage corruption* that enables the attacker to modify the voltage samples (as *measured* by the VIDS) of any target ECU including the victim.

This tactic exploits the unique behavior of an attacker in error-passive state while transmitting an *attack message* simultaneously with a victim's message. In its attack message, the attacker transmits the same bits as the victim until a specific bit location, where the attacker and victim transmit the recessive bit and dominant bit, respectively. At that location, a bit-error occurs in the attack message, which terminates the attacker's transmission. Since the resultant bit on the bus is dominant, the victim does not observe the bit-error and continues to successfully transmit its message. This way, VIDS computes the voltage fingerprint of the victim's message using the measurements from two sets of voltage samples: (1) corrupted samples resulting from the overlap between the attacker's and victim's transmissions before the bit-error, and (2) benign samples corresponding to the benign bits of the victim after the bit-error. An example illustrating the voltage corruption tactic is shown in Figure 4: The error-passive attacker transmits simultaneously with a victim and causes a bit-error after one byte of data, as illustrated in Figure 4a. Then it is clear in Figure 4b that corrupted voltage values of the overlapped bits before the bit-error are higher than the voltage values of the benign bits after the bit-error.

To implement this attack tactic against a victim on a real CAN bus while evading VIDS, the attacker must: (1) transition to the error-passive state, (2) transmit simultaneously with the victim, (3) transmit an attack message with the same content as the victim's message until a bit location, and (4) prevent the retransmission of its attack message. We present the techniques facilitating these requirements for the attacker as follows.

1) Error State Transition: In the attack tactic, the attacker must be in the error-passive state so that before the bit-error, the attack message overlaps with the victim's message without hindering it; whereas, after the bit-error, the attacker waits until the end of the victim's message and then transmits a passive

[1]We decided to anonymize the make and model of our experimental vehicles because DUET, which exploits fundamental characteristics of CAN, is *not* specific to those vehicles.

error frame. Since the passive error frame consists of only recessive bits which induce 0 V on the bus, it is *transparent* to the victim and leaves no trace on the bus.

Role of the Accomplice. Similar to a typical ECU, the attacker is in the error-active state by default. We note that the error state is a characteristic of CAN controller (hardware), and it cannot be configured through software. Hence, the attacker requires the assistance of an accomplice that can deliberately cause bit-errors in the attacker's messages to transition the attacker from the error-active state to the error-passive state. Similar to [7], the accomplice first synchronizes with the attacker's message transmission on the bus, and then transmits another message comprising of the same message identifier but different payload as compared to those of the attacker's message. The collisions of these messages from the attacker and accomplice trigger bit-errors and increase the attacker's error counter, enabling the transition to the error-passive state.

2) Simultaneous Transmission: A simultaneous transmission comprises of two or more concurrently transmitted messages. In the attack tactic, the attacker must perform the simultaneous transmission with the victim, i.e., the attacker must start its transmission at the same time as the victim. However, it is difficult to do this in practice because of two challenges: (1) Although the victim transmits its message periodically, there is an inherent jitter in its transmission time based on its contemporary workload. (2) The attacker cannot precisely control the transmission time of its own attack message because the detection of the bus' availability and its readiness to start transmission are performed by its CAN controller, which is implemented in the hardware.

To address these challenges, we employ a *preceded ID message*, which is injected right before the periodic transmission of the victim's message [7]. The preceded ID message forces any other message to wait for the completion of its transmission. This way, although the victim and attacker may generate their messages at slightly different times when the preceded ID message is transmitted, they can start transmitting their messages right after the compulsory inter-frame spacing (IFS) following the preceded ID message. Hence, the time synchronization between the victim and attacker can be realized if both of them are required to have the same IFS.

Role of the Accomplice. Unfortunately, the attacker being in the error-passive state requires an extended 11-bit IFS between its successive messages. On the contrary, the victim being in the error-active state requires only a regular 3-bit IFS. This means that when the attacker transmits a preceded ID message, the victim and attacker must wait for 3-bit and 11-bit IFS, respectively, before the transmission of their messages. As such, the victim gets to transmit its message before the attacker. This implies that the attacker cannot first transmit the preceded ID message to synchronize with the victim and then transmit the attack message to corrupt the victim's message. Therefore, to realize the attack tactic, the attacker must inevitably obtain help from the accomplice, which can transmit the preceded ID message. As a result, the attacker can now have the regular 3-bit IFS after the accomplice's preceded ID message. This enables the attacker to transmit its attack message simultaneously with the victim's message.

3) Content Impersonation: In the attack tactic, the attack message must overlap the victim's message until the desired bit location. In a benign environment, since each of the CAN message transmitted by ECUs is allotted a *unique* ID, only one ECU wins the arbitration, which eliminates the risk of such an overlap by any other ECU. To bypass this constraint, the attacker deliberately utilizes the victim's message ID in its attack message. Further, the CAN traffic from the five cars we profiled (Appendix A) reveals an interesting observation: The content after the arbitration field associated with a given message ID contains a *predictable payload-prefix* (PREP) of bits representing constants, counters, and multi-valued numbers which can be reliably predicted in advance. Since the payload length for a specific message ID remains the same, the PREP consists of at least six constant bits of the control field. With offline reverse-engineering of PREP in the victim's payload, the attacker can readily identify additional PREP bits in the data field (Section VII-A2). Then the attacker can utilize a selected portion of the PREP in its attack message and enforce the bit-error at any desired location within the PREP.

4) Retransmission Prevention: Without special precaution, the attack tactic will exhibit an anomaly that can be detected by the VIDS: After encountering the bit-error in the attack message (Figure 4a), the attacker ECU's CAN controller will attempt to retransmit the attack message. Since this retransmitted attack message will contain the victim's message ID and the attacker's voltage fingerprint, the VIDS can detect this anomaly and raise an alarm. We eliminate this anomaly by exploiting a capability called *one-shot transmission mode*, available in all the popular CAN controllers [30], [41]. On the one hand, in the normal transmission mode, an ECU's CAN controller attempts to retransmit a message until it is transmitted successfully. On the other hand, in the one-shot transmission mode, the controller transmits the message only once. This implies that if such a one-shot transmission is interrupted due to any reason (including the loss in arbitration or bit-error), the message is lost. Due to this unreliability, the one-shot transmission mode is typically not employed in the traditional CAN. Yet in the attack tactic, we discover that the attacker can advantageously exploit the one-shot transmission mode for transmitting the attack message. In this case, since the attack message's transmission is terminated after encountering the bit-error, the controller does not attempt to retransmit the attack message. As such, the attacker evades VIDS detection while performing voltage corruption.

B. DUET

The voltage corruption tactic reveals a fundamental vulnerability at the physical-layer of the CAN protocol stack. This vulnerability critically hinders the ability of the VIDS to collect benign voltage samples of any ECU, including the victim. Also, no machine learning (ML)-based technique employed in the VIDS is immune to this vulnerability. However, it is non-trivial to exploit this attack tactic for evading the VIDS during an ECU masquerade attack because the attacker cannot forge (but only corrupt) the victim's voltage fingerprint using the tactic. As such, the attacker and accomplice duo face two fundamental challenges: (1) They must deceive the VIDS into considering the corrupted fingerprint of the victim as the "true" fingerprint of the victim. (2) They must also trick the VIDS

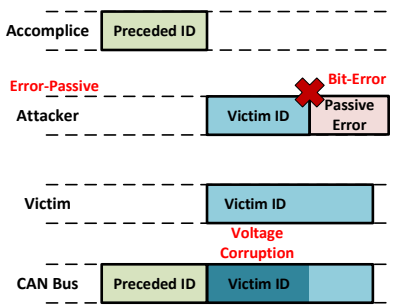


Fig. 5: Voltage fingerprint manipulation (Stage 1).

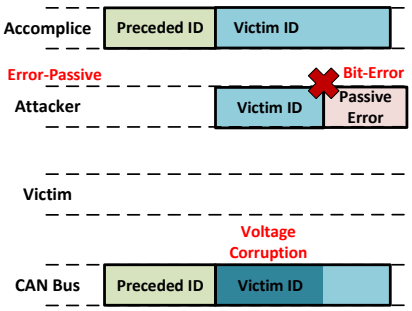


Fig. 6: Voltage fingerprint-based impersonation (Stage 2).

into detecting that the voltage fingerprint of a forged message is close to the victim's corrupted voltage fingerprint.

To address the above challenges in evading the VIDS, we present DUET, an ECU masquerade attack strategy that exploits the voltage corruption tactic realized by the duo of attacker and accomplice. DUET is driven by the motivating idea that the ML-based IDS including those used in automotive control systems [40], can be susceptible to adversarial ML techniques leveraging data poisoning. As such, DUET is among the first efforts in training set poisoning on CAN. DUET follows a two-stage coordinated attack which includes: (1) the voltage fingerprint manipulation to poison the VIDS training set, and (2) the voltage fingerprint-based impersonation to forge the victim's messages without raising a VIDS alarm.

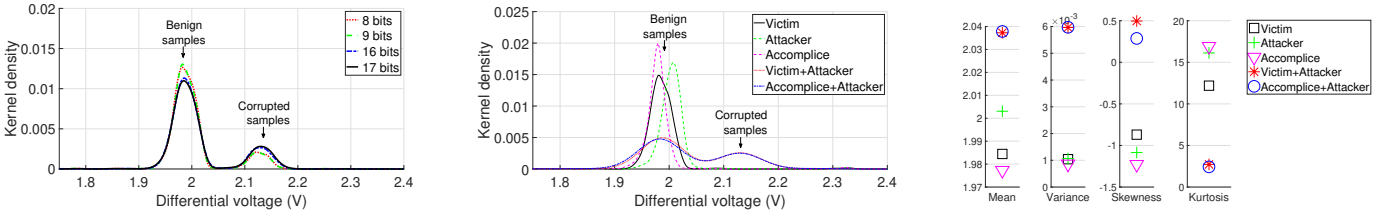
1) Stage 1. Voltage Fingerprint Manipulation: The first stage of DUET, voltage fingerprint manipulation, is carried out by the attacker and accomplice to stealthily distort the voltage fingerprint of the victim ECU (as learned by the VIDS) by utilizing the voltage corruption tactic against the victim. Specifically, as shown in Figure 5, the accomplice first assists the attacker to attain the error-passive state. Then the attacker and accomplice independently estimate the time-of-transmission of the victim's message by exploiting its periodic behavior. The accomplice injects a preceded ID message right before the transmission of the victim's message to help the attacker synchronize with the victim's transmission. The error-passive attacker utilizes the victim's ID and a portion of the PREP bits in its attack message. Finally, the attacker transmits its attack message simultaneously with the victim's message and corrupts the victim's voltage samples. Since a VIDS learns from the (now corrupted) voltage measurements, the *distorted fingerprint of the simultaneous transmission* of "victim + attacker" is mistaken for the victim's fingerprint by the VIDS. Further, to manipulate the distorted victim's fingerprint, the attacker can control the bits corrupted in the victim's message by selecting an appropriate portion of the PREP bits.

We note that Stage 1 of DUET has a critical objective: corrupting the PREP bits of the victim's messages in the training set without raising a VIDS alarm. This objective can be effectively fulfilled if the messages utilized for retraining the VIDS can be identified, e.g. when they contain distinct IDs (such as those in [10]) or distinct payloads (such as MACs in [22]). However, a VIDS might be trained silently (without any observable indication on the bus) with regular CAN messages; thus, identifying the training set messages can be impractical. In such cases, manipulating victim's fingerprint is a particularly difficult endeavor because a significant corruption in victim's messages not used for retraining could be detected by the VIDS as an attack.

To address this challenge, DUET exploits the fact that a VIDS requires periodic retraining (e.g., daily or when the engine starts) to account for changing environmental and weather conditions [14]. As such, for a VIDS, we define the *inter-retraining period* as the time between two VIDS retraining sessions. Meanwhile, to enhance the impact of the voltage corruption tactic, DUET intends to increase the number of corrupted bits in the victim's message from zero to the desired bits in PREP. As such, we define the *inter-manipulation period* as the time between a single increment in the number of corrupted bits. DUET sets the inter-manipulation period *higher* than the inter-retraining period of the VIDS. Then throughout one inter-manipulation period, DUET manipulates the same number of corrupted bits in the victim's messages. In other words, DUET increases the corruption one bit at a time and then continues conservatively with the same number of corrupted bits through at least one inter-retraining period of VIDS. This way, DUET tricks the VIDS into learning an increasingly manipulated fingerprint of victim over multiple inter-retraining periods without raising an alarm (Section VII-B).

Due to the inherent jitter in the transmission time of the victim's messages, the attacker fails to transmit simultaneously with some of the victim's messages and corrupt their voltage fingerprints during the VIDS's retraining mode. These benign messages enable the VIDS to learn the benign voltage fingerprint along with the distorted fingerprint of the victim during the retraining mode. This works in DUET's favour as such a benign message of the victim does not trigger any VIDS alarm during the operation mode of the VIDS too.

2) Stage 2. Voltage Fingerprint-Based Impersonation: With the victim's fingerprint corrupted in Stage 1, the DUET duo proceed to the second stage, voltage fingerprint-based impersonation where they inject forged messages to impersonate the victim. To achieve this, the attacker utilizes the voltage corruption tactic on the accomplice as the "victim". Specifically, the attacker first indicates the accomplice to transition from Stage 1 to Stage 2 of DUET using its periodic message. Then, as shown in Figure 6, the attacker and accomplice synchronize using the preceded ID message transmitted by the accomplice. Finally, the accomplice injects a (forged) victim's message, and the attacker transmits simultaneously corrupting



(a) Distributions of victim's voltage samples demonstrating the closeness between those with consecutive number of corrupted bits.

(b) Voltage distributions illustrating the closeness of distributions observed for the "victim + attacker" and "accomplice + attacker".

(c) Illustration of the closeness of voltage features observed for the "victim + attacker" and "accomplice + attacker".

Fig. 7: Characteristics observed by a VIDS during DUET attack.

the accomplice's voltage samples. Hence, the VIDS records the distorted fingerprint of the simultaneous transmission of "accomplice + attacker". DUET succeeds if receiver ECUs on the CAN bus accept this spoofed message, but the VIDS does not raise any alarm. We note that similar to Stage 1, the attacker can control the length of the superposition of its attack message and the accomplice's forged message by regulating the location of the bit-error in its attack message. The formal and empirical analysis of the stealth of DUET against VIDS are presented in Section V and Section VII, respectively. Below we describe the fundamental reasons for the stealth of DUET against the VIDS through an illustrative example.

C. Example Illustrating Stealth of DUET

Figure 7 presents the distribution and features of voltage samples measured by the VIDS when the attacker corrupts two bytes of PREP bits in the victim's message during the retraining mode and in the accomplice's message during the operation mode. For reference, we also include the uncorrupted characteristics of the victim, attacker and accomplice.

Distribution of Voltage Samples. DUET's goal in Stage 1 is to corrupt the fingerprint of the victim's messages by corrupting their PREP bits over multiple inter-retraining periods of VIDS. In Figure 7a, we observe that the voltage distribution of victim's messages with two *consecutive* number of corrupted bits are statistically close to each other. This implies that VIDS will fail to detect the voltage corruption if DUET gradually increments the number of corrupted bits from zero to the maximum number of PREP bits in the victim's messages.

Figure 7b illustrates the distinct distributions of the voltage samples in an *individual transmission* (by the victim, attacker, or accomplice) and a *simultaneous transmission* (by the "victim + attacker" or "accomplice + attacker"). On the one hand, the voltage samples of an individual transmission closely fit a *unimodal* Gaussian distribution which is consistent with the findings in prior art [9]. On the other hand, we discover that the voltage samples in a simultaneous transmission follow a *bimodal* Gaussian distribution. This is because in a simultaneous transmission, the voltage corruption tactic results in two sets of voltage samples in a CAN message: corrupted and benign samples, such that corrupted samples have higher voltage values than benign samples as shown in Figure 4. We highlight that a bimodal distribution of voltage samples can also be observed on the bus in benign environments, e.g., when electromagnetic interference from other sources distort the voltage samples of an ECU and when two messages (with similar, but non-identical IDs) transmitted by two benign ECUs overlap over a significant portion of their arbitration fields.

Voltage Features. In Figure 7c, we observe that the voltage features computed using the two bimodal distribution are statistically "closer" to each other than those computed using a unimodal distribution. In other words, the values of features (such as variance and skewness of samples) of a simultaneous transmission ("victim + attacker") are closer to those of another simultaneous transmission ("accomplice + attacker") than those of an individual transmission. Hence, an ML-based VIDS classifier (which utilizes a multitude of such features to identify the source) is likely to *mis-classify* the fingerprint of the simultaneous transmission of "accomplice + attacker" as that of "victim + attacker", irrespective of the features and classification algorithm employed by the classifier.

Root Cause for Successful Impersonation. Exploiting the above fundamental shortcoming of the VIDS, DUET, in Stage 1, utilizes a simultaneous transmission ("victim + attacker") to trick VIDS into learning a distorted fingerprint of the victim during the retraining mode. Since the fingerprint of all other ECUs on the CAN bus remain uncorrupted, VIDS learns the decision boundaries such that it can identify the sender of any message with a distorted fingerprint to be the victim. Hence in Stage 2, DUET successfully impersonates the victim by crafting the distorted fingerprint with another simultaneous transmission ("accomplice + attacker"). Note that each individual feature of the resultant "victim + attacker" and "accomplice + attacker" transmission after voltage corruption does *not* need to be identical. DUET evades VIDS since the fingerprint (the combination of all features) of "accomplice + attacker" (the impersonated fingerprint) lies within the decision boundaries learned through the fingerprint of the "victim + attacker" (the manipulated fingerprint of the victim).

V. ANALYSIS OF DUET

We now formally analyze the stealth of DUET against VIDS by modeling the objective of DUET as an adversarial data poisoning problem. Further, we describe the metric to quantify the stealth of DUET, and the parameters impacting it.

VIDS. We define the functionality of a VIDS as follows: Let the training data be represented as $\mathbf{D}_o = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^q$ represents a fingerprint/vector of q features,

$y_i \in [1, C]$ represents the class, C represents the total number of classes, and N represents the number of samples utilized for retraining. In the retraining mode, the VIDS classifier learns the model $\theta_o = \text{argmin}_{\theta \in \Theta} \ \mathcal{O}_t(\mathbf{D}_o, \theta)$, where Θ represents the hypothesis space, and $\mathcal{O}_t$ represents the objective function of the classifier in the retraining mode. During the operation mode, for a given vector of features $\mathbf{x}_*$ and the model θ_o, the classifier outputs $y_* = \text{argmax}_{c \in [1,C]} \ \mathcal{O}_d(c \mid \mathbf{x}_*, \theta_o)$, where $\mathcal{O}_d$ represents the objective function of the classifier in the operation mode. Further, the VIDS employs a function (denoted by $\mathcal{F}$) to detect an unknown (i.e., significantly distinct from previously observed) fingerprint during the retraining/operation mode. Let the VIDS detect a fingerprint $\mathbf{x}_*$ to be anomalous if $\mathcal{F}(\mathbf{x}_*) > \tau$, where τ represents the detection threshold.

Attack Success. We now formally define the "success" of DUET. Let the class of the victim be y_v, the space of the feature vectors corresponding to the simultaneous transmission of the victim and attacker be $\mathbf{X}_{v+a}$, and the space of the feature vectors corresponding to the simultaneous transmission of the accomplice and attacker be $\mathbf{X}_{a+a}$. In Stage 1, DUET makes the VIDS record a manipulated training set (denoted by $\mathbf{D}_m$) which includes the samples $(\mathbf{x}_{\hat{v}}, y_v)$, such that $\mathbf{x}_{\hat{v}} \in \mathbf{X}_{v+a}$. The classifier learns the manipulated model (denoted by θ_m) using this training set in Stage 1. When the manipulated model is utilized by the VIDS in Stage 2, DUET succeeds if the classifier outputs y_v given the input $\mathbf{x}_{\hat{a}} \in \mathbf{X}_{a+a}$.

Success Rate. To quantify the stealth of DUET, we define a metric called *success rate* as the probability of an impersonated message evading the VIDS detection. The success rate of DUET is affected by two major factors: (1) *Timing Accuracy:* The success of DUET relies on the timing accuracy of the attacker in synchronizing the transmission of its attack message with the victim's message during Stage 1 (denoted by p_v) and that with the accomplice's forged message during Stage 2 (denoted by p_a). We note that the values of p_v and p_a depend on the characteristics (e.g., busload) of the targeted CAN as discussed in Section VII-A1. (2) *PREP Length:* The length of PREP in the victim's message (denoted by L) limits the number of bits which can be corrupted by the attacker in the victim's message during Stage 1 (denoted by l_v) and that in the accomplice's forged message during Stage 2 (denoted by l_a). We note that the attacker can control the values of l_v and l_a as elaborated in Section VII-A2.

Data Poisoning Problem. The problem of finding the best values of l_v and l_a to maximize the probability (denoted by Pr) of DUET's success can be defined as:

$$\begin{aligned} \underset{l_v, l_a \in [0,L]}{\text{argmax}} \ \Pr\left(y_v = \underset{c \in [1,C]}{\text{argmax}} \ \mathcal{O}_d(c \mid \mathbf{x}_{\hat{a}}, \theta_m) \right), \\ \text{s.t. } \theta_m = \underset{\theta \in \Theta}{\text{argmin}} \ \mathcal{O}_t(\mathbf{D}_m, \theta), \ \mathcal{F}(\mathbf{x}_{\hat{v}}) \leq \tau, \\ \mathcal{F}(\mathbf{x}_{\hat{a}}) \leq \tau, \ \mathbf{x}_{\hat{v}} \in \mathbf{X}_{v+a}, \text{ and } \mathbf{x}_{\hat{a}} \in \mathbf{X}_{a+a}. \end{aligned}$$

In this expression, DUET first tricks the VIDS into learning the manipulated model θ_m during the retraining mode, and then tricks the VIDS into classifying the spoofed fingerprint $\mathbf{x}_{\hat{a}}$ to the victim's class y_v. We note that it is extremely challenging to find a closed-form solution to the above optimization problem for an instantiation of VIDS. Nevertheless, by setting values as $l_v = l_a > 2$ bytes, the DUET duo manages to evade the existing VIDS with a high (75%) success rate as shown in the empirical results presented in Section VII-C.

VI. Implementation Details

DUET Implementation. DUET is written in 1400 lines of C++ code[2]. Utilizing only 950 bytes of memory, and 13 KB of flash storage, DUET is lightweight and can be easily deployed in existing ECUs. The demonstration of DUET on a real car can be accessed at https://youtu.be/NGuh0iXNE20.

VIDS Implementation. We evaluate the stealth of DUET against two representative state-of-the-art VIDS: (1) *Viden* [9] with online learning and (2) *Scission* [22] with periodic retraining. We note that the evaluation results of DUET against Scission can be utilized to infer the performance of DUET against other VIDS [10], [14], [23] since these VIDS fundamentally employ a subset of samples/features employed in Scission.

Viden. The voltage samples for Viden are collected by sampling the CAN bus at 50 kS/s. Due to the low sampling rate and the need to map the samples back to the message ID in real-time, the number of measured voltage samples per message is limited to eight samples total, four for CAN-H and four for CAN-L. Viden employs three features for CAN-H (50^{th}, 75^{th}, and 90^{th} percentiles of CAN-H samples) and another three features for CAN-L (10^{th}, 25^{th}, and 50^{th} percentiles of CAN-L samples). Each of the three features in CAN-H or CAN-L is calculated over at least 30 voltage samples. Hence, one set of six features is obtained over eight messages. During the retraining mode, we record CAN-H and CAN-L voltage samples corresponding to 1600 messages transmitted by each ECU. We compute the thresholds (corresponding to CAN-H and CAN-L) to exclude the samples corresponding to the acknowledgment. For each ECU, we then compute 200 samples of the six features. We finally train a 200-tree random forest classifier with these samples. During the operation mode, we collect $100,000$ messages, record the CAN-H and CAN-L voltage samples corresponding to them, and classify their source using the trained model.

Scission. The voltage samples for Scission are collected by sampling the CAN bus at 20 MS/s. During the retraining mode, we record differential voltage samples corresponding to 200 messages transmitted by each ECU. Using the samples in each message, we compute 24 features (e.g., mean, variance, skewness, and kurtosis) in the time domain, and 24 features in the frequency domain. We then obtain the standardized values for each of the features. We pre-process the features to find the 18 most significant features by utilizing the Relief-F algorithm. We finally train a multinomial logistic regression model with the 18 features. During the operation mode, we again record voltage samples of $100,000$ messages and classify their source using the trained model. We also consider modified versions of Scission by replacing the logistic regression with other mainstream ML algorithms, including support vector machine (SVM), Naive Bayes, and random forest classifiers. The hyperparameters for these classifiers are tuned using grid search within a nested 10-fold cross-validation.

Evaluation Platforms. We evaluate DUET through comprehensive experiments on our lab testbed and two real vehicles.

[2]The code is available upon request only.

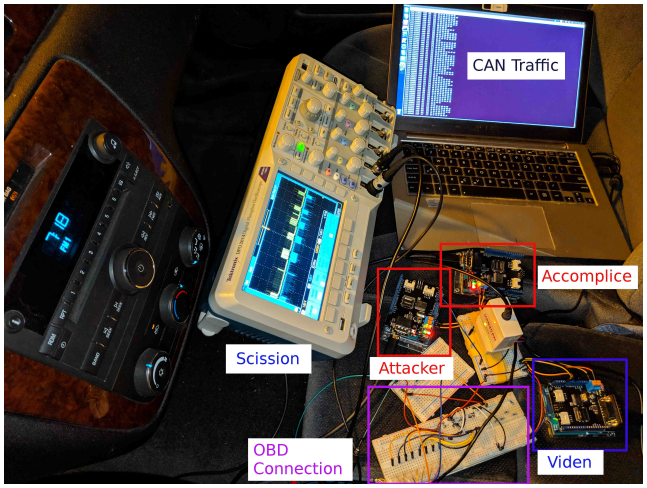


Fig. 8: Experimental setup within a real vehicle.

Lab Testbed. We set up a CAN bus testbed with 10 nodes. Each node is an ECU based on an Arduino UNO board with a 16 MHz Microchip and a SeeedStudio CAN shield with a MCP2515 CAN controller. Two of those nodes play the attacker and accomplice ECUs, one plays the victim targeted by DUET, and the remaining six nodes represent other uncompromised ECUs. The VIDS (i.e., Viden and Scission) run in a laptop connected to the bus. The voltage samples for Viden and Scission are collected by an uncompromised node and the Tektronix DPO2014 oscilloscope (with 200 MHz bandwidth and 8 bits vertical resolution), respectively. The standard CAN bus speed of 500 kbps is set in the testbed. We follow the benchmark proposed in [25] to generate a total of 60 messages with different sizes and periodicity for these nodes, resulting in 50% busload. To emulate a vehicular CAN bus, we remove the built-in resistances in CAN shields, and terminate the bus with a $120\,\Omega$ resistance at each end. We utilize a stub resistance of $2.4\,\text{k}\Omega$ for connecting the oscilloscope to the bus.

ExpCar-1[1]. The 2011 ExpCar-1 contains a CAN bus operating at 500 kbps with four ECUs. The CAN bus traffic comprises of 50 messages resulting in 35% busload. The experimental setup in the ExpCar-1 is shown in Figure 8. Through a customized OBD connector, we connect the vehicular CAN bus to two external ECUs acting as the attacker and accomplice. We also connect another ECU and the oscilloscope for recording voltage samples for Viden and Scission, respectively. With a CAN USB adapter (USB2CAN), a laptop is used for recording and analyzing the bus traffic.

ExpCar-2[1]. The 2013 ExpCar-2 contains *two* CAN buses operating at 500 kbps: *Bus-1* supports six ECUs which transmit 88 messages resulting in 61% busload and *Bus-2* supports three ECUs which transmit 27 messages resulting in 34% busload. To record the traffic and voltage samples, we utilize a setup similar to the one shown in Figure 8.

VII. EVALUATION OF DUET

A. Feasibility of Voltage Corruption

DUET's voltage corruption is feasible only if the attacker is able to transmit simultaneously with the victim's messages beyond their arbitration fields. This condition is readily satisfied by real-world vehicles due to the predictable message timing (due to periodicity) of CAN messages and the existence of PREP in those messages. The generality of these conditions are confirmed by profiling the CAN traffic in five cars (details in Table II and Appendix A), and utilizing publicly available reverse-engineering results for recent cars [11], [15]. The generality of PREP is also corroborated by studies of CAN traffic in the existing literature. For example, the researchers in [28] found that out of 456 total bytes of payload across all CAN messages in a 2012 Ford Focus, 338 bytes belonged to predictable categories (i.e, constants, counters and multi-valued numbers), and only 118 bytes belonged to unpredictable categories (i.e., sensor readings and unclassified bits).

TABLE III: Message timing accuracy in DUET.

Platform	Messages	Busload	Accuracy in Stage 1 (p_v)	Accuracy in Stage 2 (p_a)
Lab Testbed	60	50%	87%	92%
ExpCar-1	50	35%	81%	89%
ExpCar-2: Bus-1	88	61%	60%	85%
ExpCar-2: Bus-2	27	34%	80%	90%

We validate the above results through our analysis of the traffic on the three CAN buses in the two experimental cars. For each bus, the data was collected for 15 minutes in the stationary car, and for another 15 minutes in the moving car.

1) Timing Accuracy: In DUET, the attacker estimates the time of transmission of the victim's messages and accomplice's messages, and corrupts their voltage fingerprints by simultaneously transmitting its attack messages. Hence, the timing accuracy of the attacker's attack message with the victim's message in Stage 1 (p_v) and that with the accomplice's forged message in Stage 2 (p_a) affect the success rate of DUET. Table III presents the average values of p_v and p_a in all four evaluation platforms. Due to the highest busload in the ExpCar-2: Bus-1, its p_v and p_a are the lowest among the platforms. We also observe that although the testbed experiments were conducted with higher busload than ExpCar-2: Bus-2 and ExpCar-1, its p_v and p_a are higher. This is attributed to the larger jitter in message transmission by real ECUs compared with that by ECU prototypes in our testbed. We note that the factors affecting the timing accuracy, which include the number of CAN messages, busload, and message periodicities are independent of the control state of the car, i.e., they do not change while the car is moving or is stationary.

2) PREP Length: We calculate the length of PREP in a CAN message as follows. For each bit location after the arbitration field in the message, we calculate the conditional entropy, i.e., the randomness in the current bit given the bits (at the same location in the message) in the previous 16 messages. We consider the bit to be *predictable* if the conditional entropy is less than or equal to 0.01. We note that while the conditional entropy is one for a randomly generated bit, it is equal to zero for the bit which remains constant in all messages. The length of PREP is given by the number of bits from the first bit in the control field to the first variable bit with conditional entropy more than 0.01. Figure 9 presents the *cumulative* fraction of messages containing different lengths of PREP on the CAN buses of our experimental vehicles. For example, out

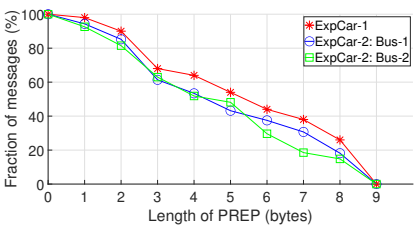


Fig. 9: Cumulative fraction of CAN messages with the given PREP length demonstrating that the majority of messages have less than three bytes of PREP.

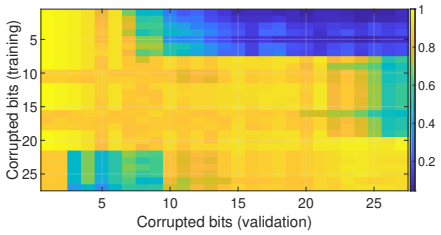


Fig. 10: Viden's classification accuracy for victim's corrupted messages during Stage 1 of DUET illustrating small increments in corrupted bits evades Viden.

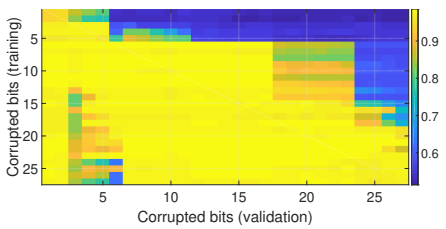


Fig. 11: Scission's classification accuracy for victim's corrupted messages during Stage 1 of DUET illustrating small increments in corrupted bits evades Scission.

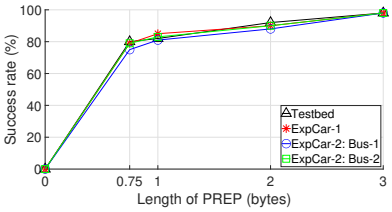


Fig. 12: DUET's success rate against Viden on different platforms.

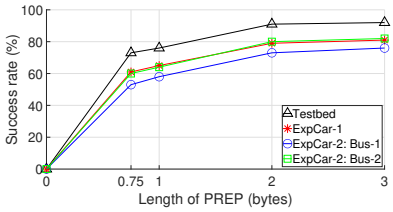


Fig. 13: DUET's success rate against Scission on different platforms.

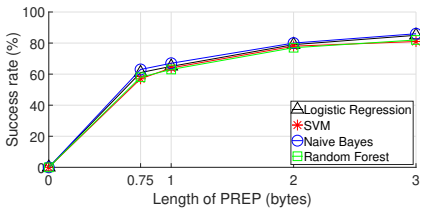


Fig. 14: DUET's success rate against Scission with different ML algorithms.

of 88 messages on ExpCar-2: Bus-1, 75 messages ($\approx$ 85%) have at least two bytes of PREP. This means that the attacker can readily continue to transmit simultaneously with one of these 75 messages until two bytes after the arbitration field.

B. *Stealth in Stage 1 of* DUET *against VIDS*

Figures 10 and 11 present the classification accuracy of Viden and Scission, respectively, for different number of corrupted bits in the victim's messages utilized for the training set (y-axis) and those utilized for the validation set (x-axis). These results depict that DUET effectively tricks both VIDS into correctly classifying the validation set's messages with *slightly more* corrupted bits after these VIDS learn the model from the corrupted training set's messages. Hence, DUET evades these VIDS by enhancing the voltage corruption while keeping the inter-manipulation period higher than the inter-retraining period. In other words, between any two retraining sessions of the VIDS, DUET takes a downward step of only one bit from the upper-left corner and stays just above the diagonal in these figures so that these VIDS fail to detect DUET's attempts of manipulating the victim's voltage fingerprint.

Time Needed for Desired Manipulation. Since the inter-manipulation period set by DUET is higher than the inter-retraining period of the VIDS, Stage 1 of DUET spans over multiple retraining sessions of the VIDS. As such, on the one hand, Stage 1 of DUET will quickly conclude against an online training-based VIDS, e.g, within a few seconds against Viden which learns from each message. On the other hand, it will take a relatively longer time to conclude against a periodic learning-based VIDS, e.g., within a few days against Scission which may have a daily retraining schedule. Nevertheless, DUET tricks the VIDS into learning an increasingly manipulated fingerprint of the victim and ensures that the fingerprint is manipulated stealthily. We point out that such "low-and-slow" nature of DUET is aligned with other in-vehicle APTs [26].

Victim's Benign Messages. As DUET cannot corrupt all victim's messages due to imperfect timing accuracy as shown in Table III, these VIDS inherently learn the voltage fingerprints of both corrupted *and* benign/uncorrupted victim's messages in the training set. As shown in Figures 10 and 11, this adds to the stealth of DUET's since the victim's benign messages in the validation set are also correctly classified by the VIDS. This explains why during its operation mode, the VIDS does not raise any alarm on observing the victim's benign messages.

C. *Stealth in Stage 2 of* DUET *against VIDS*

Success Rate. Figures 12 and 13 present the *per-message success rates* of DUET against Viden and Scission in the four CAN platforms, respectively. We observe that with larger PREP, the distorted fingerprints of the simultaneous transmissions are more distinct from the fingerprints of individual transmissions, leading to a higher success rate. The success rate against Viden reaches more than 75% with the corruption of *only* control bits (i.e., PREP = 0.75 byte), and 95% with just three bytes of PREP (Figure 12). Similarly, the success rate against Scission reaches more than 50% by corrupting only control bits, and 76% with three bytes of PREP (Figure 13). We also observe similar performance of DUET against different ML algorithms (for Scission) as illustrated in Figure 14. This observation validates that DUET's success rate is not limited by any specific characteristic of the ML algorithm employed in VIDS, but by the attacker's timing-accuracy and the PREP length in the victim's messages.

DUET fails whenever the attacker and accomplice fail to transmit simultaneously in Stage 2 due to imperfect timing accuracy (Table III). Hence, DUET's lower success rate on the real cars, in comparison with the testbed, can be attributed to

Fig. 15: Demonstration of Scission failing to raise an alarm while DUET spoofs RPM messages in a real vehicle.

the lower timing accuracy due to higher busload and transmission jitter. On the one hand, the highest success rate against Viden reaches 100% since Viden makes its decision by taking an aggregate of samples received over eight messages. On the other hand, as Scission detects the source of each message individually, the highest success rate against Scission is limited by the value of p_a shown in Table III, i.e., 89% for ExpCar-1, 85% for ExpCar-2: Bus-1, and 90% for ExpCar-2: Bus-2. Nevertheless, the success rates of DUET against the VIDS (shown in Figures 12 and 13) render the VIDS ineffective in practice, specifically in safety-critical applications.

Evading VIDS Alarm. Although DUET's per-message success rate against a VIDS may not be 100%, DUET can still evade it successfully. This is because a real-world VIDS must consider encountering messages that are corrupted by electromagnetic interference and temperature changes, and must minimize the resultant false alarms. For example, Scission employs a mechanism with an *alarm counter* which is incremented by four for each suspicious (wrongly classified) message and decremented by one for each trustworthy (correctly classified) message, raising an alarm if the alarm counter exceeds 200. Such a mechanism of alarm-raising over aggregate traffic makes it *even easier* for DUET to evade Scission, as any per-message success rate of more than 80% would keep the alarm counter near its expected value of zero.

As such, DUET does not cause any VIDS alarm on three of the evaluated platforms, and even on the fourth one (ExpCar-2: Bus-1), Scission will raise an alarm only after an average of 1000 $\left(\approx \frac{200}{(1-0.76)\cdot 4 - 0.76\cdot 1}\right)$ spoofed messages with three bytes of PREP (Figure 13). For a brake control message with a 10 ms period, this translates into a sizable 10 seconds of alarm delay, which is sufficient to accomplish malicious activities, such as disabling brakes to cause an accident [32]. In Figure 15, we illustrate that the DUET duo successfully impersonate the Engine Control Unit of the ExpCar-1 and forge the RPM messages, but Scission fails to raise any alarm because the alarm counter never exceeds 200.

VIII. DEFENSE AGAINST DUET

We introduce a defense that can not only prevent DUET, and but also safeguard CAN against all other potential attacks that exploit the voltage corruption tactic.

DUET-Aware VIDS. DUET follows a powerful masquerade attack strategy that evades all existing VIDS by corrupting the voltage samples of only a few bits in the victim's messages. It might be possible, however, to build a DUET-aware VIDS. For instance, to detect the voltage corruption caused by DUET, a VIDS may be modified to record the fine-grained changes in the voltage values between message bits or detect the presence of a bimodal distribution of voltage samples. Unfortunately, changes in the voltage values may also be caused by non-malicious electromagnetic interference sources typically present in an automobile [9], [42]. The bimodal distribution can also be observed when two benign messages with similar IDs overlap over significant portions of their arbitration fields. As such, the modified VIDS must find a tedious balance between detecting DUET and avoiding false alarms.

Therefore, we decide to avoid the "attack vs. IDS" arms race and instead address the root cause that makes DUET feasible. DUET exploits a major deficiency of the CAN protocol: Each message on CAN is allotted a unique identifier (ID) which remains the same for its lifetime. This static nature of the message ID enables priority scheduling and deterministic latency for messages on the bus. It also ensures robust arbitration on simultaneous transmissions of two different messages by two ECUs. However, from an attacker's perspective, this means that the same ID is set in the arbitration fields of all periodic messages transmitted by the victim. Such predictability allows the attacker to craft its attack message with the same ID, and perform simultaneous transmission with the victim. This further facilitates the attacker's unrestricted maneuverability in corrupting the data bits of the victim's message, and in manipulating the victim's voltage fingerprint.

We propose a novel lightweight defense called Randomized Identifier Defense (RAID) which mitigates the aforementioned deficiency of CAN and complements the VIDS to detect/prevent all ECU masquerade attacks including DUET. Different from (and orthogonal to) VIDS, RAID *prevents* DUET by restricting the attacker's fundamental ability to predict all the ID bits in the victim's message.

A. RAID *Design*

While most of the automotive ECUs utilize the standard frame format by default (Table II), some ECUs may employ the extended frame format to send their messages on the CAN bus. To defend both types of ECUs against masquerade attacks, RAID establishes a unique protocol dialect which is "spoken" by all ECUs on the CAN during the VIDS retraining mode. Under RAID, the VIDS is trained using dedicated messages with the standard frame format [10]. Then, whenever the VIDS undergoes retraining, every sender ECU upgrades these standard frames to extended frames (Figure 2). The 11-bit ID-A field of the standard frame is mapped to the ID-A field of the extended frame. Then, the ID-B field of the extended frame is set as an 18-bit nonce generated using a cryptographically secure pseudo-random bit generator (PRBG) [38]. At receiver ECUs, while the ID-A field is utilized for the identification of messages, the bits in the ID-B field are discarded.

Although RAID makes VIDS retraining mode obvious to the attacker, the protocol dialecting in RAID ensures that the sender randomizes the arbitration fields of its messages without hindering the receiver's ability to interpret the messages. Moreover, the protocol dialecting ensures that the attacker cannot

easily guess the ID bits in the victim's messages and overlap its attack message beyond them. This way, RAID inhibits the voltage corruption tactic and prevents the corruption of victim's voltage fingerprint right in Stage 1 of DUET.

B. RAID *Evaluation*

We now analyze and evaluate RAID's effectiveness against DUET and its impact on CAN during the VIDS retraining. The RAID code can be accessed at [1].

Impact on Success Rate of DUET. In Stage 1 of DUET, the attacker needs to correctly predict the bits in the arbitration field (which includes ID-A and ID-B fields) of the victim's message utilized for the VIDS retraining. With RAID's 18-bit randomness in the ID-B field, the probability that the attacker can find a collision between its guess and the complete arbitration field of the victim's message is 2^{-18}. Thus, the probability of successfully corrupting the voltage fingerprint of the victim during the retraining of the VIDS which learns each fingerprint over N messages, is $2^{-18 \cdot N}$. This probability is negligibly small for a large N. For instance, RAID reduces the success rate of DUET against Scission (with $N = 200$) to *zero percent* (from at least 76%) on all evaluation platforms.

Computation Overhead. The implementation of RAID on a commodity ECU will incur some computation cost in generating the pseudo-random nonce. We estimate this cost with the running time of the built-in PRBG in the Arduino UNO board [2], a single invocation of which is able to generate 32 random bits in 50 μs. We note that PRBG can be executed during the ECU idle time, and the results can be stored and used when needed (i.e., during the VIDS retraining). Hence, RAID effectively produces negligible computation overhead during actual message transmission.

Communication Overhead. Since a standard frame is shorter than its extended version, RAID suffers communication overhead during VIDS retraining. RAID increases the busload which we measure via simulation, based on *real* traffic traces collected from the three CAN buses in our vehicles. On the three buses, RAID increases the busload by 13% (61% to 74%, for ExpCar-2: Bus-1), 7% (34% to 41%, for ExpCar-2: Bus-2), and 7% (35% to 42%, for ExpCar-1 CAN bus) respectively. RAID also increases the end-to-end message latency by 50 μs (25 bit-periods on a 500 kbps CAN bus). We note that according to the CAN schedulability analysis [13], the latency encountered by CAN messages remains within their deadlines when the busload is below 80%, and the jitter is below 100 μs. Hence, with RaID deployed on our CAN buses, the increase in the busload and the message latency remain well within these acceptable values of the general automotive tolerances.

C. Comparison with Other Potential Defenses

We highlight the effectiveness of RAID by comparing it with the following potential defenses against DUET.

Transmission Time Randomization. In Stage 1 of DUET, the accomplice enables the time synchronization between the victim and attacker as discussed in Section IV-A2. As such, the simultaneous transmission of the victim and attacker might be deemed preventable by randomizing the transmission time of the victim. However, the accomplice may readily counter such defense. We note that while the victim may randomly choose the transmission time, it must remain within the message periodicity to deliver the designed real-time responsiveness [13]. To launch DUET in this case, the accomplice will transmit multiple (instead of just one) back-to-back preceded ID messages, which will force the victim to wait for the completion of these messages. This process will re-enable synchronization of the attack message with the victim's message. Moreover, transmission time randomization adversely affects the priority scheduling of messages on CAN and may result in significantly degraded worst-case real-time response of the system [13]. RAID does not suffer from such adverse effects since the message priority is preserved by mapping the bits of the ID-A field in the standard frame to those in the extended frame.

TABLE IV: Computation cost (in μs) for MAC schemes.

Algorithm	Hashing cost (per byte)	Finalization cost (per operation)	Total cost (8-byte data)
SHA-2	44.75	2895.89	3253.89
SHA-3	60.13	8099.97	8581.01
Blake	20.40	1317.48	1480.68
GHASH	74.68	9.1	606.54
Poly1305	24.67	463	857.56
AES (CBC-MAC)	-	-	488

Message Authentication Code (MAC). The automotive CAN employs resource-constrained ECUs and remains bandwidth-constrained. Therefore, there are four fundamental challenges in developing a practical MAC scheme for preventing ECU masquerade attacks including DUET: (1) The scheme must have a lightweight and secure key agreement protocol so that the keys shared among ECUs can be refreshed periodically [21]. (2) The scheme must have a counter synchronization mechanism among ECUs so that the shared keys can be utilized over a sufficiently long duration for a large number of communicated messages. (3) The scheme must ensure that the computation cost to generate the MAC must be low because it directly impacts the end-to-end message latency and the response time of ECUs. (4) The scheme must provide "satisfactory" cryptographic strength even with a short MAC which can fit into the 64-bit payload of a CAN message. Unfortunately, the MAC schemes presented in the existing literature fail to address these challenges comprehensively. Hence, these schemes are either not secure or impractical to implement with the existing CAN [16], [19].

To further explore the feasibility of a MAC scheme in CAN, we utilize the Arduino UNO board to evaluate multiple MAC schemes [33]. Our evaluation includes hash-based schemes [37] as well as block cipher-based schemes [17]. For the hash algorithms, the total computation cost is the sum of the hashing cost and finalization cost. For each hash algorithm, although the hashing cost increases with the length of the message data, the finalization cost is fixed per operation of the algorithm, regardless of the number of data bytes. Here, we consider a CAN message with eight data bytes to calculate the total computation cost of generating a MAC with each algorithm using a 128-bit key. We present our results in Table IV. We observe that unlike RAID, computationally-intensive cryptographic operations used in the considered schemes for the generation and verification of MACs bring a

significant computation overhead for the resource-constrained ECUs. For instance, computing a MAC with *AES (CBC-MAC)* and *SHA*-3 takes 0.49 ms and 8.56 ms, respectively.

Moreover, the communication overhead of transmitting a MAC on the bandwidth-limited CAN bus is non-trivial because the length of the payload in each CAN message is limited to only 64 bits. Specifically, a scheme that communicates 128-bit MACs in two chunks may lead to a 200% increase in the busload [17]. Such a high busload may lead to consequences such as delayed car functions and communication faults [50]. This observation implies that the end-to-end message latency in a MAC scheme is significantly higher than 0.05 ms encountered in RAID. Finally, unlike the MAC scheme, the ECUs in RAID do not need to follow any key agreement protocol or counter synchronization mechanism. Hence, we assert that a VIDS complemented with RAID is a more practical solution than a MAC scheme for protecting ECUs against DUET.

IX. DISCUSSION

Relevance and Generalisability. Prior efforts [6], [24], [32] have illustrated various techniques for compromising in-vehicle ECUs. We build upon these efforts and present a novel ECU masquerade attack, DUET, which follows an APT-style CAN attack strategy and evades any retraining-based VIDS by poisoning its training set. Further, the CAN characteristics that enable our attack tactic are fundamental to the CAN protocol and not specific to any vehicle model/maker. The various characteristics leveraged by DUET are also native in all CAN buses, e.g., the periodicity of CAN messages and predictability of message contents are commonly observed, as confirmed by our study of CAN traffic from five different vehicles (Table II).

Secure VIDS Retraining. To provide a secure retraining mode for the VIDS, Kneib et al. propose to utilize message authentication along with other existing defenses against training set poisoning attacks [22]. We note that DUET will successfully manipulate the fingerprint of the victim's messages *with* authentication since the attacker can still corrupt voltage samples in the payload. Also, existing defenses against poisoning attacks utilize techniques to remove outliers in the training samples [20], [29]. Since DUET can poison a significant portion (>50%) of the training samples (as presented in Table III), the poisoned samples are no longer outliers and cannot be eliminated by such defenses. Moreover, appending MACs to messages does not change the PREP bits, and hence cannot defeat DUET. Encryption of payload – though considered impractical – may limit the length of the PREP to 0.75 byte, i.e., only the bits in the control field. As shown in Figures 12 and 13, this may lower the success rate of DUET, but *cannot* completely prevent DUET. Further, a VIDS may try to detect DUET by differentiating the bimodal distribution of victim's voltage samples from the unimodal distribution of a typical ECU's samples. Unfortunately, this method will result in a large number of false positives because a bimodal distribution can also be observed when two messages with similar (not same) IDs transmitted by two *benign* ECUs overlap over a significant portion of their arbitration fields, which is not unusual in modern automotive CANs.

Evading MIDS and CIDS. DUET is designed to evade the state-of-the-art VIDS defenses. Besides, DUET can be integrated with other attack strategies, e.g., (1) the strategy presented by Miller and Valasek [31] against MIDS [34] that analyzes the message content to detect an anomaly, and (2) the attack method presented by Sagong et al. [43] against CIDS [8] that analyzes the clock-skew-based fingerprint to detect a masquerade attack. In other words, before DUET, MIDS and CIDS – *but not* VIDS – have already been shown defeat-able hence are *not* the focus of DUET.

X. RELATED WORK

Attack. Prior efforts [6], [24], [31], [32], [35], [36] have shown that the increasing number of automotive CAN ECUs are exhibiting various remote attack surfaces due to their connectivity with the widespread IoT networks. A malicious attacker can infiltrate the CAN bus by compromising any of these vulnerable ECUs and launch a variety of attacks against ECUs which cannot be remotely compromised. For instance, Checkoway et al. demonstrated that the Telematics Control Unit could be compromised through USB, Bluetooth, WiFi, vehicle-to-vehicle, or cellular connections [6]. Miller et al. exploited the long-range wireless communication unit used for the road-side assistance and crash reporting to obtain full control of the vehicle [32]. Nie et al. were able to concurrently compromise the gateway ECU, body control unit, and autopilot ECU of a Tesla car [36]. DUET builds upon these efforts and presents a powerful ECU masquerade attack strategy.

The VIDS represents the state-of-the-art CAN defense which exploits the message's voltage fingerprinting on the bus to identify its sender ECU [9], [10], [14], [22], [23]). Since a *lone* attacker ECU cannot controllably alter its voltage fingerprint ingrained in its hardware characteristics, the VIDS has been shown to robustly detect all known ECU masquerade attacks. However, the VIDS must be retrained to update the ECU fingerprints that vary with time due to changing environmental factors, firmware updates, and aging effects [9]. This enables DUET to follow a two-stage training set poisoning-based attack strategy to successfully evade the VIDS. DUET employs a duo of attacker and accomplice ECUs manipulating the victim's voltage fingerprint during the retraining mode and then impersonating it during the operation mode. Foruhandeh et al. [14] also noted that the retraining makes VIDS vulnerable to training set poisoning. Moreover, they presented a hill-climbing-style attack against Viden [9] that employs online retraining mechanism. In contrast, DUET is among the first work to successfully evade all known VIDS irrespective of the (online, incremental or periodic) retraining mechanism.

Defense. Various cryptographic solutions [16], [19] have been proposed to secure CAN by authenticating message payloads. Unfortunately, cryptographic solutions remain impractical as vehicular CAN employs resource-constrained ECUs and remains bandwidth-constrained. A countermeasure using hardware-based identifiers [46] will hinder ECUs' reconfigurability, making ECUs specific to a car's year-make-model, hence increasing their manufacturing cost. More practical and deployable CAN defenses favor signature and fingerprint-based IDS, such as MIDS [34], [44], CIDS [8] and VIDS [9], [10], [14], [22], [23]). While MIDS and CIDS have been shown vulnerable to impersonation attacks [8], [43], VIDS are still considered the state-of-the-art defenses. However, as shown in

this paper, VIDS are comprehensively beaten – for the first time, by DUET. We note that a trivial countermeasure such as retraining VIDS using only random payload bits would not be generic as extracting voltage samples corresponding to specific bits is infeasible in some VIDS (e.g., Viden) due to their low sampling rates. Foruhandeh et al. [14] presented the VIDS called SIMPLE that mitigates the frequency of retraining by updating the learned fingerprints based on the changes in the supply voltage and surrounding temperature. As such, DUET cannot evade SIMPLE if it does not require retraining. However, it is difficult to precisely model all factors affecting the ECU fingerprints in practice, e.g., those originating from firmware updates and aging effects. Hence, RAID (our defense) goes beyond modifying the existing VIDS and provides an orthogonal, lightweight, and effective defense that not only *prevents* (instead of just detects) DUET, but also – in conjunction with VIDS – detects or prevents all known masquerade attacks.

XI. CONCLUSION

The voltage fingerprint of an ECU is considered "immutable" in prior research and hence widely utilized in the state-of-the-art VIDS defenses to detect ECU masquerade attacks. In this paper, we have demonstrated the power of the "attacker + accomplice" duo– in comparison with existing "lone-attacker"–in corrupting an ECU's voltage fingerprint. We have presented DUET, a stealthy, two-stage ECU masquerade attack strategy that successfully evades all existing VIDS regardless of the ML algorithms utilized by them. In future, VIDS can be improved by developing mechanisms to differentiate between benign and malicious bimodal distributions of voltage samples while limiting the potential false alarms. However, by proposing the protocol dialecting-based RAID defense against DUET, we advocate the development of orthogonal (to IDS), cost-effective defenses that break away from the "attack vs. IDS" arms race to protect ECUs against masquerade attacks.

ACKNOWLEDGMENTS

We thank the anonymous reviewers and Dr. Alexandra Dmitrienko for their valuable comments and suggestions. We also thank Drs. Ryan Gerdes and Ming Li for their timely technical feedback. This work was supported in part by Office of Naval Research (ONR) under Grant N00014-18-1-2674. Any opinions, findings, and conclusions in this paper are those of the authors and do not necessarily reflect the views of the ONR.

REFERENCES

[1] Anonymous, "Duet and RaID: Supplementary files," https://github.com/CAN-Bus-Duet/CAN-Bus-Duet, 2020, [Online; accessed March 1, 2020].

[2] Arduino, "Random function," https://www.arduino.cc/reference/en/language/functions/random-numbers/random/, 2019, [Online; accessed March 1, 2020].

[3] R. Baker and I. Martinovic, "Losing the car keys: Wireless PHY-layer insecurity in EV charging," in *USENIX Security Symposium*, 2019, pp. 407–424.

[4] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, 2012, pp. 1467–1474.

[5] R. Bosch, "CAN specification - Version 2.0," 1991.

[6] S. Checkoway, D. Mccoy, B. Kantor *et al.*, "Comprehensive experimental analyses of automotive attack surfaces," in *USENIX Security Symposium*, 2011, pp. 77–92.

[7] K. T. Cho and K. G. Shin, "Error handling of in-vehicle networks makes them vulnerable," in *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2016, pp. 1044–1055.

[8] K. T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection," in *USENIX Security Symposium*, 2016, pp. 911–927.

[9] K. T. Cho and K. G. Shin, "Viden: Attacker identification on in-vehicle networks," in *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017, pp. 1109–1123.

[10] W. Choi, K. Joo, H. J. Jo *et al.*, "VoltageIDS: Low-level communication characteristics for automotive intrusion detection system," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 8, pp. 2114–2129, 2018.

[11] Comma.ai, "opendbc: Democratize access to car decoder rings," https://github.com/commaai/opendbc, 2019, [Online; accessed March 1, 2020].

[12] Crash Reconstruction Research Consortium, "CAN traffic data," http://tucrrc.utulsa.edu, 2019, [Online; accessed March 1, 2020].

[13] R. I. Davis, A. Burns, R. J. Bril, and J. J. Lukkien, "Controller area network (CAN) schedulability analysis: Refuted, revisited and revised," *Real-Time Systems*, vol. 35, no. 3, pp. 239–272, 2007.

[14] M. Foruhandeh, Y. Man, R. Gerdes *et al.*, "SIMPLE: Single-frame based physical layer identification for intrusion detection and prevention on in-vehicle networks," in *Annual Computer Security Applications Conference (ACSAC)*, 2019, pp. 229–244.

[15] D. Frassinelli, S. Park, and S. Nürnberger, "I know where you parked last summer: Automated reverse engineering and privacy analysis of modern cars," in *IEEE Symposium on Security and Privacy (S&P)*, 2020, pp. 1184–1198.

[16] B. Groza and P. Murvay, "Security solutions for the controller area network: Bringing authentication to in-vehicle networks," *IEEE Vehicular Technology Magazine*, vol. 13, no. 1, pp. 40–47, 2018.

[17] B. Groza, S. Murvay, A. V. Herrewege, and I. Verbauwhede, "LiBrA-CAN: A lightweight broadcast authentication protocol for controller area networks," in *International Conference on Cryptology and Network Security*, 2012, pp. 185–200.

[18] T. Hoppe, S. Kiltz, and J. Dittmann, "Security threats to automotive CAN networks–practical examples and selected short-term countermeasures," *Reliability Engineering & System Safety*, vol. 96, no. 1, pp. 11–25, 2011.

[19] Q. Hu and F. Luo, "Review of secure communication approaches for in-vehicle network," *International Journal of Automotive Technology*, vol. 19, no. 5, pp. 879–894, 2018.

[20] M. Jagielski, A. Oprea, B. Biggio *et al.*, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *IEEE Symposium on Security and Privacy (S&P)*, 2018, pp. 19–35.

[21] S. Jain and J. Guajardo, "Physical layer group key agreement for automotive controller area networks," in *International Conference on Cryptographic Hardware and Embedded Systems (CHES)*, 2016, pp. 85–105.

[22] M. Kneib and C. Huth, "Scission: Signal characteristic-based sender identification and intrusion detection in automotive networks," in *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2018, pp. 787–800.

[23] M. Kneib, O. Schell, and C. Huth, "EASI: Edge-based sender identification on resource-constrained platforms for automotive networks," in *Network and Distributed System Security Symposium (NDSS)*, 2020, pp. 1–16.

[24] K. Koscher, A. Czeskis, F. Roesner *et al.*, "Experimental security analysis of a modern automobile," in *IEEE Symposium on Security and Privacy (S&P)*, 2010, pp. 447–462.

[25] S. Kramer, D. Ziegenbein, and A. Hamann, "Real world automotive benchmarks for free," in *International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS)*, 2015.

[26] A. Lima, F. Rocha, M. Völp, and P. E. Veríssimo, "Towards safe and secure autonomous and cooperative vehicle ecosystems," in *Proceedings of the 2nd ACM Workshop on Cyber-Physical Systems Security and Privacy*, 2016, pp. 59–70.

[27] M. Marchetti and D. Stabili, "READ: Reverse engineering of automotive data frames," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 4, pp. 1083–1097, 2019.

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

[29] S. Mei and X. Zhu, "Using machine teaching to identify optimal training-set attacks on machine learners," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015, pp. 2871–2877.

[30] Microchip, "MCP2515: Stand-alone CAN controller with SPI interface," http://ww1.microchip.com/downloads/en/devicedoc/21801e.pdf, 2007, [Online; accessed March 1, 2020].

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

[32] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, vol. 2015, p. 91, 2015.

[33] P.-S. Murvay, A. Matei, C. Solomon, and B. Groza, "Development of an AUTOSAR compliant cryptographic library on state-of-the-art automotive grade controllers," in *IEEE International Conference on Availability, Reliability and Security (ARES)*, 2016, pp. 117–126.

[34] M. Müter, A. Groll, and F. C. Freiling, "A structured approach to anomaly detection for in-vehicle networks," in *Sixth International Conference on Information Assurance and Security (IAS)*, 2010, pp. 92–98.

[35] S. Nie, L. Liu, and Y. Du, "Free-fall: Hacking Tesla from wireless to CAN bus," *Briefing, Black Hat USA*, 2017.

[36] S. Nie, L. Liu, Y. Du, and W. Zhang, "Over-the-air: How we remotely compromised the gateway, BCM, and autopilot ECUs of Tesla cars," *Briefing, Black Hat USA*, 2018.

[37] S. Nürnberger and C. Rossow, "–vatican–vetted, authenticated can bus," in *International Conference on Cryptographic Hardware and Embedded Systems*, 2016, pp. 106–124.

[38] P. P. Lopez, E. S. Millan, J. C. V. Lubbe, and L. A. Entrena, "Cryptographically secure pseudo-random bit generator for RFID tags," in *International Conference for Internet Technology and Secured Transactions (ICITST)*, 2010, pp. 1–6.

[39] M. D. Pesé, T. Stacer, C. A. Campos *et al.*, "LibreCAN: Automated CAN message translator," in *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2019, pp. 2283–2300.

[40] J. Petit and S. E. Shladover, "Potential cyberattacks on automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 546–556, 2014.

[41] Philips, "SJA1000: Stand-alone CAN controller," https://www.nxp.com/docs/en/data-sheet/SJA1000.pdf, 2000, [Online; accessed March 1, 2020].

[42] M. Pous, A. Atienza, and F. Silva, "EMI radiated characterization of an hybrid bus," in *10th International Symposium on Electromagnetic Compatibility*, 2011, pp. 208–213.

[43] S. U. Sagong, X. Ying, A. Clark *et al.*, "Cloaking the clock: Emulating clock skew in controller area networks," in *Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, 2018, pp. 32–42.

[44] 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 *International Conference on Information Networking (ICOIN)*, 2016, pp. 63–68.

[45] N. Šrndic and P. Laskov, "Practical evasion of a learning-based classifier: A case study," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2014, pp. 197–211.

[46] V. E. Von Bokern, P. Goel, S. Schrecker, and N. M. Smith, "Hardware-based device authentication," US Patent 8,955,075, issued Feb. 10, 2015.

[47] H. Wen, Q. A. Chen, and Z. Lin, "Plug-N-pwned: Comprehensive vulnerability analysis of OBD-II dongles as a new over-the-air attack surface in automotive IoT," in *USENIX Security Symposium*, 2020.

[48] H. Wen, Q. Zhao, Q. A. Chen, and Z. Lin, "Automated cross-platform reverse engineering of CAN bus commands from mobile apps," in *Network and Distributed System Security Symposium (NDSS)*, 2020.

[49] 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 Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 993–1006, 2015.

[50] G. M. Zago and E. P. de Freitas, "A quantitative performance study on CAN and CAN FD vehicular networks," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 5, pp. 4413–4422, 2018.

TABLE V: Message periodicity in experimental vehicles.

2011 ExpCar-1 (50 messages)

Period (ms)	**9**	**10**	**12.5**	**18**	**25**	**30**
No. of messages	3	1	4	4	4	3
Period (ms)	**50**	**100**	**250**	**500**	**1000**	**5000**
No. of messages	4	9	3	3	11	1

2013 ExpCar-2: Bus-1 (88 messages)

Period (ms)	**10**	**12.5**	**20**	**25**	**50**	**100**
No. of messages	6	10	6	14	8	16
Period (ms)	**250**	**500**	**1000**	**5000**		
No. of messages	4	6	17	1		

2013 ExpCar-2: Bus-2 (27 messages)

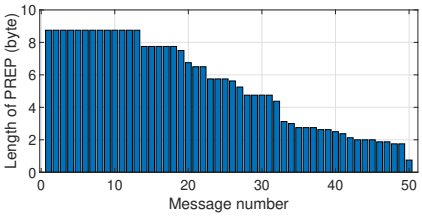
Period (ms)	**10**	**20**	**50**	**100**
No. of messages	10	5	2	10

APPENDIX A
CAN TRAFFIC ANALYSIS

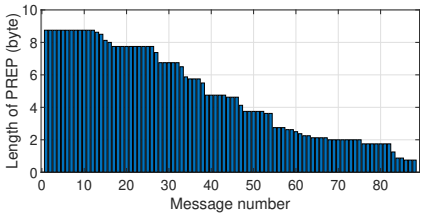
DUET exploits three key characteristics of CAN messages: (1) static identifiers, (2) periodic messages, and (3) predictable message content. In automobiles, messages (with information related to engine, brake, steering and other critical equipment) are exclusively transmitted on the high-speed CAN bus in the standard format. Conventionally, each particular type of message always contains the same ID. This characteristic is motivated by the safety-critical requirement of providing robust message arbitration and minimizing the worst-case delay (with theoretical guarantees) in the communication of messages [13]. Further, the messages on the CAN are periodic because the fine granularity of periodic message communication is required for making safety-critical collaborative decisions (e.g., control of accelerator, brake and steering) in the vehicles. Hence, the next time-of-transmission of a periodic message can be readily estimated by knowing the current time-of-transmission of the message on the bus. The third key characteristic is the existence of PREP in the CAN messages which makes the voltage corruption tactic in DUET feasible. We validate these characteristics by analyzing the CAN traffic in two experimental and three non-experimental vehicles.

A. Experimental Vehicles

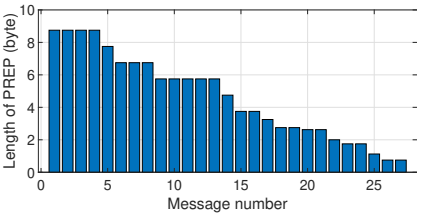
We have performed extensive experiments on the CAN buses of two vehicles: 2011 ExpCar-1 and 2013 ExpCar-2. We note that while the non-periodic messages (e.g., door lock/unlock) are usually communicated on the *low-speed secondary* CAN bus or local interconnect network, most of the safety-critical messages (e.g., brakes, steering, and engine speed) are communicated periodically on the *high-speed primary* CAN bus. We record and analyze the CAN traffic by connecting our experimental setup (Figure 8) to the primary CAN buses of these vehicles using their OBD II ports. While we could access only one primary CAN bus in the ExpCar-1,



(a) 2011 ExpCar-1.

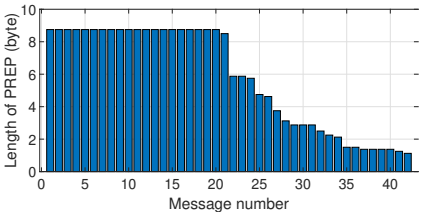


(b) 2013 ExpCar-2: Bus-1.

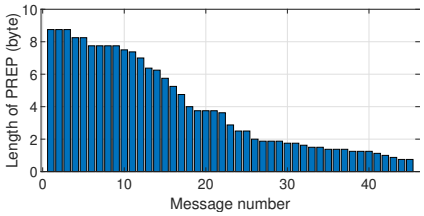


(c) 2013 ExpCar-2: Bus-2.

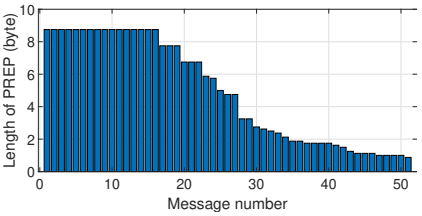
Fig. 16: Length of PREP in the CAN messages of the experimental vehicles.



(a) 2012 Toyota Camry.



(b) 2012 Honda Civic.



(c) 2010 Dodge Ram.

Fig. 17: Length of PREP in the CAN messages of non-experimental vehicles.

TABLE VI: Message periodicity in non-experimental vehicles.

2012 Toyota Camry (42 messages)

Period (ms)	**10**	**20**	**30**	**100**	**200**	**300**
No. of messages	4	3	5	1	2	2
Period (ms)	**500**	**1000**	**2000**	**5000**		
No. of messages	3	20	1	1		

2012 Honda Civic (45 messages)

Period (ms)	**10**	**20**	**40**	**100**	**200**	**300**
No. of messages	11	7	6	11	2	8

2010 Dodge Ram (51 messages)

Period (ms)	**10**	**20**	**50**	**60**	**100**	**200**
No. of messages	3	17	1	1	11	1
Period (ms)	**300**	**500**	**1000**	**2000**		
No. of messages	3	3	7	4		

two CAN buses are accessible in the ExpCar-2. The traffic traces from these vehicles are available at [1].

While there are 50 messages on the ExpCar-1 CAN bus, there are 88 messages on the Bus-1 and 27 messages on the Bus-2 in the ExpCar-2. All messages in these three CAN buses are transmitted in the standard frame format with static identifiers. Also, all message in these two vehicles are periodic with their periodicity presented in Table V. Further, each message with a specific ID is transmitted with the same length of the message payload, which means that the bits in the control field remain the same. Additionally, we observe the message payloads contain constants, counters and predictable contents. Figure 16 presents the length of PREP for messages on the three buses. We observe that most of the messages on these CAN buses have at least one byte of PREP.

B. Non-Experimental Vehicles

We also analyze the CAN traffic data (published at [12] by other independent researchers) of three other vehicles: 2012 Toyota Camry, 2012 Honda Civic, and 2010 Dodge Ram. For the convenience of readers, these traces of CAN traffic are also made available at [1]. While the Camry has 42 messages on its CAN bus, there are 45 messages in the Civic. All the messages in the Camry and Civic are periodic with their respective periodicity shown in Table VI. In the Ram, we observe 51 periodic messages and 4 non-periodic messages. Also, each message in these three vehicles is transmitted in the standard frame format. Further, Figure 17 illustrates the distribution of lengths of PREP in the CAN messages of the three non-experimental vehicles. We observe that most of the messages have at least one byte of PREP. Additionally, when compared with the Civic, the Camry and the Ram have significantly higher number of messages in which all data bits can be readily predicted resulting in the PREP of 8.75 bytes.