SIMPLE: Single-Frame based Physical Layer Identification for Intrusion Detection and Prevention on In-Vehicle Networks

Mahsa Foruhandeh[*]
Virginia Tech
mfhd@vt.edu

Yanmao Man[*]
University of Arizona
yman@email.arizona.edu

Ryan Gerdes
Virginia Tech
rgerdes@vt.edu

Ming Li
University of Arizona
lim@email.arizona.edu

Thidapat Chantem
Virginia Tech
tchantem@vt.edu

ABSTRACT

The Controller Area Network (CAN) is a bus standard commonly used in the automotive industry for connecting Electronic Control Units (ECUs) within a vehicle. The broadcast nature of this protocol, along with the lack of authentication or strong integrity guarantees for frames, allows for arbitrary data injection/modification and impersonation of the ECUs. While mitigation strategies have been proposed to counter these attacks, high implementation costs or violation of backward compatibility hinder their deployment. In this work, we first examine the shortcomings of state-of-the-art CAN intrusion detection and identification systems that rely on multiple frames to detect misbehavior and attribute it to a particular ECU, and show that they are vulnerable to a Hill-Climbing-style attack. Then we propose SIMPLE, a real-time intrusion detection and identification system that exploits physical layer features of ECUs, which would not only allow an attack to be detected using a single frame but also be effectively nullified. SIMPLE has low computational and data acquisition costs, and its efficacy is demonstrated by both in-lab experiments with automotive-grade CAN transceivers as well as in-vehicle experiments, where average equal error rates of close to 0% and 0.8985% are achieved, respectively.

CCS CONCEPTS

• **Security and privacy → Intrusion detection systems**; **Hardware attacks and countermeasures**; • **Hardware →** *Buses and high-speed links*; *Networking hardware.*

KEYWORDS

Physical Layer Identification, Controller Area Networks, Electronic Control Units, Hill-climbing Attacks

[*]M. Foruhandeh and Y. Man are co-first authors.

ACM Reference Format:
Mahsa Foruhandeh, Yanmao Man, Ryan Gerdes, Ming Li, and Thidapat Chantem. 2019. SIMPLE: Single-Frame based Physical Layer Identification for Intrusion Detection and Prevention on In-Vehicle Networks. In *2019 Annual Computer Security Applications Conference (ACSAC '19), December 9–13, 2019, San Juan, PR, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3359789.3359834

1 INTRODUCTION

The Controller Area Network (CAN) bus is the de facto industry standard for in-vehicle networks used in modern vehicles for connecting Electronic Control Units (ECUs). ECUs are embedded systems with specific automotive-related duty, such as engine control, braking, etc. ECUs are involved with safety-critical tasks, such as braking and engine control, which require information that is communicated via the CAN bus, consequently the safety of the passengers is directly dependent upon the security of the bus [16, 52]. It has been demonstrated should an attacker gain access to the CAN bus, forged messages can be sent that affect the safe operation of the vehicle (e.g., causing the vehicle to accelerate [12], stopping the engine, disabling the brakes, selectively braking [30], or disabling the transmission [19]). The connection can be through direct connection [30] or via a remotely compromised ECU [5].

Based on existing studies, the confidentiality of CAN messages is not strictly necessary to provide safe operation, whereas authentication and integrity are essential. A CAN frame can only accommodate eight bytes of both data and cryptographic information (Sec. 2.1), hence, providing authentication and integrity via message authentication codes (MACs) or digital signatures is not a straightforward proposition. In [49] an out-of-band channel (CAN+ [55]) is leveraged to transmit authentication information; in [39] a delayed authentication scheme is proposed that uses multiple frames to generate a compound MAC; and variable-length MACs are used in [43] to offer protection commensurate with a message's criticality. In short, existing techniques are either insecure or computationally intensive which makes them incapable of a reliable authentication on a frame-by-frame basis.

As strong authentication guarantees cannot be provided for legacy CAN, intrusion/anomaly detection systems (I/ADS) have been proposed that would at least allow for appropriate countermeasures to be taken in the event of an attack (e.g., ignoring suspect messages or putting the car into a safe state) [20, 21, 31, 37, 38, 44]. Of particular relevance to the current work is the subset of IDS that leverage, broadly speaking, device fingerprinting techniques based on differentiating devices according to either timing [8, 35] or the physical-layer characteristics of frames, such as voltage [9, 10, 36]. However, most approaches rely on multiple frames sent by an ECU to evaluate and update the fingerprints.

In this paper, we first demonstrate that state-of-the-art device fingerprinting-based intrusion detection and identification systems (e.g., [36], [8], and [9] etc.) are fundamentally vulnerable to *Hill-climbing-style* attacks, because they either entirely or partially depend on multiple frames to identify the sender of a frame. Due to the multi-frame dependence, a Hill-climbing-style attacker is able to exploit a compromised ECU to impersonate another ECU without being detected *or* identified. The attack is achieved by carefully injecting an increasing amount of malicious messages among legitimate messages, so as to gradually shift the profile of the target ECU toward the attacker's. Since during consecutive time periods the profile only shifts slightly, the attack remains undetected or unidentified. Ultimately, the attacker ECU will be able to inject a majority, or replace all, of the frames sent by the target ECU with its own. Our attack can be regarded as a type of online data poisoning attack against machine learning systems [28] (especially, for classification) which acquire/update training data in an online manner.

Motived by this attack, we propose SIMPLE, a SIngle-fraMe based Physical-LayEr identification solution to detect intrusion and identify the source of each single CAN frame that is transmitted on a bus by a specific ECU regardless of its message ID. We extract unique voltage-based features (*fingerprints*) in the time-domain from each individual CAN frame transmitted and contribute to an ECU. Unlike existing multi-frame IDS systems, SIMPLE performs secure updates of training data by modelling and compensating for the changes in environment and operating conditions (e.g., temperature and supply voltage). Since fingerprinting in SIMPLE is done on a *per-frame* basis and is very computationally lightweight, the detection can finish even before a frame's transmission ends, thus enabling real-time prevention of intrusion attacks by invalidating the spurious frame before it takes effect on the vehicle. SIMPLE is a single-frame based intrusion detection and identification system that (to the best of our knowledge) for the first time achieves attack prevention with secure updates.

Our main contributions are summarized as follows:

- We demonstrate that a Hill-climbing-style attack can defeat *multi-frame* based intrusion detection and/or identification systems, in particular for vehicular CANs. We validate the effectiveness of our attack against two existing IDSs: physical layer-based Viden [9] and clock-based IDS CIDS [10].
- We propose SIMPLE, a *single-frame* based physical-layer identification solution. SIMPLE is a dual intrusion detection and identification system, which is computationally lightweight and can make detection and identification decisions before a frame ends.
- SIMPLE performs secure updates of the fingerprints to compensate for environmental changes, such as temperature and supply voltage. SIMPLE can also prevent intrusion attacks by invalidating spurious frames before they take effect.
- We evaluate SIMPLE using both an in-lab testbed with ten automotive-grade ECUs, and in-vehicle experiments. For the in-lab experiments we show that SIMPLE can distinguish perfectly between the frames transmitted from compromised ECU and benign frames, even when the fingerprint changes due to environmental effects. For the in-vehicle experiments we show that SIMPLE achieves a low equal error rate (EER) around 0.8985%.

1.1 Paper structure

Section 2 provides a brief background on the preliminary of the CAN protocol, the applications of PLI, and a survey of existing works. Sec. 3 presents our attack model. Sec. 4 demonstrates the Hill-climbing-style attack against two state-of-the-art fingerprinting schemes. Sec. 5 describes SIMPLE, which is then analyzed and evaluated in Sec. 6 and Sec. 7, respectively. Finally, we draw our conclusion in Sec. 8.

2 BACKGROUND AND RELATED WORK

2.1 CAN protocol

CAN is a protocol created in 1986 by Robert Bosch GmbH [4], for communication among ECUs within in-vehicle networks. It is a two-wire, half-duplex bus generating CAN High (CANH) and CAN Low (CANL) signals as output which are shown in Fig. 1b. The details of the CAN protocol is given in [15]. All the ECUs are connected to the same bus, thus they can receive all the frames broadcasted on the bus.

There are a few features of the CAN protocol that are related to this work. First, each ECU can be assigned one or multiple message *identifiers* (IDs) that it can send out, which usually represents the data type. Two or more ECUs that want to transmit at the same time have to participate in so-called *priority-based arbitration* in order to occupy the CAN bus. The lower the numerical value of the ID is, the higher the priority the message has. Second, any ECU that observes an error in a frame will transmit an Error Frame that will cause other ECUs to discard the previous/current frame (an ACK slot at the end of each CAN frame allows the ECU that transmitted the frame to determine if a single ECU successfully received the frame) [4]. In addition, message reception is not based on destination address but on message ID (for example, an engine ECU is programmed to receive only certain subset of "interesting" message IDs related to engine status/control).

From the security aspect, weak integrity check is performed by calculating the cyclic redundancy checks (CRC) in each frame. Considering that an ECU can forge the ID, the protocol lacks strong authenticity, which can be fulfilled by using physical layer identification.

2.2 Physical Layer Identification (PLI)

To enhance authenticity, most of the solutions offered so far impose modifications to the protocol. Cryptographic solutions such as message authentication codes (MACs) [43] are not ideal due to limited length of CAN frames and the computational constraints of ECUs. The PLI technique is potentially free of these drawbacks, while it avoids changing the CAN protocol.

PLI takes into account the hardware and manufacturing inconsistencies that cause minute and unique variations in the signalling behaviour of devices, and translates them into features that can provide reliable identification [50]. A typical PLI system includes three major steps, namely, data acquisition, feature extraction and decision [18]. The data acquisition converts an analogue voltage to a digital signal using an analogue to digital converter (ADC) for further processing; the feature extraction module performs the task of acquiring *fingerprints* for the devices by leveraging the statistical or physical layer characteristics of the signals collected in the previous step and using them to find features; and lastly, the decision module compares the extracted fingerprints from the training data with the ones from the test data using a specific distance metric and defines a threshold for final identification decision making based on how close the features are. Fig. 1a illustrates an overview of a PLI system. Interested readers are encouraged to read about PLI and its applications in [14, 17].

Our proposed scheme SIMPLE also leverages PLI. Among the diverse techniques that are available, we use the Fisher Discriminant Analysis (FDA) transform, which is a dimension reduction technique, followed by a distance calculator named Mahalanobis distance [2], so as to extract and process relevant features from the analog signals and use them as the discriminators in our system.

2.3 Related work

Existing PLI systems can be categorized into *timing-based* methods and *voltage-based* methods.

Timing-based. CAN messages are typically sent in a periodic manner. The analysis on the interval/frequency of the messages might be able to show the first evidence of intrusion. Müter et al. [37] use entropy to analyze the randomness of intervals. Moore et al. [35] proposed a data-driven inter-signal arrival times model to detect injection attacks. Abnormality detection sensors [38] evaluate the payload length, frequency or correlation, etc. Taylor et al. [44] proposed to detect anomalies of the sequence data transmitted from an ECU by applying neural networks. A similar work [31] automatically classifies the fields in CAN messages and measures valid ranges based on previous data, which is computationally expensive. Ying et al. proposed [53] TACAN based on shared crytographic keys and inter-arrival times, which introduced computational overhead. Cho and Shin [8] proposed clock skew based fingerprints for their PLI-based IDS, CIDS. However, the timing-based approaches can be easily defeated by an adversary imitating the target ECU's timing behavior [10, 40]. Avatefipour et al. [1] train a neural network to capture the features in both frequency and time domain, which is impractical for a CAN bus because of an ECU's weak computational ability.

Voltage-based. The voltage output nature differs among the transmitters of ECUs, enabling voltage-based approaches to detect the intrusion and identify its source. Murvay et al. [36] took the first step towards voltage identification, which is later extended by several contributions such as Viden [9], VoltageIDS [10], and Scission [29]. These methods construct and update voltage profiles for the ECUs and use them for identification of malicious messages. However, the effect of the variations of voltage power source and ambient temperature on these features is non-negligible and has not been taken into consideration. That is, the life-span of the valid features would be limited to a short interval as the source voltage would be easily affected by any change in ambient temperature. This is the reason the features need to be updated every time the vehicle is restarted, which motivates applying adaptive profile updates [9]. In fact, because the features smoothly evolve through time, Viden has to keep track of these changes by updating their profiles through time, which makes Viden vulnerable due to the lack of secure training data that updates with time.

Table 1: Comparison among voltage-based approaches in sampling rate (S.R.), false negative (F.N.), true possitive (T.P.), time complexity (T.C.), signal type (S.T.), environmental compensation (E.C.), secure feature update (S.F.U.), unknow ECU (U.E.).

	Viden [9]	VoltageIDS [10]	Scission[1] [29]	SIMPLE in lab	SIMPLE in vehicle
S.R.	50 KS/s	2.5 GS/s	20 MS/s	500 KS/s	1 MS/s
F.N.	0.2%	3.52%	0.15%	0%	0.899%
T.P.	99.8%	96.48%	99.85%	100%	99.1%
S.T.	CANH&L	Diff.	Diff.	CANH&L	Diff.
T.C.	$\Omega(n^2)$	$\Omega(n \log n)$	$\Omega(n \log n)$	$\Theta(n)$	$\Theta(n)$
E.C.	No	No	No	Yes	Yes
S.F.U.	No	No	Yes	Yes	Yes
U.E.	No	No	No	Yes	Yes

Most existing works (e.g., [9], [8], etc.) in both timing and voltage-based categories rely on multiple messages to make detection and identification decisions. We found that, such reliance on multiple messages makes them vulnerable to a variance of Hill-climbing-style attack [42], in which the adversary is able to inject carefully chosen fraction of malicious messages without being either detected or identified. Furthermore, the adversary is able to iteratively increase the injection rate so that the detection and identification decision threshold can be shifted. We will demonstrate such vulnerability later in Sec. 4. Even though VoltageIDS and Scission does not rely on multiple messages, it is not motivated by the vulnerability of multi-frame-based IDS techniques. Besides, they exploit features in both time and frequency domain, which leads to high complexity.

Our proposed voltage based PLI scheme, SIMPLE, chooses features only from the time domain and avoids the complexity of frequency domain transformations. It performs a single-frame based detection that is not vulnerable to changes in ambient conditions. Hence, we avoid the unnecessary feature retraining every time the vehicle gets restarted. Both are accomplished by using a higher resolution and higher sampling rate sampler; i.e., a cost-vs-security tradeoff is made to detect an attack using a single frame. Specifically, while Viden [9] only needs to acquire a few samples per frame over multiple frames, and thus needs only a low-rate sampler, SIMPLE needs multiple samples for a single frame and thus requires a high-rate sampler. Nevertheless, SIMPLE is still practical since a sufficiently high resolution and high sample-rate sampler can be acquired for less than \$10 per CAN bus [45]. Additionally, unlike Viden, we do not require a separate intrusion detection system (IDS) next to our identification system. Unlike CIDS [8], we can handle both periodic and aperiodic messages. Finally, our approach incurs lower overhead and cost than VoltageIDS [10], as we require fewer samples and a lower sampling rate.

[1]Scission used a slightly different setup for the prototype in which each ECU has a different stub length (2.45 - 13 meters) necessitating stub terminations. This imperfection in CAN bus topology can effect the voltage profile of the ECUs and bias the results knowing that the strongest features in Scission are extracted from the overshoot at the rising edge. In the setup used by SIMPLE however, the length of stubs are all identical and short enough (less than 5 cm) to be negligible.

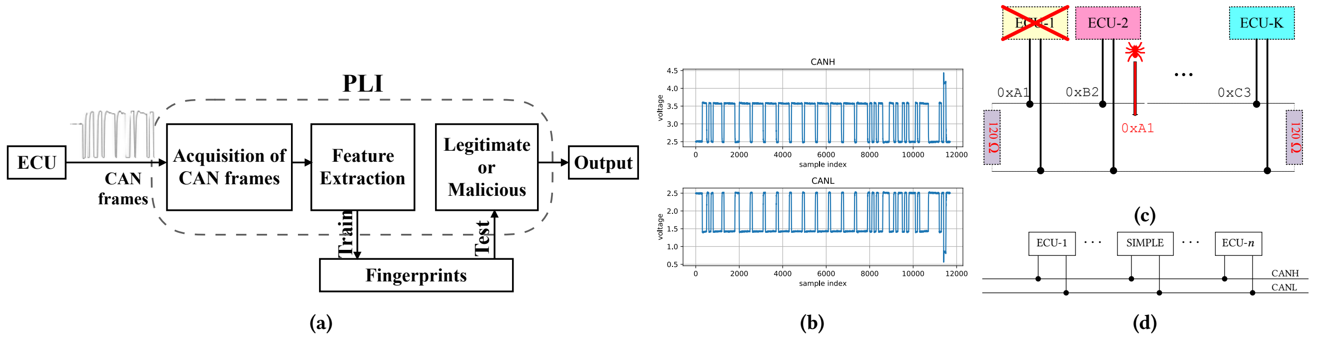


Figure 1: (a) Components of a physical layer identification (PLI) system. (b) A CAN frame captured from a Nissan Sentra. (c) Masquerade attack model on a CAN bus. ECU-2 is compromised by an adversary and reprogrammed to forge the message ID of ECU-1 and send spurious frames instead it. (d) SIMPLE runs on a device which is placed on the CAN bus just as other normal ECUs.

All the voltage-based identifiers given in Table 1 illustrate a non-zero value for the error rates. As small as these values are, not even a frame can go wrong for safety critical applications such as air bags. Further efforts like augmented solutions need to be taken to provide the required security for such applications. SIMPLE has the lowest time complexity (Table 1), while Viden takes $\Omega(n^2)$ time for each update due to the use of the Recursive Least Squares algorithm. VoltageIDS and Scission need $\Omega(n \log n)$ time because they perform Fourier Transforms in every update to obtain frequency domain features.

3 ATTACK MODEL

We consider an adversary capable of compromising one or more[2] ECUs either remotely via wireless interfaces (e.g., the telematic port [19, 33]) or physically (e.g., via OBD-II [30]). Once compromised, the ECU is under full control of the adversary and becomes an *attacker ECU*. With the full control, the adversary can either suspend the ECU or even re-program it to inject arbitrary messages, e.g., use arbitrary message IDs to impersonate another ECU and/or transmit messages containing forged/spurious data [3]. These messages are called *attack messages*. For example, the attacker could compromise an ECU belonging to the entertainment system, and send out "accelerate" or "shut down engine" commands under a different message ID (which is normally sent by the engine control ECU), so as to spoof the engine to carry out wrong actions. We assume that the adversary is aware of the IDS that is installed on the CAN bus and the adversary is able to implement the same algorithms as the IDS. The adversary can also obtain necessary information that can be measured on the CAN bus (e.g., timing and voltage information of other ECUs) using the compromised ECU. Note that this is also assumed by CIDS [8] and Viden [9]. We additionally assume that the ECUs are equipped with temperature sensors with secure measurements.

Furthermore, there are two types of objectives that the adversary may wish to achieve for an impersonation attack. First, it may choose to pursue *dominant impersonation* where at the end a majority of the messages with the targeted message ID are attack messages; or it can choose to pursue *complete impersonation* which is even stronger as the adversary is now able to replace *all* legitimate messages with attack messages. The methods to achieve these attacks will be detailed in the next section. We assume that the attacker can either directly inject its own frame onto the CAN (when no other ECUs are transmitting) or preventing another ECU from transmitting. The latter can be done by synchronizing to the regular messages on the CAN bus and play attack messages right before the one that a legitimate ECU is about to send [34], or by contending with the latter during the arbitration phase [4, 7].

Note that, the attacker ECU could also inject false data under its own ID, however this is not as effective as impersonation, because CAN messages are addressed by its message IDs which typically represent the data type (e.g., an engine ECU would ignore a message with one of the IDs belonging to an entertainment ECU). Hence, false data injection attack detection is out of scope of this work. Detection of denial-of-service attacks, such as the bus-off attack [7] will also be studied in the future.

4 VULNERABILITY OF MULTI-FRAME BASED FINGERPRINTING SYSTEMS

In a multi-frame based fingerprinting system, a batch of multiple frames has to be collected in order to perform one update of the fingerprinting record/threshold. Such fingerprinting schemes are vulnerable to the so-called *Hill-climbing-style* attack, where the adversary is able to control the quantity of attack frames among the batch of frames collected, so that the attacker ECU can both hide its identity and shift the fingerprinting decision threshold gradually. Specifically, from the batch of n collected frames, only m of those are attack frames. The injection ratio $r = m/n$ can be carefully chosen for each step, so that the IDS cannot identify the attacker. More importantly, the fingerprinting decision threshold will be shifted via raising r iteratively, so that eventually, the attacker will be able to impersonate the legitimate ECU.

In this section, we will use the Viden fingerprinting system [9] as our case study to demonstrate how multi-frame based schemes

[2]Our Hill-Climbing style attack requires compromising only one ECU to work. While SIMPLE can defend against multiple compromised ECUs.
[3]Similar attack goals have been considered in previous works [8, 9]

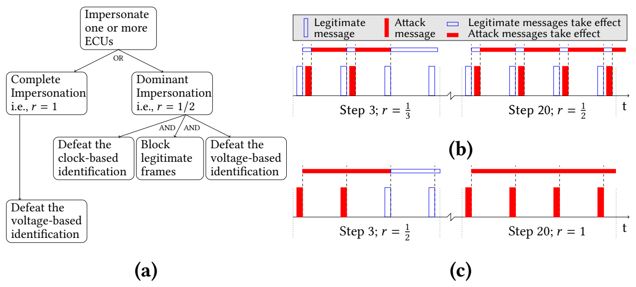

Impersonate one or more ECUs
OR
Complete Impersonation i.e., $r = 1$
Dominant Impersonation i.e., $r = 1/2$
AND
AND
Defeat the clock-based identification
Block legitimate frames
Defeat the voltage-based identification
Defeat the voltage-based identification
Legitimate message
Attack message
Legitimate messages take effect
Attack messages take effect
Step 3; $r = \frac{1}{3}$
Step 20; $r = \frac{1}{2}$
Step 3; $r = \frac{1}{2}$
Step 20; $r = 1$
t
(a)
(b)
(c)

Figure 2: (a) The attack tree against Viden's fingerprinting system. Hill-climbing-style attacks towards (b) Dominant impersonation and (c) Complete impersonation.

are vulnerable to the Hill-climbing-style attack. In the system, two fingerprinting schemes are employed independently in parallel: (i) Clock-based IDS (CIDS) [8] that tries to estimate the clock skews of different ECUs; and (ii) Voltage-based Identification (Viden) [9] that tries to abstract the voltage output characteristics. Both schemes collect a batch of frames, calculate the momentary features that ideally are constant over time, and then accumulate the latest feature with all historical ones. Because of the constancy of features, the cumulated quantity appears linear over time; hence the slope of it can be regarded as the profile of an ECU (i.e., the clock-skew S and the voltage profile Υ). Details of the two works are re-stated in Appendix A. Here, we mainly focus on the fingerprinting system and we leave the discussion about the intrusion detection system to the end of this section.

As discussed in Sec. 3, the adversary can pursue the dominant impersonation or the complete impersonation. It will be demonstrated in the rest of this section that the former is easier to succeed, whereas the latter is stronger. This is summarized in an attack tree (Fig. 2a).

4.1 Dominant Impersonation

An example of the dominant impersonation is given in Fig. 2b, where the attack frames are injected right after the legitimate frames so that the attack frames will be in charge of the vehicle for most of the time [12]. With the carefully chosen injection ratio r for each step, the adversary will be able to evade identification [4]. Note that CIDS does not work because the periodicity of frames is disrupted, thus the adversary only has to defeat the voltage-based scheme [9] at this point.

Let us suppose the adversary compromises ECU $\mathbb{A}$ and runs the voltage-based fingerprinting scheme on it for k_0 steps. The adversary decides to start the intrusion at step $k_0 + 1$ to impersonate ECU $\mathbb{B}$. In each of these future steps, the voltage-based fingerprinting scheme will observe a batch of n *mixed frames*, consisting of m attack frames sent from ECU $\mathbb{A}$ and $n - m$ legitimate frames sent from ECU $\mathbb{B}$, both with $\mathbb{B}$'s target ID. The adversary wants to enlarge m as much as possible, while it does not want the fingerprinting scheme to identify $\mathbb{A}$. To do so, the injection ratio r has to be chosen carefully.

Given the learnt voltage profiles $\Upsilon_{\mathbb{A}}[k-1]$ and $\Upsilon_{\mathbb{B}}[k-1]$ from the previous $k-1$ steps, the adversary derives the desired voltage profile, $\Upsilon_{\mathbb{B}}[k]$, of the mixed frames for the next step (i.e., k-th step) by solving the following optimization problem:

$$\begin{aligned} &\text{minimize} && \left|\Upsilon_{\mathbb{B}}[k] - \frac{1}{2}\Big(\Upsilon_{\mathbb{A}}[k-1] + \Upsilon_{\mathbb{B}}[k-1]\Big)\right| \\ &\text{subject to} && |\Upsilon_{\mathbb{B}}[k] - \Upsilon_{\mathbb{B}}[k-1]| < |\Upsilon_{\mathbb{B}}[k] - \Upsilon_{\mathbb{A}}[k-1]|\,, \end{aligned} \tag{1}$$

where the objective is to make $\Upsilon_{\mathbb{B}}[k]$ as close to the threshold (the average of two profiles) as possible. With the constraint, the adversary makes sure that $\Upsilon_{\mathbb{B}}[k]$ is still closer to $\Upsilon_{\mathbb{B}}[k-1]$ than $\Upsilon_{\mathbb{A}}[k-1]$, so that the fingerprinting scheme regards $\mathbb{B}$ as the source of all the n frames. Eq. 1 yields

$$\begin{aligned} \Upsilon_{\mathbb{B}}[k] =& \frac{1}{2}\Big(\Upsilon_{\mathbb{A}}[k-1] + \Upsilon_{\mathbb{B}}[k-1]\Big) + \\ & \epsilon \cdot \operatorname{sign}(\Upsilon_{\mathbb{B}}[k-1] - \Upsilon_{\mathbb{A}}[k-1]), \end{aligned} \tag{2}$$

where ϵ is a small enough factor for satisfying the constraint. See Fig. 3a for its illustration, from which we see that the profile of the mixed frames is sitting below the decision threshold by ϵ, fooling the fingerprinting scheme into believing that the intrusion source is $\mathbb{B}$. Furthermore, the threshold is shifted a bit closer to $\mathbb{A}$'s profile after each step, so is the profile of $\mathbb{B}$. This is done by gradually increasing the injection ratio r.

The relationship between the desired profile $\Upsilon_{\mathbb{B}}[k]$ and the injection ratio $r[k]$ of k-th step is

$$\Upsilon_{\mathbb{B}}[k] = r[k]\Upsilon_{\mathbb{A}}[k-1] + (1 - r[k])\,\Upsilon_{\mathbb{B}}[k_0], \tag{3}$$

where $\Upsilon_{\mathbb{B}}[k_0]$ is the ground truth profile of $\mathbb{B}$ because the intrusion started at step $k_0 + 1$; $\Upsilon_{\mathbb{B}}[k]$ is the profile of the n mixed frames at step k, and $k > k_0$. The detailed derivation of Eq. 3 can be found in Appendix A.2. Finally we have the maximum injection ratio

$$r^*[k] = \frac{\Upsilon_{\mathbb{B}}[k] - \Upsilon_{\mathbb{B}}[k_0]}{\Upsilon_{\mathbb{A}}[k-1] - \Upsilon_{\mathbb{B}}[k_0]}. \tag{4}$$

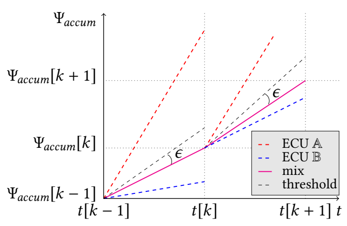
With the maximum injection ratio, the adversary will be able to inject as many attack frames as possible at step k while avoiding identification. Since the legitimate ECU $\mathbb{B}$ is identified as the source, Viden accepts this batch of frames and updates $\Upsilon_{\mathbb{B}}$ to be a bit closer to $\Upsilon_{\mathbb{A}}$. With the updated profile, the adversary will be able to get an even higher r, thus injecting more and more attack frames at each future step. Hence, the injection ratio r will be able to reach $1/2$. Note that the attack will not trigger the random forest classifier of Viden because the attack will not bring two profiles close to each other; instead, it is only the profile of the legitimate ECU that gets shifted (not the profile of the attacker ECU).

The adversary can of course continue shifting the profile, i.e., r is approaching one. In the dominant impersonation, it is unnecessary to do so because it will not increase the amount of time that the attack frames take effect on the vehicle (Fig. 2b). However, it becomes necessary in the complete impersonation.

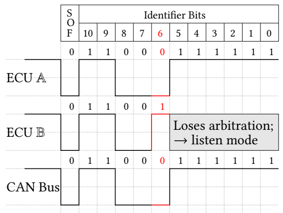
4.2 Complete Impersonation

Although the dominant impersonation is strong enough to let the vehicle follow what the attack frames instruct, there are still some

[4]For dominant impersonation, our goal is not to evade detection since it may be trivially detected based on the periodicity of CAN frames. But complete impersonation can evade both detection and identification.



(a) Hill-climbing-style attack



(b) Contention

Figure 3: (a) Two contiguous steps of a Hill-climbing-style attack against the voltage-based fingerprinting scheme. Υ is the slope of Ψ_{accum} with respect to t. (b) Contention with $\mathbb{B}$ during the arbitration phase.

legitimate frames being transmitted on the CAN bus and instructing the vehicle to behave as normal. This conflict may cause suspicion.

As a result, the adversary wants to block $\mathbb{B}$ from sending any frame, i.e., $r = 1$, by contending with the legitimate ECU $\mathbb{B}$ during the arbitration phase, using a forged and smaller identifier (than $\mathbb{B}$'s), making $\mathbb{B}$ lose the contention.

4.2.1 Blocking the legitimate frames. In order to block the legitimate frame, the adversary has to know beforehand when the legitimate frame will be sent. To do so, the attacker ECU $\mathbb{A}$ learns the time skew $S_{\mathbb{AB}}$ of the legitimate ECU $\mathbb{B}$, just as a CIDS's ECU (say $\mathbb{C}$) that learns of $S_{\mathbb{CB}}$. After the learning phase where i legitimate frames have been sent, $\mathbb{A}$ calculates the moment when the next frame will be sent as

$$t_{\mathbb{A}}^{(i+1)} = t_{\mathbb{A}}^{(i)} + T \cdot (1 + S_{\mathbb{AB}}), \tag{5}$$

where T is the preset interval between two legitimate frames. Similar results were derived by two previous works [10, 40]. The difference is that here the clock-skew imitation is used to block the legitimate frames, rather than directly inject the attack frames. An example of Eq. 5 is given in Fig. 4, in which $\mathbb{A}$ learns the time skew of $\mathbb{B}$ as $S_{\mathbb{AB}} = (10.03 - 10)/(10) = 0.003$. Meanwhile, CIDS learns it as $S_{\mathbb{CB}} = (10.01 - 10)/10 = 0.001$. According to Eq. 5, at the $\mathbb{A}$'s time $10.03 + 10 + 10 \cdot 0.003 = 20.06$ (or $\mathbb{B}$'s time $10 + 10 = 20$), $\mathbb{A}$ blocks the legitimate frame.

The blocking is done by taking advantage of the arbitration phase. In order to block a legitimate frame at $t_{\mathbb{A}}^{(i+1)}$, the attacker ECU $\mathbb{A}$ simply sends a frame with a smaller identifier than the legitimate ECU's. An illustration of the contention is shown in Fig. 3b.

The contention will not trigger the "bit-error" because the CAN bus standard [4] does not count the wrong bits in the identifier field as bit-errors. Also, the feasibility of flipping a bit has been demonstrated by the *bus-off* attack where the adversary is able to disable a targeted ECU by letting the attacker ECU flip a bit in the data field of the frames sent by the targeted ECU [7].

4.2.2 Defeating the clock-based scheme. In addition to defeating the voltage-based fingerprinting scheme, now the adversary has to defeat the clock-based scheme [8] *at the same time* in the complete impersonation. This is because the periodicity of the frames still exists when the legitimate frames are blocked, and such periodicity can be used by CIDS to do identification. Since $t_{\mathbb{A}}^{(i+1)}$ has been

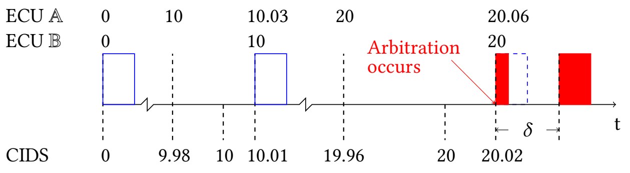


Figure 4: Complete impersonation zoomed-in from Fig. 2c. The three rows of numbers represent the time that different ECUs report.

occupied for blocking, the adversary has to inject the attack frame δ seconds later. The injection offset δ is illustrated in Fig. 4 and it can be determined as follows.

Similar to Sec. 4.1, the attacker ECU $\mathbb{A}$ has been learning the clock offset O as well as the clock skew S of the legitimate ECU $\mathbb{B}$ for $k-1$ steps. Given the maximum injection ratio $r^*[k]$ derived from Eq. 4, the maximum injection offset $\delta[k]$ can be obtained by solving the following optimization problem:

$$\begin{aligned} &\text{minimize} && \left| S_{\mathbb{B}}[k] - \frac{1}{2}\Big(S_{\mathbb{A}}[k-1] + S_{\mathbb{B}}[k-1]\Big)\right| \\ &\text{subject to} && |S_{\mathbb{B}}[k] - S_{\mathbb{B}}[k-1]| < |S_{\mathbb{B}}[k] - S_{\mathbb{A}}[k-1]| \,. \end{aligned} \tag{6}$$

Similar to Eq. 1, the adversary tries to change $S_{\mathbb{B}}[k]$ as much as possible but not too far. In practice, because of the relativity of clock skew, $S_{\mathbb{A}}[k-1] = 0$ and $S_{\mathbb{B}}[k-1]$ is the clock skew from the perspective of $\mathbb{A}$. Eq. 6 yields the optimal desired clock skew $S_{\mathbb{B}}[k]$ that is similar to Eq. 2, which we will omit due to the page limit. Then $\delta[k]$ can be derived from its relationship with $S_{\mathbb{B}}[k]$, which is

$$S_{\mathbb{B}}[k] = \frac{O[k-1] + \delta[k]r^*[k]n/(n-1)}{t[k] - t[k-1]}, \tag{7}$$

where $O[k-1]$ is the clock offset of the previous step. Readers may go to Appendix A.1 to see why Eq. 7 holds.

It should be noticed that we choose such $\delta[k]$ in order to evade the identification of CIDS, not its detection. Of course, the adversary can always evade detection by adding the detection thresholds (Eq. 14) as constraints to Eq. 6. In Sec 7.2.2, however, we evaluate only the identification evasion due to experimental data limitations.

4.3 Attack Evaluation

We use real-world data collected from vehicles to evaluate the attacks against the voltage-based and the clock-based identification schemes. Results show that the attacker ECU was able to impersonate the legitimate ECU and was identified by neither of the schemes. Meanwhile, the profiles of the legitimate ECU was shifted to the attacker ECU profile gradually. Moreover, the number of steps required to achieve dominant impersonation increases with the difference between two profiles. Refer to Sec. 7.2.2 for the details of the attack evaluation results.

4.4 Discussion

Intrusion Detection System. In Viden's fingerprinting system [9] there is an underlying IDS that detects the intrusion and submits the suspicious frames to the fingerprinting schemes. However, Cho et al. did not specify what (type of) IDS is used, thus we do not specify the IDS in this work, either. Instead, we assume that as long as the fingerprinting result is the legitimate ECU, the vehicle will behave normally and no one gets suspicious. Note that if the IDS is based on multiple frames, it is also vulnerable to Hill-climbing-style attacks.

Arbitrary attack targets. In the previous example, we assumed that ECU $\mathbb{A}$ and $\mathbb{B}$'s profiles are adjacent. In general, there may be another ECU $\mathbb{C}$, whose profile is in between $\mathbb{A}$'s and $\mathbb{B}$'s, in which case Viden would regard $\mathbb{C}$ as the source of intrusion and an alarm may be triggered. To avoid so, the adversary can only impersonate the legitimate ECU that has the nearest voltage profile to the attacker ECU's profile. Such limitation can be addressed by enabling the adversary to imitate other ECU's voltage profile, e.g., changing the temperature. This can be achieved by extra code executions or increasing the CPU's clock speed.

Retransmission. Because the CAN bus protocol [4] did not specify what an ECU would do if it loses a contention during the arbitration phase, we assumed in Sec. 4.2 that it would simply give up this frame. We argue that even if retransmission occurs once the CAN bus becomes idle in some specifications/implementations, the adversary can always choose to pursue the dominant impersonation, which does not involve any contention.

Voltage/timing profile knowledge. When the compromised ECU does not have an analog to digital converter (ADC), or a precise clock embedded, the adversary cannot calculate the optimal/maximum injection rate r. It can, however, be conservative. That is, it can find out the common minimum distance between profiles in car ECUs and use that as an empirical value to get a minimum injection rate. The attack will then take more time to succeed (or probabilistically). From our real-world experiment, however, r can still be high even with a small distance. See Fig. 10 for an example, where the distance is just 0.05 but r can still be as high as 8/28 at the first step.

5 SECURE PLI FOR CAN

In Sec. 4, we showed that existing fingerprinting schemes are vulnerable to the Hill-climbing-style attack due to their dependence on multiple frames to make one detection or identification decision. In this section, we will describe our fingerprinting scheme, SIMPLE. Since it only requires a single CAN frame to perform the identification, it is immune to the Hill-climbing-style attack.

The overview of SIMPLE can be found in Fig. 5a, where the samples collected from the entire frame (identifier as well as the data frame) are used for generating features, dimension reduction and eventually are fed into a Mahalanobis distance calculator. On the CAN bus, the location of SIMPLE is shown in Fig. 1d where it listens to all the traffic. Beyond securing against Hill-climbing-style attacks, another benefit of SIMPLE is the ability to *prevent* malicious frames from having their intended effect by intentionally introducing errors that will cause ECUs to ignore the frames.

5.1 Feature Extraction

As an essential step of fingerprinting, feature extraction should not be time consuming and should reduce or eliminate the domain transformations as much as possible. We first, select proper features, and then apply a dimension reduction transformation which eventually saves computational power. For the first step, we intend to use the sample points after high-to-low or low-to-high transitions in a CAN frame. After detecting such a transition in the line voltage, we use Alg. 3 to separate the high/low sample points, named hereafter as bins. Next, we apply an intra-frame average on the bins to increase the signal to noise ratio (SNR) given at Alg. 2.

In the state-of-the-art IDSs, usually inter-frame average of CAN frames is used to achieve sufficiently high SNR features, which requires multiple frames for finalizing the detection process; otherwise, the identification performance will be degraded. Fig. 5b illustrates the difference of inter-frame average of features with intra-frame average of features. To be more specific about the intra-frame averaging (our method for achieving a high SNR), sample points from a CAN frame are shown in Fig. 5c, where the S_1^1 and S_1^2 and any other *first sample point after a high-to-low transition* throughout the frame will be averaged together and create the first feature, denoted as F_1 here. The same process will be repeated for generating the second feature, F_2, by averaging all the second sample points after all of the high-to-low transitions within a single CAN frame. We extract eight features from the low bins and eight from the high bins; and the reason behind not proceeding further is that increasing the number of features does not enhance the accuracy of our work based on the experiments. Choosing fewer than eight features however increases false positive/negative rates.

We collect signals from high/low bins of CANH/CANL at the same time which aggregates to 32 features overall. The feature selection process is based on simple operations all in the time domain, which eliminates the need for intricate operations. This enables us to efficiently identify the source of a frame on a single-frame basis analysis.

Next, we perform Fisher-Discriminant Analysis (FDA) [2] in order to derive a transformation W to reduce the dimension of the data. In fact, FDA captures the most discriminant information in the features. It has been used with identification/classification purposes for instance to discriminate human biometrics [54]. An optimal transformation matrix, W, is the main output of the FDA algorithm which focuses on maximizing the separability of known categories in a classification problem. W is used for projecting the features, F, to a new set of features, $\mathscr{F}$,

$$\mathscr{F} = W \times F. \tag{8}$$

5.2 Mahalanobis distance calculation

Mahalanobis distance is a measure of calculating the distance of a point from a distribution. If we associate a distribution to a set of data during the training phase with mean and covariance of μ and Σ, we can next calculate the distance of a new set of observations, $\mathscr{F}$ during the test phase, from that distribution by the following definition of Mahalanobis distance [2]:

$$d = \sqrt{(\mathscr{F} - \mu)^T \, \Sigma^{-1} \, (\mathscr{F} - \mu)}. \tag{9}$$

In this context, Mahalanobis distance can be used as a measure of similarity between the features and used as a score to match and compare the features generated from an uncertain origin to the templates belonging to the legitimate devices.

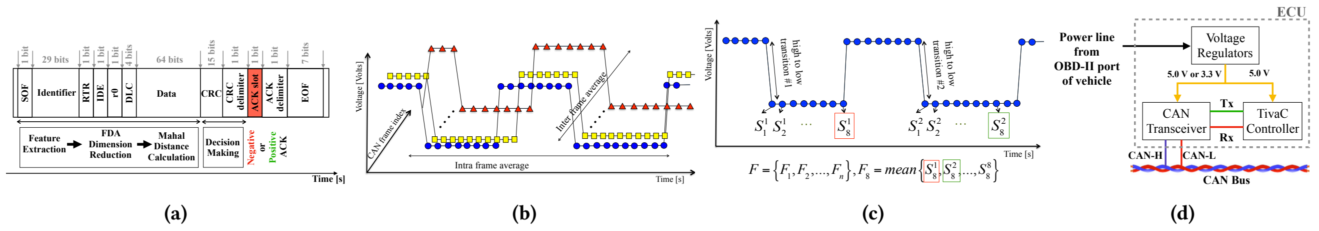


Figure 5: (a) Defense mechanism in a chronological perspective. (b) Inter-frame average of features vs. intra-frame average of features. (c) Generating features from a CAN frame. (d) The general diagram for building automotive grade ECUs using a CAN transceiver and a TM4C123GXL (TI) controller (the necessary side circuitry for proper functionality of the ICs is omitted).

5.3 Training and Testing

There are two main steps for designing SIMPLE, first, training and testing phase, which results in thresholds for identification of each device, and next, real-time identification using these thresholds.

Training Here we generate an offline database of legitimate features for each device. These templates are calculated as explained in Sec. 5.1 using 200 frames from each device. Data alignment (traditionally used to eliminate the effects of the lags in data) is not necessary because the selected features in Sec. 5.1 inherently take care of it by selecting the sample points after transitions which leads to their alignment.

Testing For every single record, the Mahalanobis distance (Sec. 5.2) of its feature is calculated from template features of the device that this frame normally should be emitted from. If the distance is not close enough, it will be identified as a malicious frame transmitted by a spoofed ECU. A binary search algorithm [51] is used to find the threshold for these distances by looking for EERs. EER is a common measure of performance of biometrics systems which indicates a condition of the system where the number of false positives is equal to the number of false negatives [3]. The EER point, corresponds to the thresholds that we need as a distance metric. An ideal system would have an EER of 0 %.

Real-time Identification During the identification, every single frame needs to be tested to establish the frames' provenance. To do so, the same feature extraction process is applied to the data frames and the output is compared to the feature templates that are already generated and logged in the training phase. If the features of the frames under test, match as close as needed to the template of the device that they are claiming to be transmitted from, they can be marked as legitimate and mounted on the bus. This is measured using the threshold values that are generated in the testing phase. After the identification is over, the valid frames are used to securely update the template of the device that they belong to, over time.

Here is a summary of SIMPLE, with its details given at Alg. 4.

(1) Feature extraction and FDA transformation, Alg. 3 and 2.
(2) Training feature templates for each ECU, Alg. 4.
(3) Associating a threshold to each ECU by comparing the feature templates using Mahalanobis distance metric, Alg. 4.
(4) Identifying the origin of a single frame based on the proximity of its distance from the template of the device, to the device's predefined threshold.
(5) Updating the feature templates of the devices after confirming validity of the origin of a message.

5.4 Time Complexity Analysis

Referring to the algorithms in Sec. 5.3, the time complexity of SIMPLE is limited to: generating the features which depends on the number of sample points within each frame that grows with time n as $\Theta(n)$, followed by calculating the Mahalanobis distance of this feature from the template of the ECU that it claims to be emitted from, $\Theta(1)$, and finally a comparison with the threshold of that ECU (generated offline in the training phase), $\Theta(1)$. Hence, the overall complexity of the real-time IDS SIMPLE is $\Theta(n)$.

5.5 Intrusion Prevention

Since SIMPLE is able to determine that a frame is malicious before its transmission ends, it can effectively prevent the attack frame from being received (acted upon) by other ECUs. There are two possible ways to achieve this. First, upon detecting a malicious frame SIMPLE can transmit an Error Frame (signified by an error flag of six dominant "0" bits and then an error delimiter) which will cause non-compromised ECUs to ignore the frame. Second, SIMPLE could introduce errors in the frame that would cause the intended recipient devices to ignore the frame. For example, it can force the bus into the dominant state for consecutive and/or random bit periods (equivalent to sending `0`s) when the body of the frame (including CRC) is being transmitted; this can fail the CRC check (all ECUs are obligated to perform), which results in the transmission of an Error Frame that will cause all ECUs to ignore the frame. For CAN frames with extended identifiers, in the best case scenario with only one byte of data, it will take 1.575 µs to prevent the attack, while in the worst case of eight bytes, it will take 2.975 µs to prevent the attack in a device with 80 MHz of processing power.

6 ORIGIN OF FINGERPRINT VARIATIONS

In the above section, we introduced the basics of our PLI system when the fingerprints are stable over time; existing work in the area indicates that fingerprints are stochastic. In this section, we undertake experiments to identify the factors that cause fingerprint variations and propose a method to compensate for their effects.

6.1 Voltage and Temperature Effects

Changes in temperature and supply voltage are the main potential factors that could cause drift in the features that we extract for fingerprinting in our IDS [9]. In practice, a vehicle experiences different environmental conditions; e.g., it is hot when running and colder when turned on after a cold night. Here we try to evaluate the impact of these factors and compensate for them.

6.1.1 Impact of Supply Voltage. We look for a relationship between features and voltage supply so that we could scale the features taken at a voltage different than the voltage level used in the training data, and compensate for variations of Vcc in a realistic scenario. That is, we want to discover the relationship between the features generated from the new voltage levels, Vcc = Vsr, and the features generated on Vcc = Vtg, which are to be called the *source domain* and *target domain*, respectively. We use a linear regression model for all 32 features coming from high/low sample points of CANH/CANL during training phase and estimate the scaling parameters so as to transform the features from each *source domain* up to the *target domain*. These linear scaling parameters allow us to remove the effect of change in the voltage level by transferring all the features to the target domain. Later in the testing phase, after the features get generated for each frame, they get scaled up to the *target domain* using the scaling parameters that were estimated in training phase. Last, the conventional FDA-followed-by-Mahalanobis-distance-calculation is performed. Eq. 10 shows this linear regression.

$$F_{\mathrm{tg}} = a_0^f * F_{\mathrm{sr}} + a_1^f, \tag{10}$$

where F_{tg} is the vector of features on the *target domain*, F_{sr} is the vector of features on the *source domain*, and a_0^f and a_1^f are the scaling parameters for feature set f which can be one of the four different combinations of features (high/low bins of CANH/CANL).

Now we need to leverage this knowledge to upgrade our IDS in a way that it accounts for variations in the supply voltage level. To do so, we extracted features on supply voltage levels that already had training data available and scaled them to a specific target voltage domain (to avoid the drift in the fingerprints caused by variations of the voltage level). Next, we need to be capable of repeating the same process on supply voltage levels for which the training data is **not** available. This is leads to a generalized solution in the following.

Compensating for supply voltage changes. In the previous discussions, we defined different voltage domains for generating features and exploited regression as a simple tool for mapping the features from one domain to another. Now we apply the idea to our scenario and estimate matrices A and B, as $n \times 1$ matrices, where the rows correspond to the voltage level varying in the interval of $[3.00, 3.25]$ V with steps of 50 mV to cover the operating range of our ECUs. The column in A and B hold the values of a_0^f and a_1^f explained in Eq. 10. Depending on the type of feature f (low/high of CANH/CANL), the A and B values would be:

$$\begin{aligned} A &= \left[a_0^f(v_1), a_0^f(v_2), \cdots, a_0^f(v_n)\right]^{\mathsf{T}}, \\ B &= \left[a_1^f(v_1), a_1^f(v_2), \cdots, a_1^f(v_n)\right]^{\mathsf{T}}, \end{aligned} \tag{11}$$

assuming there are n voltage levels.

After generating the A and B matrices we fit another linear regression to elements of each matrix, trying to estimate the corresponding $\hat{a}_0^f$ and $\hat{a}_1^f$ for another voltage level in *source domain*, $\hat{v}$, for which we do not have access to training data. In fact, we are able to estimate the scaling parameters for mapping the features from any random voltage level within the operating range of our device up to the *target domain*, later validated in Sec. 6.2.

6.1.2 Impact of Temperature. At all rounds of data collection, the temperature values are collected and stored on a per frame basis for further analysis. We investigate the possibility of a linear relationship between the features and temperature in a controlled setup, which emulates the rise of temperature once the engine gets started. Such a linear relationship can be modelled using Eq. 12,

$$F = c_0^f * T + c_1^f \tag{12}$$

where F is the vector of features, T is the internal temperature, and c_0^f and c_1^f are the estimation parameters for feature set f which can be one of the four different combinations of features (high/low bins of CANH/CANL), later validated in Sec. 6.2.

6.2 Evaluation

An experimental setup is used for emulating the ECUs that are connected via a CAN bus and send messages that are crafted based on CAN protocol. The setup includes TM4C123GXL micro-controllers [46] integrated on a TivaC launchpad which are programmed to send extended CAN (2.0b) messages with 29 bit identifiers, where the identifier and data section for each frame in each round of data collection are configured to be random. These devices have exactly the same configuration, have been built by the same manufacturer, and purchased at the same time. The ECUs send a CAN message every 1 s with data rate of 110 Kbps, and a USB2523-MCC DAQ [32] with 12 bits of resolution is used to perform sampling at rate of 500 Ksps on each channel associated with CANH and CANL. Fig. 7a illustrates the basic setup that was used for data collection.

6.2.1 Results for Constant Voltage with Natural Variations in Temperature. We emulate 10 ECUs in this setup, and collect 10,000 CAN frames from each. Next we select 2 % of the data at random to extract feature templates for each ECU and generate thresholds. The remaining data is used for the testing phase. The ECUs take turns to be the legitimate device in each round of analysis which is indicated as Case i where the i^{th} ECU is the legitimate one. The EERs and the thresholds associated with them are reported in Table 3a.

Table 3a shows that the features generated based on SIMPLE are able to distinguish perfectly between the devices when Vcc is constant and give EER values close to 0% (the ideal EER value).

6.2.2 Results for Changing Voltage with Natural Variations in Temperature. We collect new rounds of data from devices when the Vcc is reduced by steps of 50 mV, within the operating voltage range of the transceivers covering Vcc $\in \{3.00, 3.05, 3.10, 3.15, 3.20, 3.25\}$. The primary analyses demonstrate as low EER results as 0% which are not presented here due to space limitations.

Next, we validate the proposed solution provided in Sec. 6.1, which accounts for the variations in the voltage levels with no data available for exact scaling parameter training. We collect new data at different voltages than the ones mentioned above, but within the operating range of our transceiver (3.3 V with drop value of within 0.3 V [46]). Now results of two sample voltage levels, Vsr = 3.275 V and Vsr = 3.225 V are given here as representatives of several rounds of data collection and testing. Tables 3b and 3c give the EER values and their corresponding thresholds for after using the estimated scaling parameters for mapping features from source domains (no training data) to the target domain (Vtg = 3.30 V here).

The Mahalanobis distances of features calculated during test phase of Case 2 (where all ECUs take turns to send malicious frames on behalf of ECU02, whereas ECU02 is sending legitimate frames) are plotted on top on each other as a sample result in Fig. 6b, which shows the capability of SIMPLE to compensate for Vcc variation.

6.2.3 Results for Constant Voltage with Natural Variations in Temperature. In all rounds of data collection, the temperature values are collected with an ADC and temperature sensor integrated in the same TivaC LaunchPad [26], and stored for each frame for further analysis. We train our templates for each device on different days with different conditions such as no air conditioning. Fig. 6a is a sample result when other ECUs take turns doing a masquerade attack on ECU01. After accounting for the variations in the Vcc level, and are able to identify the malicious frames with EER of zero, regardless of the temperature differences during the training and testing phase. Repeating the same analysis on more than 70 rounds of data collection on different days, and EER of zero for all, allows us conclude that features used by SIMPLE are robust enough that they are not affected by natural temperature changes up to $\pm 6^{\circ}C$.

7 EXPERIMENTAL VALIDATION

The above in-lab experiments, carried out using commonly available parts, allowed us to determine the factors that affect ECU fingerprints. In this section we extend this analysis by examining emulated ECUs that are architected to reflect actual automotive ECUs and built using automotive grade parts.

7.1 In-lab Experiments

Automotive grade transceivers, reported in Table 2, were used for building ECUs on a CAN bus shown in Fig. 7b. TM4C123GXL microcontrollers were used as the CAN protocol controller, and TI voltage regulators for proper powering of the devices (Fig. 5d).

Table 2: Automotive grade transceivers on the CAN bus prototype.

CAN transceiver	Manufacturer	Quantity	Voltage Regulator
TJA1050 [41]	NXP	3	TLV1117-50 [23]
NCV7340 [25]	ON Semiconductor	1	TLV1117-50 [23]
HA13721 [11]	Renesas	2	TLV1117-50 [23]
TCAN332 [24]	Texas Instruments	2	TLV1117-33 [23]
MAX3051 [27]	Maxim Integrated	2	TLV1117-33 [23]

7.1.1 Results for Varying Voltage with Natural Variations Temperature. The results of the lab experiments presented in Sec. 6.2.1, illustrate that the fingerprints deviate mainly due to changes in voltage source values. However, in this setup as in actual automotive ECUs, Vcc was tightly controlled via a voltage regulator in the powering circuitry. In order to analyse how voltage drops in Vcc change the fingerprints, we tested extreme cases where the Vcc was dropped with steps of 1 Volt, covering values of $V_{sr} = \{8, 9, 10, 11, 12\}$ V and collected 10,000 sample frames. Next, we did the feature transformation conducted in Sec. 6.1.1 to scale features that were collected in a source voltage level domain (could be any value out of $V_{sr} = \{8, 9, 10, 11\}$ V) up to a target voltage domain of choice (12 V in this work). Repeating the same procedure as Sec. 6.1.1, we were able to estimate the new scaling coefficients (Eq. 10) for a randomly selected voltage level in the continuous interval of $V_{sr} \in [8 : 12]$ V without collecting training data. Table 3d illustrates the conventional analysis where we collected 10,000 sample frames from the ECUs on the bus when $V_{cc} = 12.00$ V and using K-fold cross validation for the EER analysis. In results shown at Table 3f, the feature transformation coefficients were trained on the data collected at $V_{sr} = \{9, 10, 11\}$ V and validated on the data collected at $V_{sr} = 8.00$ V. Fig. 6c shows a sample result, where the data collected from the ECU with HA13721-02 (HA02) transceiver at $V_{sr} = 8.00$ was scaled up to $V_{tg} = 12.00$ and tested against any other existing ECU on the bus that could impersonate it. As shown in the temperature graph, natural variations in temperature did not affect the proper performance of SIMPLE.

Table 4: Cross validating a linear regression for modelling the change of features with temperature.

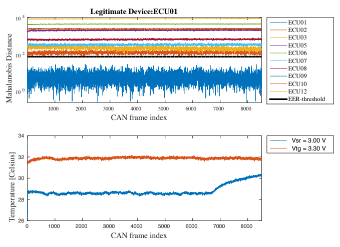
CAN transceiver	Correlation type with high-of-CAN	$\overline{R^2}$ (round-a)	$\overline{\text{MSE}}$ (round-b)
TJA1050 [41]	negative	0.9874	0.0002
NCV7340 [25]	negative	0.9330	0.0132
HA13721 [11]	negative	0.9684	0.0165
TCAN332 [24]	positive	0.9812	0.0180
MAX3051 [27]	positive	0.9850	0.0168

7.1.2 Results for Constant Voltage with Controlled High Temperature. Here we slightly modified the benchtop setup by using a cardboard covering and increasing the temperature of the ECUs using a heat gun during data collection. Fig. 8a illustrates how the first feature of NCV7340 changed over time in the upper graph, with the changes in the temperature is depicted in the lower graph. We use the linear regression model given at Eq. 12 to correlate the features of every device to temperature values, a sample of which shown at Fig. 8b for NCV7340. The average R^2 values of this fitting model are given at table 4, where the averaging was performed over different features. R^2 values of close to unity for all devices, validates that there was a linear relationship between the features and the temperature. We observe both positive and negative correlations between the feature and temperature for different devices, which is a consistent behaviour based on repetitive experiments on each device. Next, we tested the trained linear model on another round of experiment (round-b). The average of mean square error (MSE) values of estimation of features for round-b given at Table 4 are close to zero which validates the goodness of the fit in this model. Note that the graphs in this section include the first feature as a representative of the rest of features which show similar behaviour.

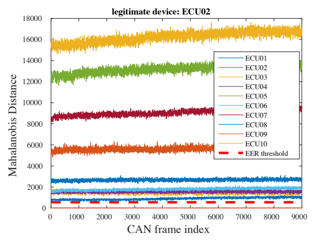
In summary, SIMPLE is robust to variations in temperature and supply voltage values.

7.2 In-vehicle Experiments

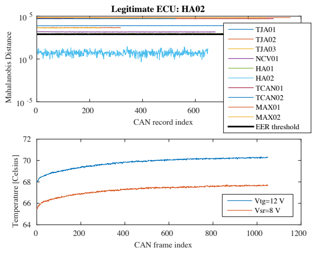
We collected CAN messages from two vehicles, Nissan Sentra 2016 (Fig. 9a) and Subaru Outback 2011 (Fig. 9b), via the OBD-II port using a Tektronix DPO 3012 oscilloscope shown in Fig. 9c. The oscilloscope sampled at 50 Msps per channel with 8 bits of resolution



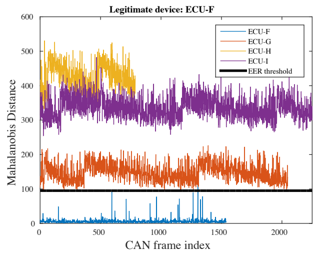
(a)



(b)



(c)



(d)

Figure 6: (a) The effect of temperature on fingerprinting process of ECU1. The training is done at Vsr = 3.00 V on an ambient temperature different from the testing phase with voltage level of Vtg = 3.30 V. Natural variations of temperature values are plotted in the lower graph with respect to CAN frame indices. (b) The Mahalanonbis distance of CAN frames transmitted from all ECUs from the template feature of ECU02. The source voltage is set to Vsr = 3.225 V and Vtg = 3.30 V. (c) The Mahalanonbis distance of CAN frames transmitted from all ECUs from the template feature of ECU:HA02. The source voltage is set to Vsr = 8.00 V and Vtg = 12.00 V. Natural variations of temperature values are plotted in the lower graph with respect to CAN frame indices. The ambient temperature is different at training and testing process, however, it does not affect the functionality of the identification process. (d) The Mahalanonbis distance of CAN frames transmitted from all ECUs from the template feature of ECU:F collected from Subaru Outback.

Table 3: The EER values and their corresponding thresholds (Mahalanobis distances). ECUi corresponds to the i^{th} emulated ECU being the legitimate device. Tables (b)(c)(e)(f) show that we were able to test the frames transmitted from ECUs at different source voltages without having training data for that source level.

(a) $V_{cc} = 3.30$ V

	EER	Thresh.		EER	Thresh.
ECU1	0%	104.238	**ECU6**	0%	245.553
ECU2	0%	120.533	**ECU7**	0%	592.987
ECU3	0%	126.051	**ECU8**	0.007%	46.585
ECU4	0.007%	49.325	**ECU9**	0.001%	35.116
ECU5	0%	860.617	**ECU10**	0%	1047.8

(b) $V_{sr} = 3.225$ V, **and** $V_{cc} = 3.30$ V

	EER	Thresh.		EER	Thresh.
ECU1	0%	235.561	**ECU6**	0%	523.422
ECU2	0%	555.872	**ECU7**	0%	1958.2
ECU3	0%	675.377	**ECU8**	0%	737.364
ECU4	0%	789.302	**ECU9**	0%	573.653
ECU5	0%	157.287	**ECU10**	0%	145.564

(c) $V_{sr} = 3.275$ V, **and** $V_{cc} = 3.30$ V

	EER	Thresh.		EER	Thresh.
ECU1	0%	208.378	**ECU6**	0%	578.659
ECU2	0%	112.142	**ECU7**	0%	341.115
ECU3	0%	143.402	**ECU8**	0.02%	45.741
ECU4	0%	982.568	**ECU9**	0.05%	44.206
ECU5	0%	1290.6	**ECU10**	0%	1235.5

(d) $V_{cc} = 12.00$ V

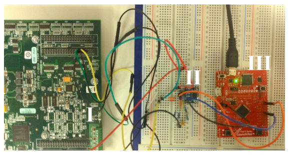
	EER	Thresh.		EER	Thresh.
ECU1	0%	9174.5	**ECU6**	0%	473.877
ECU2	0%	811.913	**ECU7**	0%	81.51
ECU3	0%	1514.2	**ECU8**	0%	87.171
ECU4	0%	8926.5	**ECU9**	0%	1589.7
ECU5	0%	868.247	**ECU10**	0%	501.142

(e) $V_{sr} = 10.00$ V, **and** $V_{cc} = 12.00$ V

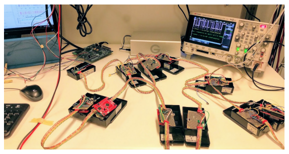
	EER	Thresh.		EER	Thresh.
ECU1	0%	5595.2	**ECU6**	0%	456.227
ECU2	0%	394.373	**ECU7**	0%	60.7209
ECU3	0%	702.926	**ECU8**	0%	306.410
ECU4	0%	6870.8	**ECU9**	0%	2296
ECU5	0%	440.148	**ECU10**	0%	52.9382

(f) $V_{sr} = 8.00$ V, **and** $V_{cc} = 12.00$ V

	EER	Thresh.		EER	Thresh.
ECU1	0%	$3.7624e + 03$	**ECU6**	0%	730.6701
ECU2	0%	752.0212	**ECU7**	2.3%	13.7138
ECU3	0.04%	400.4652	**ECU8**	0.52%	19.1959
ECU4	0%	$1.0970e + 03$	**ECU9**	0%	1515.6
ECU5	0%	$1.1843e + 03$	**ECU10**	0.08%	861.1623



(a)



(b)

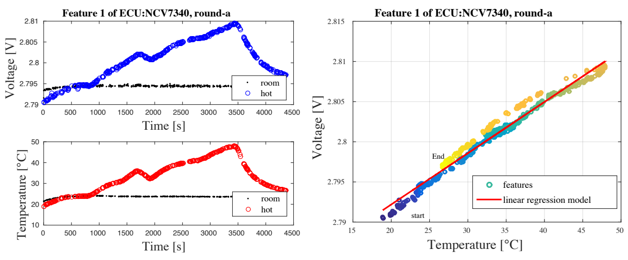
Figure 7: (a) Emulated setup for collecting CAN frames from ECUs. I: MCC DAQ, II: SN65HVD230 CAN transceiver [47], and III: TivaC micro-controller (emulating CAN controller). (b) Benchtop CAN bus setup, includes 10 ECUs listed at Table 2. Each ECU is designed based on the diagram given at Figure 5d.

and sent records to a computer via USB connection. We drove the vehicles for about forty minutes in each round, including local and highway, and collected over 16,000 frames. The in-vehicle CAN bus voltage dataset has been made public [5].

In another round of data collection shown at Fig. 9d, we collected a million CAN messages with timestamps in microseconds using a PCAN-USB device [22] and the python-can library [48] to collect these messages while driving the vehicles for about ten minutes.

7.2.1 Ground truth establishment. In the data collected from real-world vehicles, frames were labelled with IDs only. Since one ECU can send messages with different IDs, we needed to associate these frames with ECUs as well (because SIMPLE aims to identify ECUs, not IDs). Hence we used Viden and CIDS to associate IDs with ECUs. Fig. 13 shows these results. For the Nissan Sentra, messages IDs fell into the ECU clusters of {374, 375}, {644, 645, 646} , {386} , {533, 534}, and {849}, which will be referred to as ECUs $\mathbb{A}$, $\mathbb{B}$, $\mathbb{C}$, $\mathbb{D}$, and $\mathbb{E}$ hereafter. For the Subaru Outback, IDs fell into {817, 818, 819, 820}, {593, 594, 595}, {849, 850, 855}, and {561, 562, 565}, which will be referred to as ECU $\mathbb{F}$ through $\mathbb{I}$.

[5]https://github.com/harry1993/simple-dataset



(a) (b)

Figure 8: (a) The effect of temperature on features of ECU:NCV7340 with respect to time. The features are shown in the upper graph (blue) and the tempearture values are given in the lower graph (red). In both cases a data collected in room temperature is given for the sake of comparison (black). (b) Modelling the change of features of ECU:NCV7340 with hot temperature using linear regression on round-a of data collection. The drift in the features with respect to time is color coded, starting from dark blue and ending at yellow.

7.2.2 *Hill-climbing-style attacks results.*

We evaluated our attacks on both voltage-based and clock-based fingerprinting schemes, using the real-world data. We ran our attacks on ECUs $\mathbb{A}$ and $\mathbb{B}$'s data to demonstrate how $\mathbb{A}$ was able to impersonate $\mathbb{B}$. The results are plotted in Fig. 10. We can observe that $\mathbb{B}$'s profile was shifted gradually to $\mathbb{A}$'s while it never crossed the decision threshold. The dot markers indicate the moments when the number of attack frames, m was increased by one, i.e., injection ratio r was increased. In Fig. 10b, the injection offset δ was also increased because $S_{\mathbb{B}}$ was increased along with r. We plot the results up to only 140 steps due to the page limit. But from the trend of r, we can see that it will approach one.

We also calculated the average of number of steps the attack needed to achieve dominant impersonation. We ran our attack on every pair of ECUs whose profiles were next to each other. Since there were multiple IDs associated with one ECU, we were able to calculate the average numbers of steps. Results show that the number of steps increased with the difference of voltage profiles between two ECUs. E.g, the difference of voltage profiles of ECUs $\mathbb{A}$ and $\mathbb{B}$ was 0.05; $\mathbb{A}$ impersonating $\mathbb{B}$ took averagely 115.5 steps. $\mathbb{B}$ impersonating $\mathbb{C}$ required 171.75 steps because their profiles' difference was 0.089.

7.2.3 *SIMPLE's identification results.*

After associating frames with ECUs, we tested SIMPLE's ability to identify the source of a message when other ECUs impersonate the legit device. The results in terms of EERs, with their corresponding thresholds, are given at Table 5. It should be noted that the differential signal was used for feature extraction for noise cancellation purposes.

7.2.4 *Robustness against the Hill-climbing-style attack.*

Unlike previous CAN IDS schemes, SIMPLE makes an identification decision for each individual frame. On average, the source of each frame is correctly identified (and intrusion is detected) with probability of 1 − EER. We note that the hill-climbing style attack is no longer feasible against SIMPLE, because the identification threshold is not updated using the features or voltage profiles across multiple frames. Rather, it is updated based on supply voltage or temperature measurements (which are assumed to be secure). Thus, even if the attacker injects multiple malicious messages it cannot shift the voltage profile. On the other hand, even though the EERs from our in-vehicle experiments are not 0% in some cases, if the attacker injects multiple messages, the intrusion will be detected with a probability that approaches one exponentially. Finally, we note that it is infeasible for an attacker to generate malicious frames that exactly mimic the benign frames by replicating their features, since that will require an Arbitrary Waveform Generator (AWG) [13].

Table 5: The EER values and their corresponding thresholds (Mahalanobis distances) . The first column in each row corresponds to ECU being the legitimate device from Nissan Sentra and Subaru Outback.

Nissan March 24	EER	Thresh.
ECU $\mathbb{A}$	0.0372%	66.12
ECU $\mathbb{B}$	0.0342%	113.51
ECU $\mathbb{C}$	0.3766%	19.96
ECU $\mathbb{D}$	2.3824%	26.60
ECU $\mathbb{E}$	0.0238%	1881

Nissan Feb 01	EER	Thresh.
ECU $\mathbb{A}$	0%	78.0204
ECU $\mathbb{B}$	0.0297%	123.5658
ECU $\mathbb{C}$	0.2330%	20.0671
ECU $\mathbb{D}$	0.3434%	35.1643
ECU $\mathbb{E}$	0.0202%	$2.7189e+03$

Nissan Feb 18	EER	Thresh.
ECU $\mathbb{A}$	0.1899%	26.4375
ECU $\mathbb{B}$	0.1151%	48.2290
ECU $\mathbb{C}$	3.7573%	14.5754
ECU $\mathbb{D}$	3.3665%	15.3776
ECU $\mathbb{E}$	0%	5031.4

Nissan Feb 21	EER	Thresh.
ECU $\mathbb{A}$	0%	110.5868
ECU $\mathbb{B}$	0.0487%	140.2214
ECU $\mathbb{C}$	0.05%	47.4978
ECU $\mathbb{D}$	0.5250%	39.8275
ECU $\mathbb{E}$	0%	$1.5860e+177$

Subaru	EER	Thresh.
ECU $\mathbb{F}$	0.0913%	122.32
ECU $\mathbb{G}$	0.1101%	27.09
ECU $\mathbb{H}$	4.3496%	13.22
ECU $\mathbb{I}$	5.48%	12.78

7.2.5 *Stability analysis.*

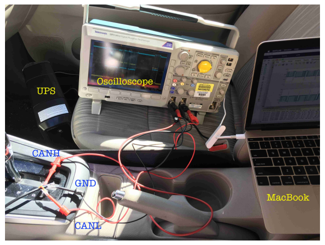
Long term data collection from Nissan Sentra shows drift in the features within the period of five months. Based on experiments in Sec. 7.1.2, the drift in the features that were used by SIMPLE was linearly correlated with change in the temperature. Using the average of the estimated linear coefficients derived in Sec. 7.1.2, we find the temperature estimates for ECU-A of Nissan Sentra plotted in Fig. 11. The temperature estimates for the ECU-A were in the range of 45°C to 52°C, which is a valid range for ECUs in a running vehicle. The reason of not having original ECU temperature data is because the ECU locations are hard to access, so we validate the model by predicting the ECU temperature from features and see if they fall into a reasonable range.The average ambient temperature in Tucson, Arizona on the day of the data collection is also illustrated in the figure, which shows a negative correlation between the ambient temperature changes and the drift in the features. This observation is in agreement with temperature estimates for ECU-A. ECUs in a vehicle were prone to drastic temperature changes and that is the reason they had a wide range of operating temperature. This affected the features used by any voltage-based IDS, hence cannot be overlooked.

8 CONCLUSION AND FUTURE WORK

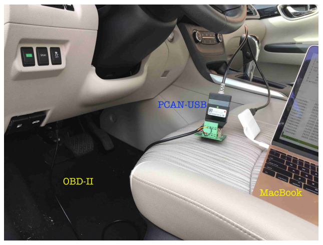
We demonstrated the vulnerability of the existing multi-frame based automotive intrusion detection and identification systems to a Hill-climbing-style attack, which allows a compromised ECU to impersonate another. We also showed the feasibility of our attacks

(a) Nissan Sentra 2016

(b) Subaru Outback 2011

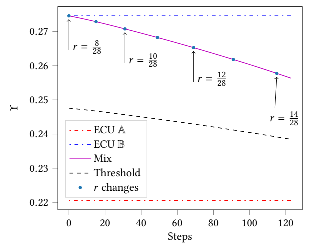


(c) CAN bus voltage measurement

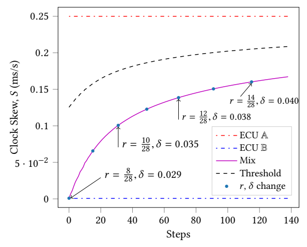


(d) CAN messages logging

Figure 9: Two experiment vehicles and experimental setups.



(a) Dominant/complete impersonation against the voltage-based scheme



(b) Complete impersonation against the clock-based scheme

Figure 10: Hill-climbing-style attacks on both schemes.

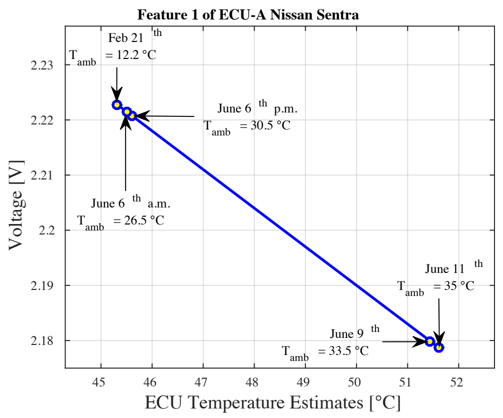


Figure 11: Validation of linear relationship between the observed drift in the features of ECU-A from Nissan Sentra and temperature over the period of five months. T_{amb} shows the average reported ambient temperature during the day for each date of data collection from the moving vehicle.

against Viden [8] and CIDS [9]. Next, we introduced SIMPLE, a novel intrusion detection and identification system for in-vehicle networks that is immune to this type of attack. Our detection system uses physical layer features within a single frame to fingerprint the ECUs on a CAN bus. In addition to the reliability and perfectly distinguishing a legit device from a non-legit one (the average EER is close to 0 % in in-lab, and 0.8985 % in in-vehicle experiments), what makes SIMPLE unique is its practicality. It requires a relatively low sampling rate, a single-frame for detection, and incurs low timing complexity and overhead. It is also able to account for the variations in the ambient conditions such as temperature and the supply voltage values. In future, we are interested in how different versions of firmware change the voltage output characteristics of an ECU. Chilenski et al. [6] discuss how the side-channel analysis of RF emissions relate to the firmware's execution.

ACKNOWLEDGMENTS

This work was supported in part by NSF under grant numbers CNS-1410000, CNS-1801402, CPS-1658225, and by ARO under the grant number W911NF-19-1-0050.
M. Foruhandeh is primarily responsible for the defence solution, while Y. Man is solely responsible for the attack design.

REFERENCES

[1] Omid Avatefipour, Azeem Hafeez, Muhammad Tayyab, and Hafiz Malik. 2017. Linking received packet to the transmitter through physical-fingerprinting of controller area network. In *IEEE International Workshop on Information Forensics and Security (WIFS)*. 1–6.
[2] Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer. http://research.microsoft.com/en-us/um/people/cmbishop/prml/
[3] Ruud M Bolle, Jonathan H Connell, Sharath Pankanti, Nalini K Ratha, and Andrew W Senior. [n. d.]. Guide to biometrics. 2004. *H. Hakobyan et al./Human Identification Using Virtual 3D Imaging to Control Border Crossing* 230 ([n. d.]).
[4] Robert Bosch. 1991. *CAN Specification v2.0.* Technical Report. Bosch.
[5] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, Tadayoshi Kohno, et al. 2011. Comprehensive Experimental Analyses of Automotive Attack Surfaces.. In *USENIX Security Symposium*. San Francisco.
[6] Mark Chilenski, George Cybenko, Isaac Dekine, Piyush Kumar, and Gil Raz. 2018. Control flow graph modifications for improved RF-based processor tracking performance. In *Cyber Sensing 2018*, Vol. 10630. International Society for Optics and Photonics, 106300I.
[7] Kyong-Tak Cho and Kang G Shin. 2016. Error handling of in-vehicle networks makes them vulnerable. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1044–1055.
[8] Kyong-Tak Cho and Kang G Shin. 2016. Fingerprinting Electronic Control Units for Vehicle Intrusion Detection.. In *USENIX Security Symposium*. 911–927.
[9] Kyong-Tak Cho and Kang G Shin. 2017. Viden: Attacker Identification on In-Vehicle Networks. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1109–1123.
[10] Wonsuk Choi, Kyungho Joo, Hyo Jin Jo, Moon Chan Park, and Dong Hoon Lee. 2018. VoltageIDS: Low-Level Communication Characteristics for Automotive Intrusion Detection System. *IEEE Transactions on Information Forensics and Security* (2018).
[11] Renesas Electronics Corporation. 2004. HA13721 High speed CAN transceiver datasheet.
[12] Austin Costley, Chase Kunz, Ryan Gerdes, and Rajnikant Sharma. 2017. Low Cost, Open-Source Testbed to Enable Full-Sized Automated Vehicle Research. *arXiv preprint arXiv:1708.07771* (2017).

[13] Boris Danev, Heinrich Luecken, Srdjan Capkun, and Karim El Defrawy. 2010. Attacks on Physical-layer Identification. In *Proceedings of the third ACM conference on Wireless network security (WiSec '10)*, Vol. 0. ACM, New York, NY, USA, 89–98.
[14] Boris Danev, Davide Zanetti, and Srdjan Capkun. 2012. On physical-layer identification of wireless devices. *ACM Computing Surveys (CSUR)* 45, 1 (2012), 6.
[15] Robert I Davis, Alan Burns, Reinder J Bril, and Johan J Lukkien. 2007. Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised. *Real-Time Systems* 35, 3 (2007), 239–272.
[16] Sibylle Fröschle and Alexander Stühring. 2017. Analyzing the capabilities of the CAN attacker. In *European Symposium on Research in Computer Security*. Springer, 464–482.
[17] Ryan M Gerdes and Saptarshi Mallick. 2015. Physical-Layer Detection of Hardware Keyloggers. In *International Workshop on Recent Advances in Intrusion Detection*. Springer, 26–47.
[18] R. M. Gerdes, M. Mina, S. F. Russell, and T. E. Daniels. 2012. Physical-Layer Identification of Wired Ethernet Devices. *IEEE Transactions on Information Forensics and Security* 7, 4 (Aug 2012), 1339–1353. https://doi.org/10.1109/TIFS.2012.2197746
[19] Andy Greenberg. 2015. Hackers Remotely Kill a Jeep on the Highway—With Me in It. *Wired* (Dec 2015).
[20] Bogdan Groza and Pal-Stefan Murvay. 2018. Security Solutions for the Controller Area Network: Bringing Authentication to In-Vehicle Networks. *IEEE Vehicular Technology Magazine* 13, 1 (2018), 40–47.
[21] Tobias Hoppe, Stefan Kiltz, and Jana Dittmann. 2011. Security threats to automotive CAN networksâĂŤPractical examples and selected short-term countermeasures. *Reliability Engineering & System Safety* 96, 1 (2011), 11–25.
[22] Grid Connect Inc. 2017. PCAN-USB FD, CAN USB Flexible Data Rate Adapter. https://gridconnect.box.com/shared/static/oggfdg2b1w46rpz821v57rsnpto4jw4g.pdf
[23] Texas Instruments Incorporated. 2014. TLV1117, Adjustable and Fixed Low-Dropout Voltage Regulator.
[24] Texas Instruments Incorporated. 2015-2016. TCAN33x 3.3-V CAN Transceivers with CAN FD (Flexible Data Rate).
[25] Semiconductor Components Industries. 2014. NCV7340 High speed CAN transceiver datasheet.
[26] Texas Instruments. 2013. TivaâĎć C Series TM4C123G LaunchPad Evaluation Board User's Guide.
[27] Maxim Integrated. 2014. MAX3051 CAN transceiver datasheet.
[28] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li. 2018. Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning. In *2018 IEEE Symposium on Security and Privacy (SP)*. 19–35. https://doi.org/10.1109/SP.2018.00057
[29] Marcel Kneib and Christopher Huth. 2018. Scission: Signal characteristic-based sender identification and intrusion detection in automotive networks. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 787–800.
[30] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, et al. 2010. Experimental security analysis of a modern automobile. In *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 447–462.
[31] Moti Markovitz and Avishai Wool. 2017. Field classification, modeling and anomaly detection in unknown CAN bus networks. *Vehicular Communications* 9 (2017), 43–52.
[32] Measurement Computing Corporation 2016. *Multi-function data acquisition*. Measurement Computing Corporation.
[33] Charlie Miller and Chris Valasek. 2015. Remote exploitation of an unaltered passenger vehicle. *Black Hat USA* 2015 (2015).
[34] Charlie Miller and Chris Valasek. 2016. CAN Message Injection - OG Dynamite Edition. http://illmatics.com/can%20message%20injection.pdf
[35] Michael R Moore, Robert A Bridges, Frank L Combs, Michael S Starr, and Stacy J Prowell. 2017. Modeling inter-signal arrival times for accurate detection of CAN bus signal injection attacks: a data-driven approach to in-vehicle intrusion detection. In *Proceedings of the 12th Annual Conference on Cyber and Information Security Research*. ACM, 11.
[36] Pal-Stefan Murvay and Bogdan Groza. 2014. Source identification using signal characteristics in controller area networks. *IEEE Signal Processing Letters* 21, 4 (2014), 395–399.
[37] Michael Müter and Naim Asaj. 2011. Entropy-based anomaly detection for in-vehicle networks. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*. IEEE, 1110–1115.
[38] Michael Müter, André Groll, and Felix C Freiling. 2010. A structured approach to anomaly detection for in-vehicle networks. In *Information Assurance and Security (IAS), 2010 Sixth International Conference on*. IEEE, 92–98.
[39] Dennis K Nilsson, Ulf E Larson, and Erland Jonsson. 2008. Efficient in-vehicle delayed data authentication based on compound message authentication codes. In *Vehicular Technology Conference, 2008. VTC 2008-Fall. IEEE 68th*. IEEE, 1–5.
[40] Sang Uk Sagong, Xuhang Ying, Andrew Clark, Linda Bushnell, and Radha Poovendran. 2017. Cloaking the Clock: Emulating Clock Skew in Controller Area Networks. *arXiv preprint arXiv:1710.02692* (2017).
[41] Philips Semiconductors. 1999. TJA1050 High speed CAN transceiver datasheet.
[42] Colin Soutar et al. 2002. Biometric system security. *White Paper, Bioscrypt, http://www. bioscrypt. com* (2002).
[43] Christopher Johnathan Szilagyi. 2012. *Low cost multicast network authentication for embedded control systems*. Ph.D. Dissertation. Carnegie Mellon University.
[44] Adrian Taylor, Sylvain Leblanc, and Nathalie Japkowicz. 2016. Anomaly Detection in Automobile Control Network Data with Long Short-Term Memory Networks. In *Data Science and Advanced Analytics (DSAA), 2016 IEEE International Conference on*. IEEE, 130–139.
[45] Linear Technologies. 2018. LTC1743—12-Bit, 50Msps ADC. https://www.analog.com/en/products/ltc1743.html#product-documentation
[46] Texas Instruments Incorporated 2014. *Microcontroller data sheet*. Texas Instruments Incorporated.
[47] Texas Instruments Incorporated 2016. *CAN transceiver*. Texas Instruments Incorporated.
[48] Brian Thorne. 2019. python-can, controller area network support for Python developers. https://python-can.readthedocs.io
[49] Anthony Van Herrewege, Dave Singelee, and Ingrid Verbauwhede. 2011. CANAuth-a simple, backward compatible broadcast authentication protocol for CAN bus. In *ECRYPT Workshop on Lightweight Cryptography*, Vol. 2011.
[50] Cliff Wang, Ryan M Gerdes, Yong Guan, and Sneha Kumar Kasera. 2016. *Digital fingerprinting*. Springer.
[51] Louis F Williams Jr. 1976. A modification to the half-interval search (binary search) method. In *Proceedings of the 14th annual Southeast regional conference*. ACM, 95–101.
[52] Marko Wolf, André Weimerskirch, and Christof Paar. 2004. Security in automotive bus systems. In *Workshop on Embedded Security in Cars*.
[53] Xuhang Ying, Giuseppe Bernieri, Mauro Conti, and Radha Poovendran. 2019. TACAN: Transmitter Authentication through Covert Channels in Controller Area Networks. *arXiv preprint arXiv:1903.05231* (2019).
[54] Wenyi Zhao, Arvindh Krishnaswamy, Rama Chellappa, Daniel L Swets, and John Weng. 1998. Discriminant analysis of principal components for face recognition. In *Face Recognition*. Springer, 73–85.
[55] Tobias Ziermann, Stefan Wildermann, and Jurgen Teich. 2009. CAN+: A new backward-compatible Controller Area Network (CAN) protocol with up to 16× higher data rate. In *Design, Automation & Test in Europe Conference & Exhibition, 2009. DATE'09*. IEEE, 1088–1093.

A PRELIMINARIES OF VIDEN AND CIDS

In this section of the appendix, we will re-state the details of CIDS and Viden.

A.1 CIDS

In order to detect the intrusion and identify the attacker ECU, Cho et al. proposed Clock-based IDS (CIDS) where the clock skews are used as the fingerprints of ECUs. This is based on the fact that the clock in every ECU advances differently. The clock skew is defined as the difference between the advancing rate of the estimated clock and the true clock. For example, after t seconds, a clock reports the elapsed time as t' seconds. The skew of this clock is then $\frac{t'-t}{t}$. However, since a CAN frame does not contain a timestamp, CIDS updates the clock skew by evaluating the arrival timestamps of a batch of n messages, i.e., the moments when these messages arrive at the receiver.

Specifically, at k-th step (during $t[k-1]$ to $t[k]$), n arrival timestamps (a_i, for $n = 1 \ldots n$) are recorded. The interval between i-th and $i-1$-th arrival timestamps is $T_i = a_i - a_{i-1}$. CIDS calculates the upper and lower control limits $L^+[k]$ and $L^-[k]$ as follows:

$$\mu_T[k] \leftarrow \frac{1}{n}\sum_{i=1}^{n} T_i, \tag{13a}$$
$$O[k] \leftarrow \frac{1}{n-1}\sum_{i=2}^{n} a_i - (a_1 + (i-1)\mu_T[k-1]), \tag{13b}$$
$$O_{acc}[k] \leftarrow O_{acc}[k-1] + |O[k]|, \tag{13c}$$
$$e[k] \leftarrow O_{acc}[k] - S[k-1]t[k], \tag{13d}$$
$$\mu_e[k] \leftarrow \frac{1}{k}\sum_{i=1}^{k} e[i], \tag{13e}$$
$$\sigma_e^2[k] \leftarrow \frac{1}{k}\sum_{i=1}^{k}(e[i]-\mu_e[k])^2, \tag{13f}$$
$$L^+[k] \leftarrow \max\left[0, L^+[k-1] + \frac{e[k]-\mu_e[k]}{\sigma_e[k]} - \kappa\right], \tag{13g}$$
$$L^-[k] \leftarrow \max\left[0, L^-[k-1] - \frac{e[k]-\mu_e[k]}{\sigma_e[k]} - \kappa\right]. \tag{13h}$$

If either of the control limits $L^+[k]$ or $L^-[k]$ exceeds $\Gamma_L = 5$, CIDS declares a detection of intrusion. If the adversary wants to defeat the intrusion detection, it can simply add the following constraints to Equation 6:

$$L^+[k] < \Gamma_L \tag{14a}$$
$$L^-[k] < \Gamma_L \tag{14b}$$

With the *accumulated clock offset* O_{acc}, the *identification error* $e[k]$ and the elapsed time $t[k]$, a linear parameter identification problem can be formulated as:

$$O_{acc}[k] = S[k] \cdot t[k] + e[k], \tag{15}$$

where the *clock skew* $S[k]$ (i.e., the slope) can be learnt using the Recursive Least Squares (RLS) algorithm. The slope $S[k]$ is viewed as the *norm clock behavior* that CIDS uses to determine the expected timing behavior of the legitimate ECU. See Figure 12 for an illustration of these terms.

A.2 Viden

Since CIDS can deal with only the case when the messages are sent periodically, Cho et al. proposed the Voltage-based attacker identification (Viden) [9] that learns the voltage output of the transmitters as the fingerprints of them. Cho et al. assumed there is a perfect underlying IDS that detects intrusions in the first place. Once an intrusion is detected, the suspicious messages are submitted to CIDS [8]. CIDS tries to identify the source of these messages. If CIDS fails, these messages are then submitted to the voltage-based identification model.

Viden is detailed in Algorithm 1. Whenever κ dominant voltage values are sampled, Viden derives the *voltage instance* $v_{1\dots6}$ from the latest κR samples (Line 5 through Line 10), representing the momentary voltage output character. Then, Viden uses the latest voltage instance to update the *voltage profile* Υ (Line 11 through

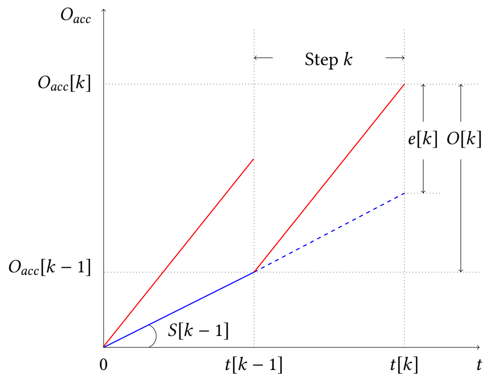


Figure 12: Accumulated clock offsets O_{acc}. From time 0 to $t[k-1]$, ECU $\mathbb{A}$ sends messages $m_{\mathbb{A}}$. Its O_{acc} is plotted in red solid line. Meanwhile, ECU $\mathbb{B}$ sends messages $m_{\mathbb{B}}$, plotted in blue solid line. From $t[k-1]$ to $t[k]$, the adversary mounts a masquerade attack, where $\mathbb{B}$ is suspended and $\mathbb{A}$ is programmed to send $m_{\mathbb{B}}$ instead. $m_{\mathbb{B}}$'s new O_{acc} is plotted in red solid line, which is different from the norm clock behavior (the blue dash line). The identification error $e[k]$ indicates how far the accumulated clock offset deviates from CIDS's expectation at $t[k]$. Based on $e[k]$, CIDS decides whether the intrusion exists or not. Furthermore, the slopes of two red solid segments being similar, provides the identification information. In other words, this tells CIDS that the attack messages are sent by ECU $\mathbb{A}$.

Algorithm 1: Update the voltage profile Υ at step k

```
1  function UpdateDispersion($V, \Lambda, P^*$):
2      return $\Lambda \leftarrow \Lambda + \alpha(P^* - \frac{\#(V<\Lambda)}{\#V})^3$
3  while # of dominant voltage samples collected $\geq \kappa$ do
4      $V_H, V_L \leftarrow$ past $\kappa R$ CANH and CANL measurements
5      $v_1 \leftarrow$ the most frequently measured CANH voltages from $\kappa$ values
6      $v_2 \leftarrow$ the most frequently measured CANL voltages from $\kappa$ values
7      $v_3 \leftarrow$ UpdateDispersion($V_H, v_3, 0.75$)
8      $v_4 \leftarrow$ UpdateDispersion($V_L, v_4, 0.25$)
9      $v_5 \leftarrow$ UpdateDispersion($V_H, v_5, 0.9$)
10     $v_6 \leftarrow$ UpdateDispersion($V_L, v_6, 0.1$)
11     for $x = 1 \dots 6$ do
12         $CVD_x[k] = CVD_x[k-1] + \Delta[k](1 - \frac{v_x[k]}{v_x^*})$
13     $\Psi[k] = \sum_{x=1}^{6} CVD_x[k]$
14     $\Psi_{accum}[k] = \sum_{i=1}^{k} \Psi[i]$
15     $\Upsilon \leftarrow$ The slope of $\Psi_{accum}$ with respect to $t$ learnt by the RLS algorithm
```

Line 15) and regards it as the fingerprint of the ECU. Once the underlying IDS detects an intrusion and submits a batch of suspicious messages, Viden runs the same algorithm to compute the *suspicious voltage profile* Υ' and matches it with the voltage profiles that Viden has learnt and trusts. Whichever the closest is decided as the attacker.

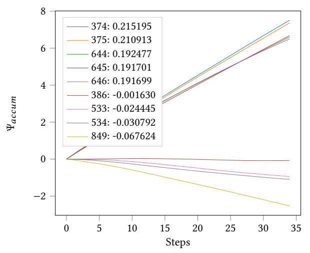
More details of Algorithm 1 are given as follows. $v^*_{\{1,3,5\}} = 3.5V$ and $v^*_{\{2,4,6\}} = 1.5V$. $CVD_{\{1\dots6\}}[0] = 0$. $\alpha = 0.5$. $\Delta[k]$ is the elapsed time since step $k-1$.

Algorithm 3: Extract upper/lower sample points from a CAN signal

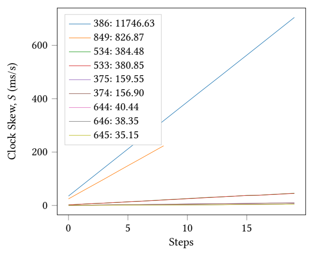
```
1 High/Low sample points of the input signal, $\boldsymbol{X}$, are extracted and stored in $\boldsymbol{Sp}$
2 $Sp_1 \subset R : \forall sp \in Sp_1 \lesseqgtr mean(X)$
3 $Sp_2 \subset R : \forall sp \in Sp_2 \lesseqgtr mean(Sp_1)$
4 $Sp_3 \subset R : \forall sp \in Sp_3 \lesseqgtr mean(Sp_2) \pm std(Sp_2)$
5 $Sp \leftarrow Sp_3$
6 $std$ represents the standard deviation of the sample points. -$std$ applied for
    upper point separation, and +$std$ applied for lower point separation.
```

Algorithm 4: Training, Testing, and Real-time Identification

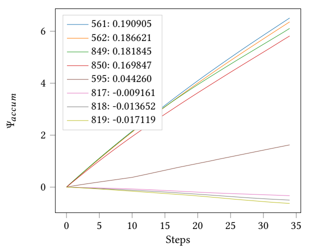
```
1 Training and testing for each ECU
2 **Training**
3 **for** *Each ECU i* **do**
4     $\mathscr{F} \leftarrow$ **goto** feature extraction Alg. 2.
5     $\mathfrak{F} \leftarrow \mathfrak{F} \cup \mathscr{F}$
1 **Testing:**
2 **for** *Each legitimate ECU i* **do**
3     $F_i \leftarrow$ **goto** feature extraction Alg. 2.
4     **for** *Each ECU j* **do**
5         $Score \leftarrow$ Mahalanobis distance of $F_i$ from $\mathfrak{F}_j$
6     $thresholds \leftarrow$ Binary search to find EERs for the scores
1 **Real-time Identification: decision making procedure**:
2 $\mathscr{F} \leftarrow$ extract features for the target CAN frame, Alg. 2
3 *MahDis* $\leftarrow$ Mahalanobis distance of $\mathscr{F}$ from the templates generated at
  **Training**
4 **if** *MahDis* < *threshold* **then**
5     **Valid** Frame
6     **Update the template of the origin of the frame**
7 **else**
8     **Malicious** Frame
```



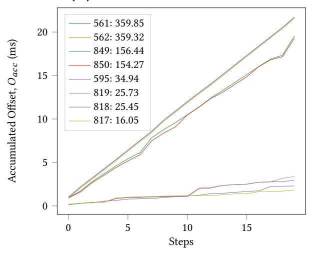
(a) Viden on Nissan



(b) CIDS on Nissan



(c) Viden on Subaru



(d) CIDS on Subaru

Figure 13: Identification results of the data from Nissan Sentra and Subaru Outback. Legends are sorted by profiles for clustering.

Algorithm 2: Feature Extraction

```
1 Generating a template feature $\mathscr{F}$ for training of an ECU
2 **for** *Each Training Frame* $i > 1$ **do**
3     Detect all low-to-high/high-to-low transitions in the line voltage
4     **goto** upper/lower level sample points Alg. 3.
5     $S_i^n \leftarrow$ the $i^{th}$ sample points after the $n^{th}$ transition
6     $F_i \leftarrow mean\{S_i^1, S_i^2, \ldots, S_i^N\}$
7     $\boldsymbol{F} \leftarrow \{F_1, F_2, \ldots, F_8\}$
8     Apply FDA to features and find W
9     $\mathscr{F} \leftarrow W \times \boldsymbol{F}$
```

When voltage samples from two ECUs are mixed, the mixed profile can be approximated as the linear combination of the two profiles under the assumption that the variants of voltage outputs from all ECUs are close. Specifically, let us suppose ECU $\mathbb{A}$'s V_H distribution follows $\mathcal{N}(\mu_{\mathbb{A}}, \sigma^2)$ and $\mathbb{B}$ follows $\mathcal{N}(\mu_{\mathbb{B}}, \sigma^2)$. In the mixed samples, r of them are from $\mathbb{A}$ and the rest are from $\mathbb{B}$, then the mixed samples' distribution is $\mathcal{N}(r\mu_{\mathbb{A}} + (1-r)\mu_{\mathbb{B}}, \sigma^2)$. The feature v_i (for $i = 1..6$) of the mixed samples can be calculated as:

$$\begin{aligned} v_i &= [0.68 - \mu]/\sigma \\ &= [0.68(1 + r - r) - (r\mu_{\mathbb{A}} + (1-r)\mu_B)]/\sigma \\ &= [r(0.68 - \mu_{\mathbb{A}}) + (1-r)(0.68 - \mu_{\mathbb{B}})]/\sigma \\ &= rv_i^{\mathbb{A}} + (1-r)v_i^{\mathbb{B}}. \end{aligned} \tag{16}$$

Similar derivations can be applied to all the other features, including v_1 and v_2 because they are basically the 50th percentiles for CANH and CANL, respectively.

Viden also uses a machine classifier based on random forest to defeat against the time-voltage-aware adversary who attempts to tune its voltage output to mimic the legitimate ECU. Since the weak adversary we assumed in the Hill-climbing attack does not attempt to do so, the machine classifier will not be triggered. As a result, we omit the details of the classifier here.

Viden's reliance on multiple messages can be seen by the fact that deriving a voltage instance needs the latest κR voltage samples, which means the minimal number of messages required is

$$\begin{aligned} n &= \frac{\text{\# of voltage samples needed } (\kappa \cdot R)}{\frac{(\text{\% of dominant bit})\times(\text{CAN frame max size})}{\text{transmission rate}} \times (\text{sample rate})} \\ &= \frac{15 \cdot 10 \text{ Samples}}{\frac{50\% \cdot 108 \text{ bit/msg}}{500 \text{ Kbps}} \cdot 50 \text{ Ksamples/sec}} \approx 28 \text{ messages.} \end{aligned} \tag{17}$$

In [9], the authors claimed 2 to 3 messages would be enough to derive a voltage instance. That is because they did not consider the $R = 10$.

B SIMPLE'S ALGORITHMS

See Algorithms 2, 3 and 4 for a better understanding of SIMPLE.

C GROUND TRUTH ESTABLISHMENT RESULTS

See Fig. 13 for the results.