Spectra: Detecting Attacks on In-Vehicle Networks through Spectral Analysis of CAN-Message Payloads

Citation for the original published paper (version of record):
Aoudi, W., Nowdehi, N., Almgren, M. et al (2021). Spectra: Detecting Attacks on In-Vehicle Networks through Spectral Analysis of CAN-Message Payloads. Proceedings of the ACM Symposium on Applied Computing: 1588-1597.
http://dx.doi.org/10.1145/3412841.3442032

Spectra: Detecting Attacks on In-Vehicle Networks through Spectral Analysis of CAN-Message Payloads

Wissam Aoudi
Chalmers University of Technology
Gothenburg, Sweden
wissam.aoudi@chalmers.se

Nasser Nowdehi
Volvo Cars Corporation
Gothenburg, Sweden
nasser.nowdehi@volvocars.com

Magnus Almgren
Chalmers University of Technology
Gothenburg, Sweden
magnus.almgren@chalmers.se

Tomas Olovsson
Chalmers University of Technology
Gothenburg, Sweden
tomas.olovsson@chalmers.se

ABSTRACT

Nowadays, vehicles have complex in-vehicle networks that have recently been shown to be increasingly vulnerable to cyber-attacks capable of taking control of the vehicles, thereby threatening the safety of the passengers. Several countermeasures have been proposed in the literature in response to the arising threats, however, hurdle requirements imposed by the industry is hindering their adoption in practice. In this paper, we propose SPECTRA, a data-driven anomaly-detection mechanism that is based on spectral analysis of CAN-message payloads. SPECTRA does not abide by the strict specifications predefined for every vehicle model and addresses key real-world deployability challenges.

ACM Reference Format:
Wissam Aoudi, Nasser Nowdehi, Magnus Almgren, and Tomas Olovsson. 2021. Spectra: Detecting Attacks on In-Vehicle Networks through Spectral Analysis of CAN-Message Payloads. In *The 36th ACM/SIGAPP Symposium on Applied Computing (SAC '21), March 22–26, 2021, Virtual Event, Republic of Korea.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3412841.3442032

1 INTRODUCTION

In-vehicle security has recently attracted notable attention as real-world attacks have demonstrated that it is possible to remotely control vehicles and compromise safety-critical functions via, for example, the Internet-enabled multimedia system, thereby threatening the safety of the passengers [6, 18, 34, 36, 37].

As early as 2010 and 2011, a group of researchers [6, 20] demonstrated two unprecedented remote attacks on vehicles. A few years later, Miller and Valasek [26] compromised a Jeep Cherokee through a vulnerability in its Internet-enabled multimedia system, allowing them to control the steering, the brakes, and the acceleration of the vehicle. Rather alarmingly, they showed how this exploit could be implemented as a worm to quickly compromise approximately 1.4 million vulnerable vehicles. The trend of cyberattacks against vehicles has continued to date with more major brands, such as Tesla and BMW, being remotely compromised [21, 22]. Although some automakers seem to have reacted to certain breaches by issuing over-the-air updates to patch exploited vulnerabilities, it was shown that the introduced security protection mechanisms can be bypassed [32].

The increasing susceptibility of vehicles to cyberattacks is in large part due to connectivity to the Internet, lack of secure network partitioning that ensures separation of safety-related domains from the rest of the network, and lack of measures to verify the integrity and authenticity of Electronic Control Unit (ECU) software and communications [7]. In addition, the communication architectures currently used in In-Vehicle Networks (IVNs) were mainly designed with no security in mind. In particular, the Controller Area Network (CAN), by far the most prevailing bus technology in IVNs, is inherently insecure and lacks the necessary means of protecting against message tampering and spoofing attacks. For instance, CAN messages are broadcast and carry no information about the sender and thus can easily be spoofed.

Most previous studies on anomaly-based attack detection leverage the high regularity of the timing behavior of IVN messages to detect malicious traffic by monitoring for unlikely changes in their periodicity. Other studies leverage the subtle, yet distinctive, differences in the physical properties of ECUs to detect intruders and identify compromised ECUs [8, 10, 19, 29]. Since a considerable portion of IVN messages are transmitted periodically, and CAN messages are inherently associated with unique low-level physical ECU properties, existing approaches are, by and large, capable of detecting attacks that cause such kinds of deviation. State-of-the-art solutions, however, fall short on two main fronts. First, there have been no noteworthy attempts to detect attacks of a more stealthy nature that do not cause drastic changes in the IVN dynamics. Among the most common types of attacks on IVNs considered in the literature (see Section 3.2), the *masquerade* attack, where the adversary injects attack messages from a compromised ECU while simultaneously muting the intended ECU, is often branded as a stealthy attack. However, in Section 3.2, we show that the masquerade attack does cause changes in the overall behavior of the CAN traffic and is hardly stealthy. This paper introduces a more stealthy attack in which the adversary *reprograms* the intended ECU and directly manipulates the payloads of its messages without affecting

SAC '21, March 22–26, 2021, Virtual Event, Republic of Korea

ACM ISBN 978-1-4503-8104-8/21/03…$15.00
https://doi.org/10.1145/3412841.3442032

the IVN characteristics. We find no clear evidence in the literature that existing techniques are capable of detecting this type of attack. Second, in most cases, prior knowledge about the underlying IVN traffic (frame ID, transmission frequency, etc.) and ECU configurations is needed. This makes the existing solutions *dependent on the underlying system specifications*, which are typically proprietary and may vary in vehicles of the same model and year produced by the same OEM, let alone in vehicles of different brands.

In this work, we introduce SPECTRA, a method for detecting attacks on vehicles by continuously monitoring CAN traffic. SPECTRA employs an exploratory time-series analysis technique to capture the *deterministic behavior* of the IVN dynamics by processing CAN traffic at the payload level. The method works by first identifying a mathematical representation of the normal behavior of IVNs and subsequently monitoring in real time for attack-indicating changes in the payload structure. In addition to being inherently capable of detecting stealthy attacks on IVNs by detecting slight variations in the monitored signal, SPECTRA requires no prior knowledge of the mechanism generating the CAN traffic. As such, the proposed technique overcomes the discussed challenges pertaining to the deployability of such solutions in practice. More specifically, this paper makes the following contributions: *(i)* We introduce a fast, lightweight, and specification-agnostic attack-detection mechanism for IVNs that goes a long way toward overcoming adoption hurdles imposed by the industry; *(ii)* We demonstrate the effectiveness of our approach by conducting extensive experiments including performing stealthy attacks that we designed to serve as real-world scenarios on a 2018 Volvo XC60; *(iii)* We show that by monitoring CAN traffic in a way that treats the entire stream of CAN message payloads as a single signal we require no comprehension of the actual encoded signals and the underlying vehicle specifications that are typically proprietary. As such, our approach is applicable to vehicles of different brands and configurations; *(iv)* We show that by identifying malicious manipulations directly at the payload level SPECTRA is capable of detecting strategic adversaries who ensure that message frequencies and low-level ECU configurations remain intact under the attack. After reviewing related literature in Section 2, we underline the vulnerability of IVNs and describe the attack scenarios in Section 3. Then, we present our detection methodology in Section 4, which we evaluate in Section 5.

2 RELATED WORK

In recent years, there have been several attempts to design and develop intrusion detection systems for IVNs [8, 16, 30, 31, 33]. Due to the long life-span (decades) of vehicles and the difficulty of maintaining regular updates, anomaly-based detection has been considered to be more viable than signature-based approaches [31]. Much of the related literature on in-vehicle attack detection lays particular emphasis on the well-defined specifications of CAN communication with respect to message periodicity and data content. Larson et al. [23] propose a specification-based method to detect malicious communication that does not comply with the configuration parameters of the ECU and the CAN protocol specifications. Müter et al. [31] propose a sensor-based detection method to detect abnormal events related to eight potentially exploitable aspects of CAN communication. An entropy-based approach is proposed in [30], where the normal entropy of CAN traffic is initially modelled, and the entropy of subsequent traffic is then monitored such that a higher value indicates a higher level of coincidence in the communication. Since IVN traffic has strict specifications, manipulation in the frequency or payload of CAN frames is expected to increase the entropy, and may thus be detected by the proposed technique. Based on the assumption that each frame ID can be associated with only one transmitter on a single bus, Matsumoto et al. [24] suggest the use of the frame ID to both detect and prevent unauthorized message transmissions. According to the authors, an ECU should verify that its own messages are not present on the bus unless the ECU is in `sending mode`. If this is not the case, then it is likely that an unauthorized node is transmitting the message on behalf of the actual sender, in which case, the receiving ECU overwrites the message before the transmission is over. To detect message-injection attacks, Moore et al. [27] propose to model the inter-arrival time (or wait time) of periodic messages. Given that in a message-injection attack the adversary must send messages with a valid ID at a rate at least equal to the original message frequency, the proposed technique detects an attack whenever the inter-arrival time between two instances of a given message deviates from the expectation. Distinctively, Murvay and Groza [29] exploit the physical characteristics of CAN frames (e.g., voltage) to fingerprint ECUs so as to achieve source detection. The authors argue that signals generated by CAN transceivers exhibit different patterns, even if the transceivers originate from the same manufacturer, due to peculiar differences in their physical properties. A follow-up study by Choi et al. [10] improves the fingerprinting capabilities by analyzing more CAN traffic features. In a similar fashion, Cho and Shin [8] propose to fingerprint CAN transceivers based on ECU clock behavior. Due to the lack of clock synchronization in CAN communication, the authors hypothesized that the `clock offset` and the `clock skew` of each CAN transceiver are inimitable because they depend only on each ECU's local crystal clock. Unlike other state-of-the-art methods, the clock-based technique has been shown capable of detecting more acute attacks. Even though the state-of-the-art IVN intrusion detection systems are capable of detecting many types of attacks, they suffer from certain drawbacks. In particular, most proposed methods require prior knowledge about the underlying IVN and ECU configurations, which may vary even in vehicles of the same model and year. Furthermore, with regards to coverage of different attacks, the proposed techniques have not been shown particularly capable of detecting attacks of a more stealthy nature.

3 ATTACKS AND VULNERABILITIES

In this section, we give a brief overview of CAN communication, introduce our adversary model, and describe the attack scenarios we used for evaluating our approach.

3.1 CAN Communication & Adversary Model

CAN is a multi-master serial bus with a typical speed of 500 kbit/s on which many of the operational and safety-critical in-vehicle functions are implemented. Each CAN *frame* has a unique ID that indicates its priority, determines its source and destination(s), and enables the receivers to decode its data content accordingly. If two or more CAN frames are to be transmitted simultaneously on a single bus that is in idle state, the frame with the highest priority (the lowest ID) is transmitted first. During this time, all other ECUs refrain from transmission, switch to `receiving` mode, and wait until the bus returns to the idle state. A single CAN frame can carry up to 8 data bytes of content, typically consisting of one or multiple signals that control different functions such as speed, steering-wheel angle, autonomous braking, indicators, and lamps. The permissible data content and transmission frequency of each CAN frame are specified by the vehicle manufacturer in the early phases of ECU development. For instance, a CAN frame with ID=0x12 and periodicity of 10 ms may belong to the engine-control ECU, in which case, it would carry data related to the engine speed and torque. A receiving ECU that is waiting for a frame with ID=0x12 examines the ID using a filtering mechanism and drops unwanted frames. Due to the strict specification of the ECU communication, the degree of randomness in IVN traffic is significantly lower than that of Internet traffic and classical computer networks.

The most common way of accessing a vehicle's internal network to inject attack messages is through its OBD-II port. It is also possible to remotely access the vehicle through other media such as the Internet, cellular networks, and Bluetooth systems [6, 26]. In our experiments, we assume that the adversary has either local or remote access to the IVN. Furthermore, we consider the worst possible situation in which a nefarious adversary aims to influence the vehicle's behavior, potentially leading to immobilization, dangerous maneuvers, or even collision. Irrespective of the attack surface used to mount the attack, after having identified the CAN ID of the intended safety-critical message, the adversary's goal is to perform one of the following malicious tasks or a combination thereof: *(i)* compromise the original sender ECU (hereinafter referred to as *target* ECU) and maliciously manipulate the message payload before it is transmitted on the bus; *(ii)* impersonate the target ECU by injecting the carefully crafted message in an indistinguishable manner; or *(iii)* prevent the target ECU from sending the message by placing it in a `listen-only mode` via diagnostic commands.

Similar to [7], we consider that ECUs can be either partially or fully compromised. If an ECU is *partially* (or weakly) compromised, the adversary lacks access to its memory and can only listen to the communication on the bus. A partially compromised ECU may not be used to inject arbitrary messages, however, the adversary can forcibly prevent it from transmitting its normal messages by either putting it offline, or by putting the transceiver into `listen-only mode` via basic diagnostic commands. ECU developers usually implement a set of service routines that may be triggered by a testing device in a workshop through a diagnostic session after the routine requirements have been met. Safety-critical routines are well protected and execute only under certain conditions, such as the vehicle being at low speed or completely turned off. Importantly, a weakly compromised ECU may not be used to trigger or exploit such routines. An adversary may essentially use a weakly compromised ECU to sniff traffic or shut it down entirely to cause one or several other ECUs that rely on its messages to malfunction. On the other hand, if an ECU is *fully* (or strongly) compromised, the adversary is assumed to have full memory access and the ability to inject crafted messages into the bus. An adversary who can fully compromise an ECU typically has comprehensive knowledge of its software and hardware specifications, and may be capable of reprogramming its firmware to add or remove ECU features. Moreover, with a fully compromised ECU, the adversary may obtain the Security Access seed/key generation algorithm by reverse-engineering the firmware to unlock the ECU and exploit its safety-critical routines. In fact, as demonstrated in recent automotive attacks [6, 20, 22, 26, 32], it is not uncommon for an adversary to unlock a strongly compromised ECU using advanced diagnostic commands and bypass (or disable) existing software protection mechanisms.

3.2 Attack Scenarios

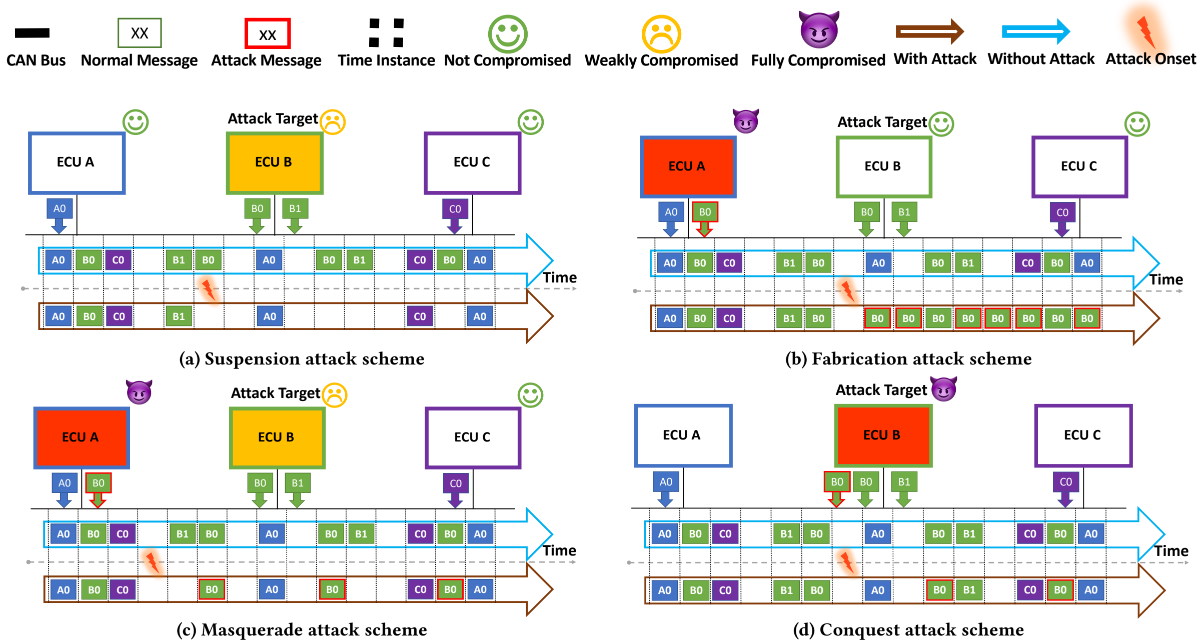
In the literature, three attack scenarios for in-vehicle networks are documented: *suspension*, *fabrication*, and *masquerade* attacks [8, 9, 25, 26], which are considered in our evaluation. In addition, we introduce and evaluate SPECTRA on an attack of a more stealthy nature, which we refer to as the *conquest attack*, to highlight the detection capabilities of our approach.

In order to provide an intelligible explanation of the attacks, let us consider a simple CAN setup (Figure 4a) in which messages A0, B0/B1, and C0 are transmitted over different time intervals by ECUs $\mathbb{A}$, $\mathbb{B}$, and $\mathbb{C}$ respectively. For the sake of illustration, we consider B0 to be a sensitive message related to one or more safety-critical functions, and thus $\mathbb{B}$ is the target ECU in the attacks.

Suspension attack. The suspension attack (Figure 1a) is a type of Denial of Service (DoS) attack, where the weakly compromised ECU $\mathbb{B}$ ceases to send messages. In order to achieve this, the adversary manages, through a diagnostic session, to put the ECU into programming mode so that it is no longer able to transmit messages. As a result, the receivers that rely on incoming data from the suspended ECU may no longer function properly.

Fabrication attack. In a fabrication attack, the adversary is incapable of compromising the target ECU, but is able to fully compromise another ECU ($\mathbb{A}$) on the bus and use it as a means to impersonate $\mathbb{B}$. Specifically, $\mathbb{A}$ is used to transmit forged messages with ID B0 at a higher frequency than $\mathbb{B}$ so that the receiving ECUs would receive conflicting B0 messages sent by both $\mathbb{A}$ and $\mathbb{B}$, except that the data received from $\mathbb{A}$ would be dominant due to higher frequency of transmission (see Fig 1b). Hence, the receiving ECUs would process the payload of the B0 messages overwhelmingly received from the adversary and react accordingly, potentially starting to malfunction. A typical scenario is the transmission of forged speedometer values at a higher frequency than the original message, resulting in the speedometer behaving erratically on the dashboard and predominantly showing the falsified values.

Masquerade attack. As can be noticed in the fabrication attack scenario, although the adversary impersonates the target ECU through a fully compromised ECU, the target ECU will keep transmitting messages. This makes the fabrication attack ineffective whenever safety-related protection mechanisms on the receiving ECUs are in effect. These mechanisms are designed to identify and react to messages with contradicting signal values [26]. In a masquerade attack (Figure 1c), on the other hand, the adversary has the additional power of weakly compromising the target ECU and suspending it from transmission, while immediately starting to transmit attack messages (B0) at the original frequency from a fully compromised ECU ($\mathbb{A}$) on the same bus. This way, the adversary can bypass possible protection mechanisms on the receiving



(a) Suspension attack scheme

(b) Fabrication attack scheme

(c) Masquerade attack scheme

(d) Conquest attack scheme

Figure 1: Schematics for *suspension* attack (1a), *fabrication* attack (1b), *masquerade* attack (1c), and *conquest* attack (1d).

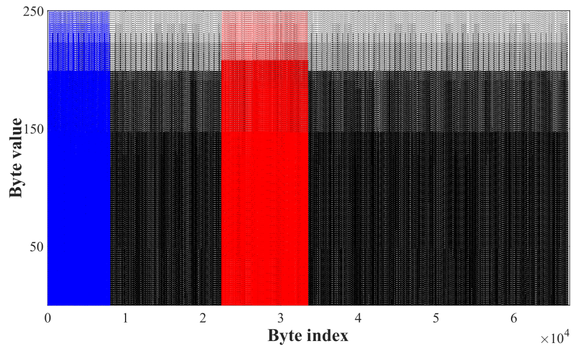
ECUs, as the latter only process B0 messages coming from $\mathbb{A}$ and no violation in the arrival times of these messages occurs. It is worth noting that since the target ECU is suspended by the adversary, it ceases to transmit *all* of its messages (and typically ECUs transmit more than a single message), including B1; hence ECUs relying on receiving message B1 may start to malfunction. A masquerade attack may have an impact on ECU $\mathbb{A}$ as well, which may become overloaded in the likely case that it has fewer transmission buffers than the number of messages it has to transmit. This in turn would introduce problems such as significant priority inversion [11] and non-abortable message transmission [12] that consequently degrade the real-time performance of the CAN bus. Thus, the masquerade attack is not particularly stealthy because it leads to more chaotic CAN communication.

Conquest attack. To highlight the full potential of our approach, we introduce the more stealthy *conquest* attack, in which the adversary is able to evade both the security protection mechanisms and the state-of-the-art intrusion detection systems. As shown in Figure 1d, in a conquest attack, the adversary directly *conquers* the *target* ECU by fully compromising it, which in none of the previous scenarios the adversary was able to achieve. In particular, the adversary is able to *reprogram* the target ECU so that instead of having to compromise another node on the network to inject the intended malicious payload, the adversary directly manipulates the payload of the sensitive message B0, *albeit only subtly*, thus forcing the corresponding safety-sensitive operation executed on the receiving ECU to function erroneously. Unlike all other scenarios we have described so far, this attack causes no changes in the normal behavior of any of the ECUs with respect to message frequency, clock offset, or clock skew behavior. Even at the payload level, a strategic adversary carrying out a conquest attack alters only a few bytes of data in a continuous stream of CAN frames. Such a stealthy attack may prove particularly effective against Advanced Driver-Assistance Systems (ADAS), such as forward collision warning, lane departure warning, and electronic stability control, that critically rely on the integrity of sensor values. Therefore, performing a conquest attack on such sensor values may have far-reaching consequences on the underlying IVN if the adversary manages to maliciously alter sensor data bytes in such a way that they fall into the normal range yet deviate from the actual values.

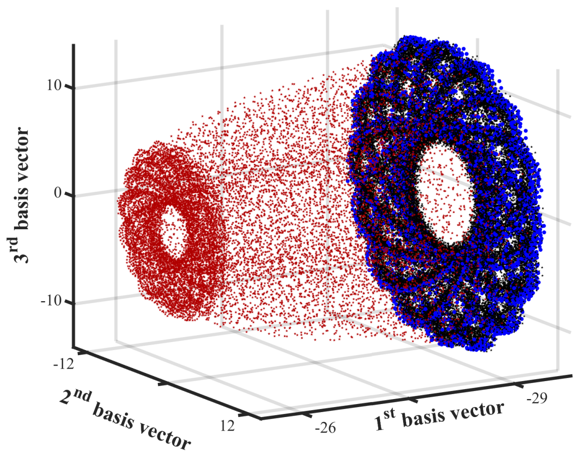
A possible real-world conquest attack scenario is one where the adversary is able to reprogram the parking assistant module and directly send crafted steering wheel angles to the steering wheel ECU causing the wheel to behave erratically. In another hypothetical scenario, the adversary reprograms the engine control module and maliciously alters data bytes in the engine speed messages in such a way that the speed sensor readings remain in a reasonable range while not reflecting the actual speed of the engine. Consequently, functions that make use of these readings to make decisions based on the engine speed will be presented with misleading information.

4 METHODOLOGY

In this section, we elucidate our attack-detection methodology. Spectra detects the types of attacks on IVNs described in Section 3.2 by monitoring CAN traffic at the *payload level* to identify *structural changes* attributable to malicious payload manipulation. The method processes the data fields in the stream of frames transmitted over the CAN bus one byte at a time. A key enabling property for spectra is the regularity of vehicular dynamics, which seem to follow a pattern with military precision. Given that this regularity is highly reflected in the CAN traffic (see Figure 2a), spectra lends itself to specification-agnostic detection of attacks on IVNs.



(a) CAN traffic



(b) Signal subspace.

Figure 2: Visualization of the departure of IVN dynamics from normal behavior during a masquerade attack.

4.1 Spectral Analysis of CAN Traffic

Spectra is rooted in *singular spectrum analysis* (SSA), a time-series analysis technique mainly used to explore different behavioral characteristics of a dynamical system purely from noisy time series of measurements [5, 14, 15, 35]. Inherently, SSA can extract essential signal information describing the deterministic behavior of the underlying system and has recently been used for anomaly detection in cyber-physical systems [1–4, 17].

Spectra takes as input a time series of CAN-message payloads and works in two phases: an offline learning phase and an online detection phase. The method is capable of detecting *slight* structural changes in the payloads by recognizing unusual byte sequences that were unseen during training, no matter whether the individual maliciously altered bytes fall into the normal range or not.

In the learning phase, an initial time series of CAN traffic used for training is embedded in a vector space, referred to as the *trajectory space*. Then, a *signal subspace* is identified through a mathematical procedure, onto which *training vectors* are projected. Owing to the regularity of IVN dynamics, the projected vectors form a *cluster* in the signal subspace, thereby defining the normal behavior. Afterwards, during an online detection phase, at every iteration, a *test vector* is composed by incorporating the most recent value. A departure score is then iteratively computed by measuring the distance between the most recent test vector and the centroid of the determined cluster. Finally, an alarm is generated whenever the score crosses a prescribed threshold.

The CAN traffic is modeled as a time series $\mathcal{T} = b_1, b_2, \cdots, b_N, \cdots$ of (the integer representation of) bytes extracted from the payloads of consecutive messages, such that if the payload of a message m_j on the CAN bus consists of bytes b_i, b_{i+1}, b_{i+2}, then b_{i+3}, b_{i+4} would belong to the following message m_{j+1} whose payload contains only 2 bytes of data. The rationale behind processing individual bytes is that the CAN data field, which may contain up to 8 bytes of data, has a variable length, yet always contains a multiple of one byte.

4.2 Learning Phase

In the learning phase, an initial subseries of $\mathcal{T}$ of length N is unfolded into a Hankel *trajectory* matrix by forming K L-lagged vectors $\mathbf{b}_i = (b_i, b_{i+1}, \cdots, b_{i+L-1})^T$ as follows

$$\mathbf{B} = \begin{bmatrix} b_1 & b_2 & \dots & b_K \\ b_2 & b_3 & \dots & b_{K+1} \\ \vdots & \vdots & \ddots & \vdots \\ b_L & b_{L+1} & \dots & b_N \end{bmatrix} \tag{1}$$

where L is called the lag parameter, $1 \le i \le K$, and $K = N - L + 1$. Then, the *singular value decomposition* of $\mathbf{B}$ is performed to obtain an orthonormal set of L eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_L$ of the covariance matrix $\mathbf{B}\mathbf{B}^T$. A matrix $\mathbf{U} = [\mathbf{u}_1 : \mathbf{u}_2 : \cdots : \mathbf{u}_r]$ is then formed, whose columns are the $r < L$ leading eigenvectors, where r is the so-called *statistical dimension*. This choice of the few leading eigenvectors plays a key role in capturing essential information about the underlying behavior of the CAN traffic while disregarding the many remaining components that correspond to noise.

The matrix $\mathbf{U}$ is a linear transformation that maps the CAN byte sequences $\mathbf{b}_i, 1 \le i \le K$ (training vectors), from the trajectory space into the lower-dimensional *signal subspace* spanned by the column vectors of $\mathbf{U}$. The transformed vectors form a cluster in the signal subspace (see Figure 2b), whose centroid is then computed as $\tilde{\mathbf{c}} = \mathbf{U}^T\mathbf{c}$, where $\mathbf{c}$ is the sample mean of the training vectors.

4.3 Detection Phase

At every iteration during the detection phase, a *departure score* is computed for the most recent lagged vector $\mathbf{b}_j$, $j > K$. This is done by computing the weighted squared Euclidean distance from the centroid, where the weights are determined by the ratio of each eigenvalue to the total sum of eigenvalues associated with the r eigenvectors determined in the learning phase. More formally,

$$\tilde{D}_j = ||\mathbf{W}(\tilde{\mathbf{c}} - \mathbf{U}^T\mathbf{b}_j)||^2 \tag{2}$$

such that $\mathbf{W}$ is an r-dimensional diagonal matrix whose i^{th} diagonal entry is defined as $e_i/(\sum_i^r e_i)$, where e_i is the eigenvalue corresponding to the i^{th} eigenvector $\mathbf{u}_i$, for $1 \le i \le r$. Finally, an alarm is generated whenever $\tilde{D}_j \ge \theta$, where θ is a prespecified threshold.

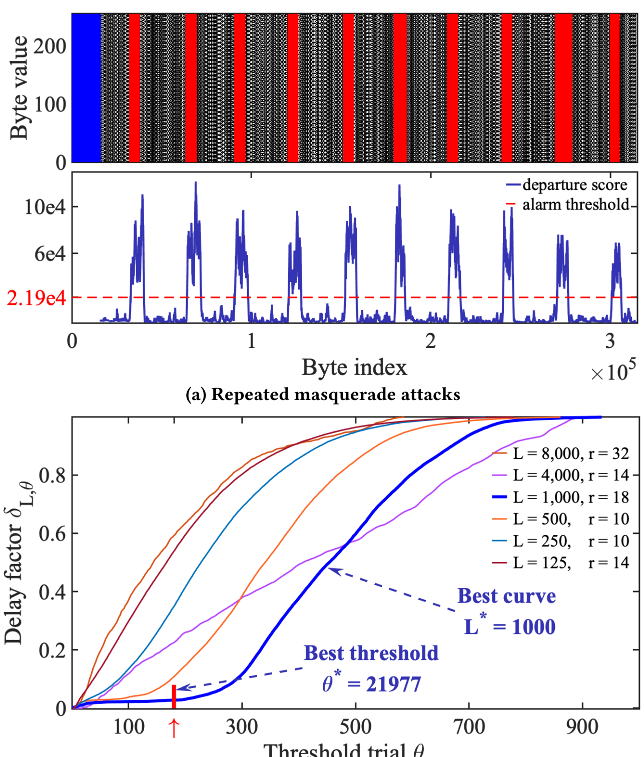
Figure 2a shows CAN traffic generated using a testbed (see Section 5.1). The initial subseries highlighted in blue corresponds to attack-free CAN traffic used in the learning phase to identify signal information that is representative of the normal behavior of the underlying vehicular communication. Once the learning phase is complete, a detection phase is initiated, in which a departure score is computed for every new traffic instance. The subseries of CAN traffic highlighted in red corresponds to malicious traffic injected by the attacker during a masquerade attack (described in Section 3.2). The signal subspace being *isomorphic* to a low-dimensional Euclidean space enables the visualization of the projected vectors in $\mathbb{R}^3$ [4]. Figure 2b displays the vectors corresponding to the CAN traffic presented in Figure 2a after they have been projected onto the identified signal subspace. Owing to the highly regular behavior of IVNs, as Figure 2b depicts, the projected training vectors occupy a firmly bounded region in the signal subspace and thereby form a cluster. Under normal conditions, the projected test vectors fall close to a cluster of training vectors. Under attack conditions, on the other hand, anomalous test vectors are forced to lie further away from the cluster and the computed distance is presumed to increase, signaling that the IVN is departing from the normal behavior.

4.4 Determining the Detection Threshold

One important metric for evaluating SPECTRA's performance is the average time it takes to detect an attack, which we refer to as the *delay factor*. Evidently, setting the alarm threshold high enough so as to avoid false positives results in a larger delay factor. More precisely, the delay factor, denoted by $\delta_{L,\theta}$, varies with the lag parameter L and is inversely proportional to the alarm threshold $\theta_{L,r}$ for a fixed choice of L and r.

We define the delay factor as the number of traffic instances (bytes) that fall into the aggregate of attack intervals T_a and whose corresponding departure scores are below the threshold, normalized by the total number of attack instances γ_a; more formally,

$$\delta_{L,\theta} = \left| \{\mathcal{T}_i \mid i \in T_a,\ \tilde{D}_i < \theta\} \right| \,/\, \gamma_a. \tag{3}$$



(a) Repeated masquerade attacks

(b) Identifying the best threshold

Figure 3: Optimizing the detection delay/false alarms trade-off.

Figure 3a shows CAN traffic from an IVN that was subject to 10 repeated masquerade attacks. To discover the *best* alarm threshold, for k different values of the lag parameter L, we determine the corresponding statistical dimension r, and compute $\delta_{L,\theta}$ for 1,000 different values of θ selected by evenly dividing the range between the minimum and maximum departure scores attained throughout the entire experiment. As illustrated in Figure 3b, this procedure yields k curves describing the trade-off between the time to detection and the likelihood of false positives for different choices of SPECTRA's lag parameter. The task then is to optimize this trade-off by determining the best delay factor δ_{L^*,θ^*}. Intuitively, the best lag parameter L^* should correspond to the curve with the minimum AuC (Area under Curve), which measures the total detection delay over the different threshold trials. After identifying the best curve as such, we determine the best threshold θ^* by visually identifying the optimal *cut* corresponding to the threshold that minimizes the delay factor while maximizing the threshold, where the latter is equivalent to minimizing the likelihood of false positives.

4.5 Limitations

Notwithstanding the importance of CAN communication, which carries most of the critical signals in modern vehicles, SPECTRA only handles CAN traffic to detect attacks on IVNs, at least as far as tested. As SPECTRA essentially performs spectral analysis of time series, the monitored messages also need to be periodic or frequently transmitted over the CAN bus, which is the case for most signals. The way SPECTRA would handle non-periodic traffic depends on the type of messages being transmitted (i.e., somewhat common event-driven messages or very rare messages). As discussed in the attack-free experiment in Section 5.6, while driving a real car, we intentionally emulated passenger-triggered controls and observed that such benign activity would be gracefully handled by SPECTRA. Furthermore, as SPECTRA monitors for changes in behavior, it may not react promptly to attacks in which the adversary manages to drive the vehicle to an unsafe state by making slow normal-looking changes at the payload level (e.g., linearly increasing engine speed) through a conquest attack.

5 EVALUATION

In order to evaluate the effectiveness of SPECTRA, we conduct a series of experiments on a *real* vehicle, a CAN bus prototype, and CAN traffic captured from two additional vehicle brands. In the first set of experiments (Exp I-VIII), we demonstrate the capability of our approach to detect all attacks described in this paper, including the stealthy conquest attack. In the second set of experiments (Exp IX, X), we examine the behavior of SPECTRA when performing under attack-free conditions. We begin with a description of the evaluation setups and the data used in validating our approach.

5.1 Evaluation Setup

CAN bus prototype. As shown in Figure 4a, we build an experimental CAN bus prototype consisting of three ECUs that communicate on a single bus. Each ECU consists of a SeeedStudio CAN-bus shield

(a) CAN bus prototype **(b) 2018 Volvo XC60** **(c) In-car OBD connection setup**

Figure 4: Different setups used for evaluation.

plugged on top of an Arduino Uno PCB. The CAN-bus shield adopts the Microchip MCP2515 SPI, a widely used CAN controller, and the Microchip MCP2551 CAN transceiver. The Arduino Uno is a microcontroller that is based on the ATmega328P with a 16 MHz quartz crystal clock, 32KB flash memory, and 2KB SRAM. ECU $\mathbb{A}$ was programmed to transmit message 0x1C every 30*ms*, and ECU $\mathbb{B}$ was programmed to transmit messages 0x01 and 0x05 every 15*ms* and 25*ms* respectively, in addition to a third ECU ($\mathbb{C}$) which was programmed to capture the CAN traffic. To produce realistic behavior, we simulate three messages with transmission frequencies and payloads (i.e., signals) that are similar to arbitrarily chosen safety-critical messages from a 2018 Volvo XC60. The CAN bus prototype was set up to operate at 500Kbps, which is the typical bit rate for in-vehicle high-speed CAN buses.

Real vehicle. In order to evaluate SPECTRA in a real-world setting and demonstrate its capability of detecting attacks on recent vehicles, we use a brand new 2018 Volvo XC60 (Figure 4b). Being a test vehicle, provided by Volvo Cars for the purpose of this research, it comes equipped with extra network interfaces, which granted us the privilege of accessing ECUs that are otherwise unreachable through the standard interfaces on a released vehicle. This privilege enabled us to perform the attacks in a much more cost-effective and time-efficient manner. It should be noted that mounting some of the attacks without such privileges requires extensive technical knowledge and would be highly costly and challenging.

To communicate with the vehicle's internal network, we chose Vector CANoe, a widely used software for ECU development and testing in the automotive industry, running on a laptop, together with a Vector VN1630A CAN interface connected to the vehicle's OBD-II port (see Figure 4c). This setup permitted us to create two virtual ECUs, namely $\mathbb{E}_1$ and $\mathbb{E}_2$, that were used for running SPECTRA and mounting attacks respectively. To evaluate the performance of SPECTRA under complex conditions in a real vehicle, we chose to monitor one of the CAN buses on the XC60 that reflects the vehicle dynamics. The highly loaded CAN bus that we monitored consists of more than five safety-critical ECUs and a gateway that enables cross-domain communication. There were more than 80 CAN messages transmitted at high frequencies between the ECUs on the bus, carrying approximately 1100 distinct signals in total. Such an environment would be a highly attractive target for attackers as it controls some major functions in the vehicle and contains messages with high integrity requirements. In this setup, $\mathbb{E}_1$ and $\mathbb{E}_2$ were practically considered as two new nodes added to the CAN bus capable of both passively monitoring the traffic in real-time and actively influencing the vehicle's internal communication by injecting crafted messages. The evaluation was performed while the vehicle was driven on a test track. However, for safety reasons, the vehicle was in parking mode when mounting an attack.

Captured CAN traffic. To investigate the performance of SPECTRA in the absence of attacks, in addition to testing the method on traffic generated by the CAN bus prototype and the real vehicle, we used CAN traffic captured from a 2012 Toyota Corolla by Mueller et al. [28] and from a 2012 Honda Civic by Diacon et al. [13]. The Toyota Corolla traffic consists of 58 distinct CAN messages periodically transmitted over different frequencies ranging from 9*ms* to 1.06 seconds. The Honda Civic traffic consists of 46 distinct CAN messages periodically transmitted over different frequencies ranging from 9*ms* to 302*ms*. In both cases, the data was captured while the subject vehicle was being driven under attack-free conditions.

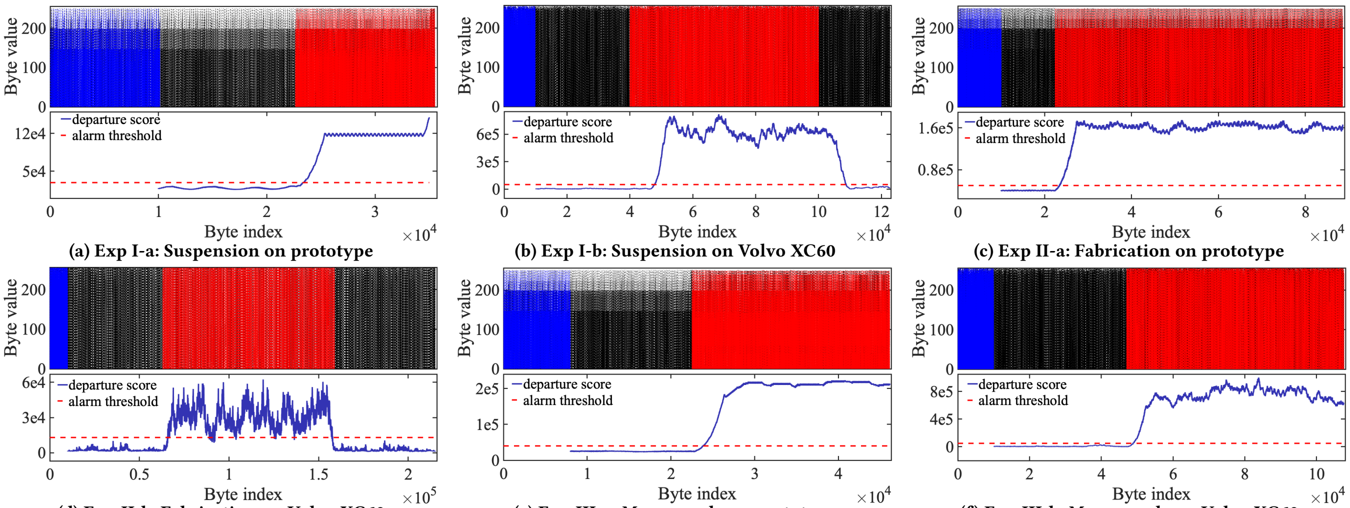
5.2 Results Overview

In all subsequent figures, the upper subplot displays the time series of raw CAN bytes. The part highlighted in blue corresponds to the subseries that was used for training SPECTRA, and the part highlighted in red corresponds to the time frame during which the attack was occurring. Figures 5 and 6 display the detection results for all four attacks. Evidently, SPECTRA successfully detects misbehaviors in the CAN traffic when the IVN is subject to all of the discussed attacks, including the stealthy conquest attack. Figure 7 shows how, with a proper choice of alarm threshold, SPECTRA triggers no false alarms during attack-free monitoring of CAN traffic generated from various sources. Next, we describe the experiments we performed. For each one of the four attack scenarios discussed in Section 3, we performed the attack on the CAN bus prototype (Figure 4a) and on a 2018 Volvo XC60 (Figure 4b).

5.3 Suspension Attack

Exp I-a: To simulate a suspension attack on the *CAN bus prototype*, we programmed ECU $\mathbb{B}$ to stop transmitting all of its messages, namely the ones with IDs 0x01 and 0x05, 20 seconds after the start of the experiment.

Exp I-b: To perform the attack on the *real vehicle*, we used ECU $\mathbb{E}_1$ to monitor the CAN traffic, while an additional OBD-II interface in the test vehicle was used to suspend a real (target) ECU on the bus by putting it into programming mode 20 seconds after establishing a diagnostic session. To observe how traffic behaves before and



(a) Exp I-a: Suspension on prototype
(b) Exp I-b: Suspension on Volvo XC60
(c) Exp II-a: Fabrication on prototype
(d) Exp II-b: Fabrication on Volvo XC60
(e) Exp III-a: Masquerade on prototype
(f) Exp III-b: Masquerade on Volvo XC60

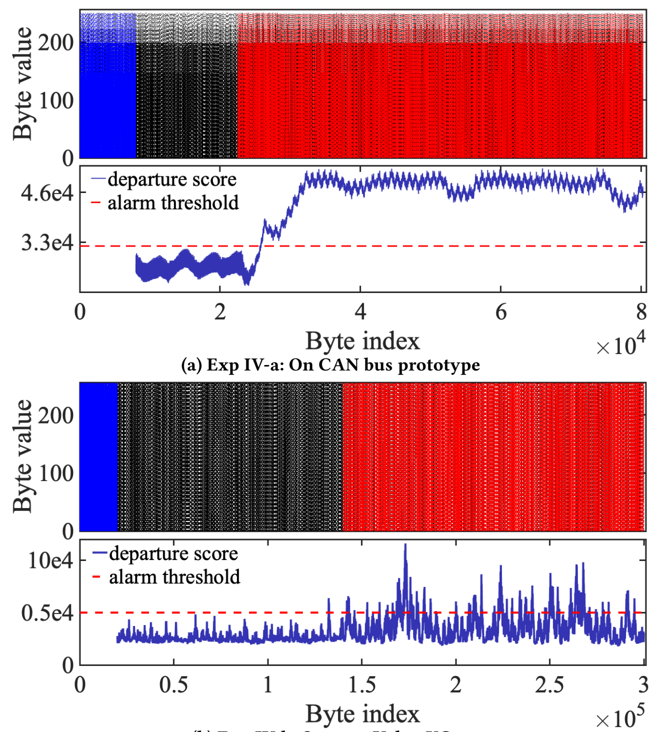
Figure 5: Detection of attacks on prototype and Volvo XC60.

after a suspension attack, we put the target ECU back online by terminating the programming session.

Discussion. As explained in Section 4, SPECTRA monitors the CAN traffic at the payload level by processing one byte of traffic at a time. Although in a suspension attack, the adversary only suspends the target ECU and does not manipulate the payloads of the CAN messages, SPECTRA still manages to detect the attack as indicated in Figures 5a and 5b. The reason why this attack is detectable by SPECTRA is that the suspension of the target ECU leads to a change in the byte sequence such that the subsequences monitored during the attack would be foreign to SPECTRA because they were unseen during the learning phase. This observation effectively implies that any change in transmission frequency is detectable by SPECTRA because it would induce a change in the payload byte sequence.

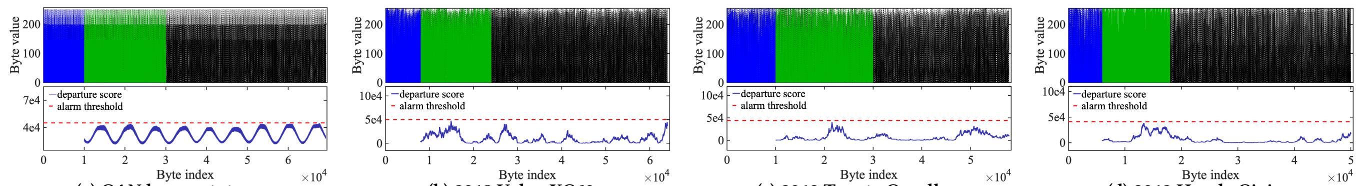
5.4 Fabrication & Masquerade Attacks

Exp II-a: To evaluate SPECTRA against fabrication attacks on the *CAN bus prototype*, we programmed ECU $\mathbb{A}$ to inject messages with ID 0x05 at the same frequency as its original transmitter $\mathbb{B}$ roughly 20 seconds after the start of the experiment. The payload of the injected message is identical to the original data sent by $\mathbb{B}$ except for the last byte, assumed to represent the target signal for the adversary. Therefore, $\mathbb{A}$ was programmed to change only the last byte of the payload when injecting forged messages with ID 0x05.
Exp II-b: To carry out the attack on the *real vehicle*, we considered a scenario in which ECU $\mathbb{E}_2$ is fully compromised by the adversary and used to mount a fabrication attack on a real ECU in the vehicle. Using the proprietary signal database, we identified the ECU that transmits Speed Limit Warning (SLW) signals in a CAN message to the gateway ECU, which subsequently forwards it to the dashboard ECU located in another network domain. Ignoring the various signals that were irrelevant to the SLW function, we located the bits in the message payload that belong to the SLW signal; specifically, the bits which trigger the warning and specify whether it shall be visual, audible, or both. Finally, we programmed $\mathbb{E}_2$ to initiate a fabrication attack on the SLW message. **Exp III-a:** To launch a masquerade attack on the *CAN bus prototype*, we programmed ECU $\mathbb{A}$, assumed to be fully compromised, to launch an attack on ECU $\mathbb{B}$, assumed to be partially compromised, in which $\mathbb{B}$ is forcibly withdrawn from transmission after 20 seconds, and immediately afterwards, $\mathbb{A}$ starts to inject messages with ID 0x05 at the original frequency on behalf of $\mathbb{B}$, but with a maliciously crafted payload.
Exp III-b: To perform a masquerade attack on the *real vehicle*, we considered a scenario in which the fully compromised ECU $\mathbb{E}_2$



(a) Exp IV-a: On CAN bus prototype
(b) Exp IV-b: On 2018 Volvo XC60

Figure 6: Exp IV: Detection of the stealthy conquest attack.



(a) CAN bus prototype (b) 2018 Volvo XC60 (c) 2012 Toyota Corolla (d) 2012 Honda Civic

Figure 7: Exp V: Evaluation of SPECTRA on attack-free CAN traffic.

launches a fabrication attack on the real ECU ($\mathbb{T}$) in the vehicle that is responsible for sending the engine rotation speed (RPM) values to the dashboard ECU and a few other receivers. For this attack scenario, we used the signal database to identify the ECU responsible for transmitting CAN messages containing RPM values (among other sensor values) to the gateway ECU, which in turn relays them to the dashboard ECU located in another network domain. We located the bits in the message payload that belong to the RPM signal and programmed $\mathbb{E}_2$ to transmit the maliciously altered RPM message, while immediately suspending $\mathbb{T}$. This was achieved by injecting forged RPM messages from $\mathbb{E}_2$ having the same ID and frequency as the original RPM message, but with forged RPM values in the payload, while placing $\mathbb{T}$ into a suspension mode by sending a reprogramming request through a diagnostic session.

Discussion. In the fabrication attack, the payloads corresponding to SLW messages are maliciously altered and, as in the suspension attack, the frequency of messages is affected since ECU $\mathbb{A}$ transmits the fabricated messages at a higher frequency. As pointed out in Section 3.2, when a fabrication attack is not feasible due to the receiving ECU having protection mechanisms in place, a *masquerade attack* can be performed instead. Technically, the masquerade attack causes similar changes in the payload byte sequence, since it involves both altering the payload of the intended message and suspending the target ECU. Figures 5c, 5e, 5d and 5f clearly show how both attacks can be detected by SPECTRA.

5.5 Conquest Attack

Exp IV-a: To simulate a stealthy conquest attack on the *CAN bus prototype*, we assumed that the target ECU $\mathbb{B}$ is fully compromised by an adversary who wants to silently manipulate specific signals in one or several messages without injecting any other messages into the bus by any other ECUs. To perform this attack, $\mathbb{B}$ was programmed to maliciously modify only the last two bytes in the payload of messages with ID 0x05 in such a way that the modified values remain within the normal range of the corresponding signal.

Exp IV-b: To perform the conquest attack on the *real vehicle*, we considered a scenario in which the target ECU $\mathbb{T}$ is fully compromised and the adversary is capable of changing certain bytes of the payload where one or more critical signals are stored, while leaving the remaining data bytes intact. In this scenario, an existing safety-critical diagnostic service (see Section 3.1) used for overwriting specific signal values on the target ECU is identified as the launching point of the attack. As this service is protected from unauthorized users by the Security Access request-response protocol, an additional OBD-II interface in the test vehicle would run the Security Access algorithm and unlock the service on the target ECU, while ECU $\mathbb{E}_1$ monitors the traffic on the CAN bus. In this attack, the signal related to fuel consumption was maliciously altered roughly 20 seconds after the start of the experiment.

Discussion. Unlike existing attacks, in a conquest attack, only a relatively small subset of payload bytes is affected, while all messages are transmitted by their original ECUs. Importantly, the adversary would inject the attack message using its *original sender* ECU and at the *original frequency*. This effectively means that the adversary is able to evade protection mechanisms, since neither the timing behavior nor the low-level physical specifications of the messages are violated. In contrast to the other attacks, the conquest attack causes minimal changes in the CAN traffic. Nonetheless, as SPECTRA is inherently capable of detecting slight variations in the monitored traffic, as shown in Figure 6, it successfully detects the conquest attack both on the CAN-bus prototype and the real vehicle. The significance of this result particularly lies in SPECTRA’s distinctive capability of detecting subtle anomalous deviations in a small fraction of signal data bytes on a heavily loaded CAN bus containing more than 1100 distinct signals, and without prior knowledge of the underlying signal specifications.

5.6 The Attack-Free Experiment

Exp V-a: To simulate normal traffic behavior on the CAN bus prototype, we programmed ECUs $\mathbb{A}$ and $\mathbb{B}$ to transmit messages 0x1C and [0x01,0x05] respectively. For the sake of producing realistic behavior, the payloads of the messages were constructed to be similar to that of existing messages on the Volvo XC60 CAN bus.

Exp V-b: In order to investigate the performance of SPECTRA under normal driving conditions, the test vehicle was driven in an urban area, where unforeseen events and ambient conditions required the driver to show different reactions (sudden acceleration, braking, etc.), thus adding normal, yet significant, variability to the IVN traffic, which was passively monitored by $\mathbb{E}_1$ throughout the experiment. Finally, in order to demonstrate the applicability of our approach to other vehicle brands, we ran SPECTRA on normal CAN traffic captured from a 2012 Toyota Corolla and a 2012 Honda Civic.

Discussion. Figure 7 shows the behavior of SPECTRA when performing on normal traffic generated by the CAN prototype and three different real vehicles. In particular, the CAN traffic displayed in Figure 7b was captured while driving the Volvo XC60 test vehicle in an urban area and, as indicated in the figure, no false alarms were triggered by SPECTRA.

6 CONCLUSION

With the rapid increase in the number of cyberattacks on vehicles, designing intrusion detection systems for CAN communication has become a major area of interest. This paper has made several noteworthy contributions to the field of automotive security. First, we have presented SPECTRA, an efficient attack-detection mechanism that is particularly suitable for the IVN domain. Second, we have demonstrated, through extensive experiments including performing attacks on a 2018 Volvo XC60 test vehicle, how SPECTRA is capable of detecting stealthy attacks on IVNs. Finally, we have shown that SPECTRA enjoys the advantage of being specification-agnostic, which makes it applicable to a wide range of vehicle models and deployable in real-world settings.

ACKNOWLEDGMENTS

This research has been supported by the Swedish Civil Contingencies Agency (MSB) through the projects "RICS", "RICS2" and "RIOT" and the Swedish Governmental Agency for Innovation Systems (Vinnova) through the projects "CyReV" (2018-05013) and "KIDSAM".

REFERENCES

[1] Magnus Almgren, Wissam Aoudi, Robert Gustafsson, Robin Krahl, and Andreas Lindhé. 2018. The Nuts and Bolts of Deploying Process-Level IDS in Industrial Control Systems. In *Proceedings of the 4th Annual Industrial Control System Security Workshop (ICSS '18)*. Association for Computing Machinery, New York, NY, USA, 17–24. https://doi.org/10.1145/3295453.3295456

[2] Wissam Aoudi and Magnus Almgren. 2020. A Scalable Specification-Agnostic Multi-Sensor Anomaly Detection System for IIoT Environments. *International Journal of Critical Infrastructure Protection* 30 (2020), 100377. https://doi.org/10.1016/j.ijcip.2020.100377

[3] Wissam Aoudi, Albin Hellqvist, Albert Overland, and Magnus Almgren. 2019. A Probe into Process-Level Attack Detection in Industrial Environments from a Side-Channel Perspective. In *Proceedings of the Fifth Annual Industrial Control System Security (ICSS) Workshop (ICSS)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/3372318.3372320

[4] Wissam Aoudi, Mikel Iturbe, and Magnus Almgren. 2018. Truth Will Out: Departure-Based Process-Level Detection of Stealthy Attacks on Control Systems. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18)*. ACM, New York, NY, USA, 817–831. https://doi.org/10.1145/3243734.3243781

[5] David S Broomhead and Gregory P King. 1986. Extracting Qualitative Dynamics from Experimental Data. *Physica D: Nonlinear Phenomena* (1986).

[6] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, and Tadayoshi Kohno. 2011. Comprehensive Experimental Analyses of Automotive Attack Surfaces. In *Proceedings of the 20th USENIX Conference on Security (SEC'11)*. USENIX Association, Berkeley, CA, USA, 6–6. http://dl.acm.org/citation.cfm?id=2028067.2028073

[7] Kyong-Tak Cho and Kang G. Shin. 2016. Fingerprinting Electronic Control Units for Vehicle Intrusion Detection. In *25th USENIX Security Symposium (USENIX Security 16)*. USENIX Association, Austin, TX, 911–927.

[8] Kyong-Tak Cho and Kang G. Shin. 2016. Fingerprinting Electronic Control Units for Vehicle Intrusion Detection. In *25th USENIX Security Symposium (USENIX Security 16)*. USENIX Association, Austin, TX, 911–927. https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/cho

[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 (CCS '17)*. ACM, New York, NY, USA, 1109–1123. https://doi.org/10.1145/3133956.3134001

[10] Wonsuk Choi, Hyo Jin Jo, Samuel Woo, Ji Young Chun, Jooyoung Park, and Dong Hoon Lee. 2018. Identifying ECUs Using Inimitable Characteristics of Signals in Controller Area Networks. *IEEE Transactions on Vehicular Technology* 67, 6 (June 2018), 4757–4770. https://doi.org/10.1109/TVT.2018.2810232

[11] Robert I Davis, Steffen Kollmann, Victor Pollex, and Frank Slomka. 2011. Controller Area Network (CAN) Schedulability Analysis with FIFO Queues. In *2011 23rd Euromicro Conference on Real-Time Systems*. IEEE, 45–56. https://doi.org/10.1109/ECRTS.2011.13

[12] Marco Di Natale, Haibo Zeng, Paolo Giusto, and Arkadeb Ghosal. 2012. *Understanding and Using the Controller Area Network Communication Protocol: Theory and Practice*. Springer Science & Business Media.

[13] Aaron Diacon, Jeremy Daily, Richard Ruth, and Carsten Mueller. 2013. Accuracy and Characteristics of 2012 Honda Event Data Recorders from Real-Time Replay of Controller Area Network (CAN) Traffic. *SAE International Journal of Transportation Safety* 1, 2 (apr 2013), 399–419. https://doi.org/10.4271/2013-01-1264

[14] Nina Golyandina, Vladimir Viktorovich Nekrutkin, and Anatoly Alexandrovich Zhigljavsky. 2001. *Analysis of Time Series Structure: SSA and Related Techniques*. Chapman & Hall/CRC.

[15] Nina Golyandina and Anatoly Zhigljavsky. 2013. *Singular Spectrum Analysis for Time Series*. Springer Science & Business Media.

[16] Min-Joo Kang and Je-Won Kang. 2016. Intrusion Detection System Using Deep Neural Network for In-Vehicle Network Security. *PLOS ONE* 11, 6 (06 2016), 1–17. https://doi.org/10.1371/journal.pone.0155781

[17] Mohammed S Kemal, Wissam Aoudi, Rasus L Olsen, Magnus Almgren, and Hans-Peter Schwefel. 2019. Model-Free Detection of Cyberattacks on Voltage Control in Distribution Grids. In *2019 15th European Dependable Computing Conference (EDCC)*. IEEE, 171–176.

[18] P. Kleberger, T. Olovsson, and E. Jonsson. 2011. Security Aspects of the In-Vehicle Network in the Connected Car. In *2011 IEEE Intelligent Vehicles Symposium (IV)*. 528–533. https://doi.org/10.1109/IVS.2011.5940525

[19] 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 (CCS '18)*. ACM, New York, NY, USA, 787–800. https://doi.org/10.1145/3243734.3243751

[20] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, and Hovav Shacham. 2010. Experimental Security Analysis of a Modern Automobile. In *2010 IEEE Symposium on Security and Privacy*. IEEE, 447–462. https://doi.org/10.1109/SP.2010.34

[21] Keen Security Lab. 2018. *Experimental Security Assessment of BMW Cars: A Summary Report*. Technical Report. https://keenlab.tencent.com/en/Experimental_Security_Assessment_of_BMW_Cars_by_KeenLab.pdf Accessed: 2018-11-15.

[22] Fred Lambert. 2017. Keen Lab Hackers Managed to Take Control of Tesla Vehicles Again. https://electrek.co/2017/07/28/tesla-hack-keen-lab/ Accessed: 2018-11-15.

[23] Ulf Larson, Dennis Nilsson, and Erland Jonsson. 2008. An approach to specification-based attack detection for in-vehicle networks. In *Intelligent Vehicles Symposium, 2008 IEEE*. IEEE, IEEE, 220–225. https://doi.org/10.1109/IVS.2008.4621263

[24] Tsutomu Matsumoto, Masato Hata, Masato Tanabe, Katsunari Yoshioka, and Kazuomi Oishi. 2012. A Method of Preventing Unauthorized Data Transmission in Controller Area Network. In *Vehicular Technology Conference (VTC Spring), 2012 IEEE 75th*. IEEE, 1–5.

[25] Charlie Miller and Chris Valasek. 2013. Adventures in Automotive Networks and Control Units. *Def Con* 21 (2013), 260–264.

[26] Charlie Miller and Chris Valasek. 2015. Remote Exploitation of an Unaltered Passenger Vehicle. *Black Hat USA* (2015).

[27] 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 (CISRC '17)*. ACM, New York, NY, USA, Article 11, 4 pages. https://doi.org/10.1145/3064814.3064816

[28] Carsten Mueller, Jeremy Daily, and Mauricio Papa. 2012. Assessing the Accuracy of Vehicle Event Data Based on CAN Messages. In *SAE 2012 World Congress & Exhibition*. SAE International. https://doi.org/10.4271/2012-01-1000

[29] 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.

[30] M. Müter and N. Asaj. 2011. Entropy-Based Anomaly Detection for In-Vehicle Networks. In *2011 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 1110–1115. https://doi.org/10.1109/IVS.2011.5940552

[31] M. Müter, A. Groll, and F. C. Freiling. 2010. A Structured Approach to Anomaly Detection for In-Vehicle Networks. In *2010 Sixth International Conference on Information Assurance and Security (IAS)*. IEEE, 92–98. https://doi.org/10.1109/ISIAS.2010.5604050

[32] Sen Nie, Ling Liu, and Yuefeng Du. 2017. Free-Fall: Hacking Tesla from Wireless to Can Bus. https://www.blackhat.com/docs/us-17/thursday/us-17-Nie-Free-Fall-Hacking-Tesla-From-Wireless-To-CAN-Bus-wp.pdf

[33] Hyun Min Song, Ha Rang Kim, and Huy Kang Kim. 2016. Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network. In *2016 international conference on information networking (ICOIN)*. IEEE, 63–68.

[34] Ivan Studnia, Vincent Nicomette, Eric Alata, Yves Deswarte, Mohamed Kaâniche, and Youssef Laarouchi. 2013. Survey on Security Threats and Protection Mechanisms in Embedded Automotive Networks. In *Dependable Systems and Networks Workshop (DSN-W), 2013 43rd Annual IEEE/IFIP Conference on*. IEEE, 1–12.

[35] Robert Vautard and Michael Ghil. 1989. Singular Spectrum Analysis in Nonlinear Dynamics, with Applications to Paleoclimatic Time Series. *Physica D: Nonlinear Phenomena* (1989).

[36] Alex Wright. 2011. Hacking Cars. *Commun. ACM* 54, 11 (2011), 18–19.

[37] Meiyuan Zhao, Jesse Walker, and Chieh-Chih Wang. 2013. Challenges and Opportunities for Securing Intelligent Transportation System. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 3, 1 (2013), 96–105.