Physical Layer Intrusion Detection and Localization on CAN bus

Pal-Stefan Murvay, Adriana Berdich and Bogdan Groza

Abstract In the light of the attacks on Controller Area Networks (CAN) recorded over the past decade, detecting intrusions has become a critical demand. While cryptographic mechanisms are largely absent on CAN buses and clever adversaries may evade intrusion detection mechanisms that rely solely on traffic analysis, using physical signal characteristics to detect the source of incoming frames started to attract a lot of interest in the recent years. This technique is based on physical imperfections of transceivers and microcontrollers as well as network characteristics that are hard if not impossible to clone. In this chapter we discuss the use of voltage fingerprints for source identification as well as the recently emerged topic of localizing controllers by means of signal propagation time. These techniques can have a number of applications ranging from forensics, the detection of unauthorized components and as a complementary mechanism to traditional cryptographic protection and intrusion detection mechanisms.

Key words: automotive security, intrusion detection, physical layer security, voltage based fingerprinting, ECU localization, machine learning

Pal-Stefan Murvay
Faculty of Automatics and Computers, Politehnica University of Timisoara, Romania, e-mail: pal-stefan.murvay@aut.upt.ro, Phone +40-256-403-242

Adriana Berdich
Faculty of Automatics and Computers, Politehnica University of Timisoara, Romania, e-mail: adriana.berdich@aut.upt.ro, Phone +40-256-403-242

Bogdan Groza
Faculty of Automatics and Computers, Politehnica University of Timisoara, Romania, e-mail: bogdan.groza@aut.upt.ro, Phone +40-256-403-242

1 Introduction

While concerns on vehicle cybersecurity were raised as early as 2004 [32], more recent demonstrations regarding security issues and their consequences coming from comprehensive security analyses of modern vehicles [17, 2, 22], led to an increased research interest in this area. Many of the identified issues come from the use of in-vehicle communication protocols which lack security mechanisms. One such protocol is the Controller Area Network (CAN) [10] which is still the most widely employed protocol that links Electronic Control Units (ECUs) even after more than three decades since its introduction. To address these issues researchers have focused on two main lines of work. A consistent body of works look at securing CAN communication by introducing cryptographic authentication or related mechanisms [8], while, more recently, many works are focusing on designing intrusion detection systems (IDS) for CAN. Reactions from the automotive industry sector and international organizations are also visible through their efforts in standardising various aspects related to vehicle cybersecurity [1, 12, 31].

As stated, the development of intrusion detection systems for CAN is a research topic that attracted considerable interest in the recent years. While many of the existing proposals adopt statistical tests and machine learning mechanisms, the various lines of work that focus on this topic generally adopt one of two approaches when it comes to sourcing data employed in the detection process. On one hand we have systems which use CAN traffic-related data (e.g., frame content, periodicity or arrival timing) that can be obtained at the application layer, from the CAN controller. Since in-vehicle CAN communication is often based on proprietary protocols which are not made public, intrusion detection systems that fall into this category generally attempt to extract meaningful behavioral data from captured CAN traces and use it to detect potential misuse [19]. Some works even go further and attempt to reverse engineer CAN frames in an attempt to extract information on signals encoded in the payload [20]. On the other hand, there are mechanisms that employ physical layer characteristics (e.g., voltage levels, propagation delays, signal rise/fall times) related to CAN communication. They rely on the well known fact that minute, uncontrollable, differences in the production process of electronic circuits introduce unique characteristics in their behavior. Therefore, this uniqueness in the signalling behavior could be used to identify transmitters.

In this chapter we focus on the former approach and discuss two approaches for intruder detection in CAN networks. The first is based on the use of timing characteristics of the CAN bus which influences signal propagation. Since detection is only the first step in thwarting potential attacks, we also cover the use of signal propagation delays for intruder localization in CAN-based networks. The second approach discussed here is based on voltage characteristics of CAN signals and makes use of machine learning algorithms to improve node identification accuracy.

The rest of this chapter is organized as follows. In Section 2 we provide some background on CAN, voltage based intrusion detection and voltage propagation delays. Then in Section 3 we discuss localization methods that use signal propagation time to localize ECUs on the bus. Section 4 contains experimental results regard-

ing ECU identification from physical layer data with the help of machine learning algorithms. Finally, Section 5 contains the conclusion of this chapter.

2 Background

In this section we provide some background on the CAN bus and its physical layer signalling. We also discuss some related works that use voltage to detect intrusions on the bus.

2.1 The CAN protocol

The CAN protocol was designed by Bosch as a solution for reliable communication for in-vehicle networks. Version 2.0 of the CAN specification [27], released in 1999, was later standardised as ISO 11898 and describes the data-link [10] and physical [11] layers which make up the CAN protocol. The data-link layer is implemented by the CAN controller, that can be used as a stand-alone chip or as a module integrated in a microcontroller (as suggested in Figure 1), and is responsible for medium access, framing and error handling. The standard CAN frame, depicted in Figure 2, can accommodate a maximum payload of 8 bytes. Larger payloads, of up to 64 bytes, can be transmitted using CAN-FD (CAN with Flexible Data-rate) a more recent extension of the original CAN protocol [10]. Each frame includes an identifier (ID) field which is usually an indicative of the frame content type or sender. While specific ID values that can be transmitted by each network node are defined at design time, the CAN protocol offers no mechanism for preventing ID misuse.

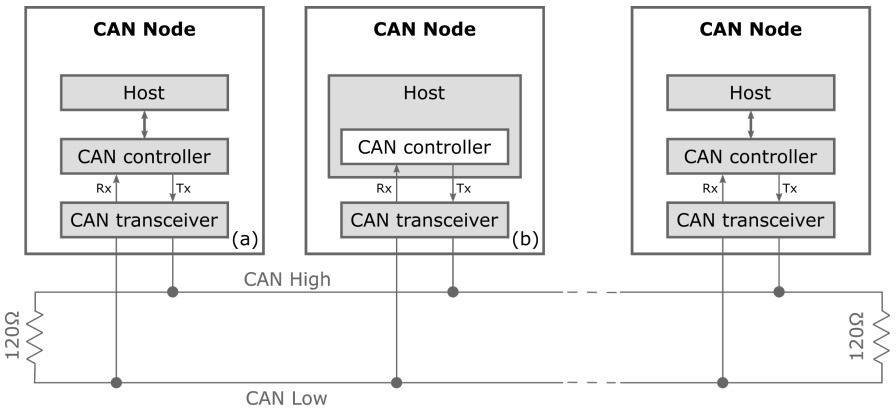


Fig. 1 CAN bus implemented with nodes using stand-alone controller chips (a) and controllers integrated in the host microcontroller (b)

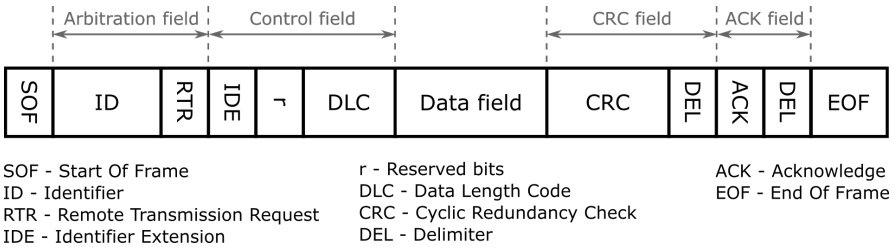


Fig. 2 Standard CAN frame format

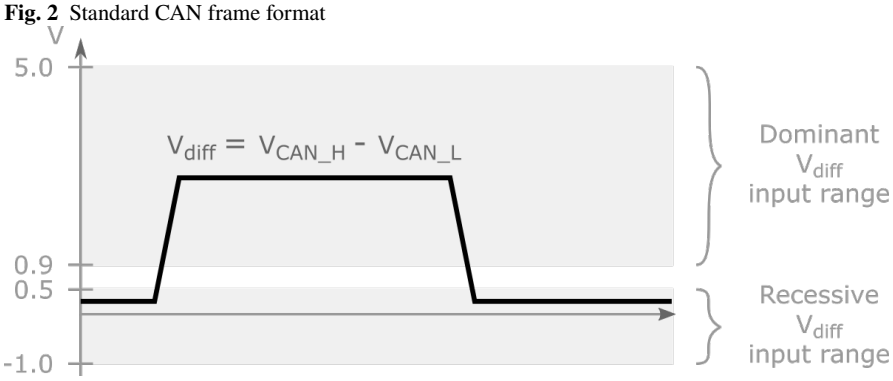


Fig. 3 Differential voltage ranges defined for the dominant and recessive states, according to ISO 11898-2

The standard high-speed CAN physical layer supports bit rates of up to 1 Mbit/s (500 kbit/s is usually used for in-vehicle communication) while its newer embodiment CAN-FD is able to deliver bit rates of up to 8 Mbit/s (the higher bit rate is only used for payload transmission). The CAN physical layer is implemented by the CAN transceiver which connects to the CAN High and CAN Low lines that form a two wire differential bus, as illustrated in Figure 1. The bus is terminated at the ends with 120 Ω resistors (matching the characteristic impedance of the bus) to suppress signal reflections. The CAN physical layer specification [11] defines ranges for the two differential voltage ($V_{diff} = V_{CAN\_H} - V_{CAN\_L}$) levels used to encode logical information as shown in Figure 3. The two logical bus states, called *dominant* and *recessive*, are used to implement a wired-AND signalling behavior. That is, a dominant state is set when at least one transceiver is actively driving the bus, while the recessive state is obtained when none of the network nodes is driving the bus. As a result, a logical "0" bit is encoded as a dominant state, while a logical "1" represents a recessive state.

While a CAN frame represents a successful transmission from a single CAN node, other nodes are allowed to actively drive the bus during the *arbitration* and *acknowledgment* fields (indicated in Figure 2). The arbitration field is dedicated to the arbitration mechanism implemented to resolve contention (i.e. the case when two or more nodes try to transmit a CAN frame at the same time). During the arbitration field, nodes competing over bus access make simultaneous bit by bit transmissions and monitor resulting values on the bus. A node stops when it transmits a recessive

bit and reads back a dominant value. Consequently, transmission priorities can be set based on the ID field, with lower values indicating higher priorities. As its name suggests, the acknowledgment (ACK) bit is used by receivers to acknowledge the successful reception of a CAN frame. Transmitters send a recessive value during this bit while all receivers are expected to transmit a dominant value if they were successful in correctly decoding the received frame.

2.2 Voltage-based intrusion detection

CAN intrusion detection mechanisms based on physical layer voltages rely on features that can be extracted from the characteristics of physical signals generated by CAN nodes. The signalling behavior of CAN nodes display unique characteristics determined by minute, uncontrollable, differences in the production process of electronic components involved (e.g., transceivers and power supply circuitry). Table 1 lists works that use various physical layer features for detecting intrusions on CAN. While some use simple threshold comparison or for matching new transmission to existing fingerprints, other works use various machine learning algorithm to achieve classification. A more recently emerging body of works, which is also discussed in the next section, i.e., [24] and [9], uses physical signal to locate ECUs on the bus.

Table 1 Comparison of existing proposals for CAN intrusion detection mechanisms based on the physical layer

Paper	**Year**	**Sampling rate**	**CAN bit rate (max)**	**Methodology**
Murvay et al. [23]	2014	2 GS/s	125 kbps	Statistical distributions
Cho et al. [4]	2017	50 kS/s	500 kbps	Machine learning
Choi et al. [5]	2018	2.5 GS/s	500 kbps	Machine learning
Choi et al. [6]	2018	2.5 GS/s	500 kbps	Machine learning
Kneib et al. [13]	2018	20 MS/s	500 kbps	Machine learning
Foruhandeh et al. [7]	2019	50 MS/s	500 kbps[1]	Statistical distributions
Rumez et al. [28]	2019	$\geq$ 2GS/s[2]	any	Statistical distributions
Kneib et al. [15]	2020	2 MS/s	500 kbps	Machine learning
Murvay et al. [24]	2020	250 MS/s	any	Threshold comparison
Groza et al. [9]	2021	250 MS/s	any	Signal slope

[1] Extracted from the associated dataset [7].
[2] Estimated based on paper details.

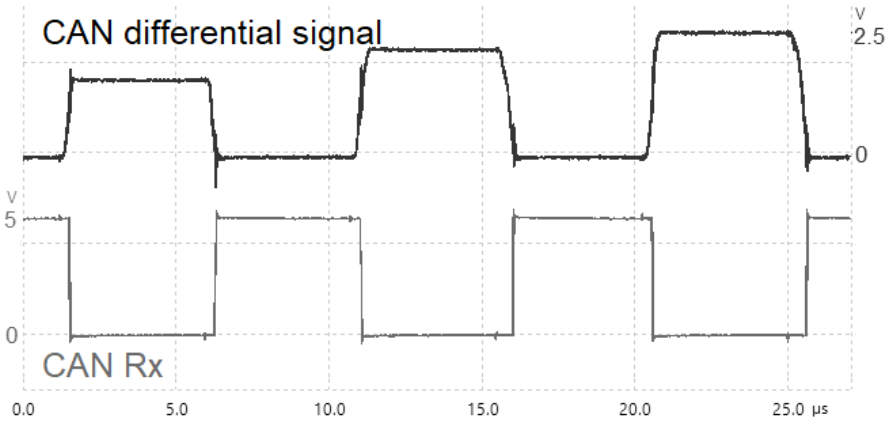
The dominant voltage level was the first among the characteristics used for uniquely identifying CAN transmitters and is still the most commonly employed. The idea was introduced in [23], which applies basic signal processing tools (i.e., mean squared error, convolution and mean-value) to extract unique sender characteristics from samples captured at the start of the arbitration field of CAN frames. The

detection accuracy of this approach is later improved by Choi et al. [5] which apply classification algorithms on a set of 17 features extracted from samples obtained during the ID field of extended CAN frames (i.e., CAN data frames that use a 29 bit ID field instead of the 11 bit found in standard CAN frames). Another line of work by Choi et al. [6] brings further improvements by considering not only the dominant level voltage for feature extraction but also the rising and falling edges generated by transitions between the recessive and dominant state as these can contain transients with a potential to reveal additional unique transmitter features.

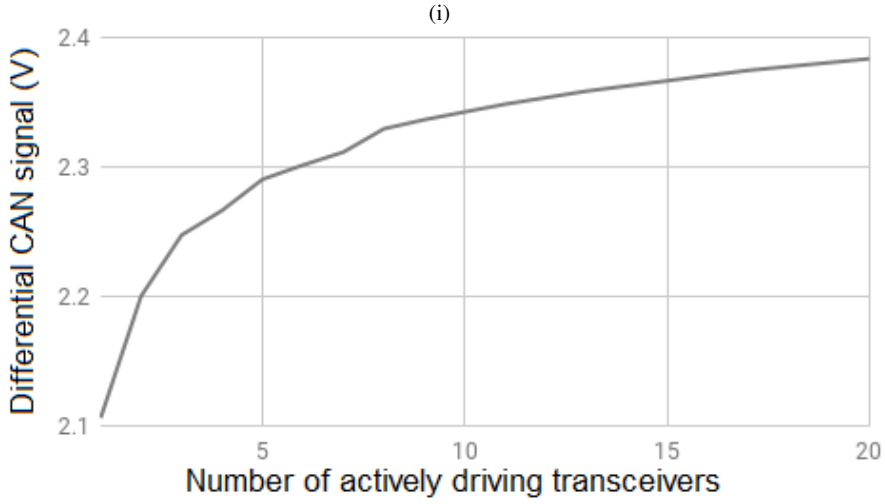
The first works on CAN physical layer intrusion detection did not consider which of the frame fields are more appropriate for sampling. As explained in the previous section, CAN signals generated during the arbitration field can be the result of more than one node actively driving the bus which affects the resulting dominant voltage levels. Figure 4 (i) illustrates three dominant bits generated by the simultaneous transmissions of up to three transceiver circuits all from MC33742 system basis chips. While the resulting dominant bus value increases with each additional transceiver driving the bus (as indicated in Figure 4 (ii)), the value is always correctly decoded by receiving transceivers (i.e., CAN Rx pin value) as long as the bus voltage levels stay within the specified ranges (Figure 3). Therefore, even under normal CAN bus usage conditions, samples acquired during the arbitration field might not be representative for the characteristics of a single node while the dominant bit in the ACK field is generated by receiver nodes. Such aspects are first considered in Viden [4], a first practical implementation of a physical layer IDS using a sampling rate of 50 kS/s which is very low in contrast with other solutions that require sampling rates in the order of Ms/s or GS/s. In Viden the ACK field thresholds are isolated when generating ID-based dominant voltage profiles, however, the arbitration field can still be used as a sampling area.

Scission [13] is the first line of work to carefully consider the frame fields to be used for reliable transmitter-related feature extraction. By using samples around the rising and falling edges of dominant bits and simple machine learning support, Scission is able to achieve better detection accuracy than previous works. Kneib et al. improve on their initial approach and propose EASI [15] reducing the sample rate requirements to 2 MS/s which should be feasible for analog-to-digital converters in common automotive-grade microcontrollers with the use of random interleaved sampling.

A completely different approach and a first step towards location based intrusion detection is presented in the work of Rumez et al. [28]. The authors use time domain reflectometry (TDR) for measuring the network response to a pulse sent by the IDS. The pulse response is an indicative of the network structure (i.e., network nodes and their location on the bus) and is compared to prerecorded reference responses to determine potential changes. This approach is able to detect when nodes are added or removed from the bus and can correlate the response signal with network node locations. On the downside, using TDR will not be effective in detecting existing network nodes that were compromised.



(i)



(ii)

Fig. 4 Influence of simultaneous transmissions on bus voltage levels: (i) Example of bus voltage and received bit values for 1 to 3 simultaneous CAN transmissions of dominant bits, (ii) Effect of the number of transceivers actively driving the bus line at the same time on dominant voltage level

2.3 Signal propagation delays

As they propagate along the bus, signals generated by CAN nodes travel through a non-ideal medium which introduces propagation delays. Sources for such delays can be found in the characteristics of the physical transmission medium as well as in local alterations of the transmission medium characteristic behavior caused by the nodes connected to the bus.

A transmission line is characterised by a specific propagation speed which is the main responsible for propagation delays. A common way of approximating the line

delay is by using the distributed model of the transmission line. As illustrated in Figure 5, transmission lines can be modeled as an infinite number of elementary line components connected in series. This is the model of a lossy transmission line in which each elementary component represents a line segment of infinitely small length with its behavior characterised by a series resistance R, a series inductance L, a parallel capacitance C and a conductance G, caused by imperfect insulation between line conductors. The values of these parameters are defined per line unit length and can be used to calculate the complex characteristic line propagation constant $\gamma(\omega) = \sqrt{(R + j\omega L)(G + j\omega C)} = \alpha(\omega) + j\beta(\omega)$. Here $\alpha(\omega)$ represents the line attenuation factor and $\beta(\omega)$ represents the propagation coefficient of the transmission line and are both dependent on frequency ($\omega = 2\pi f$). In practice, the lossless line model is used more often and it is obtained by considering that the conductance G and line resistance R are negligibly small. These assumptions can be safely made in the case of CAN lines based on the fact that G is very small in comparison to the ωC component, while R is in the order of tens of $m\Omega/m$ ($70m\Omega/m$ according to ISO 11898-2 [11] or $25m\Omega/m$ according to SAE J1939[29]). This simplifies the propagation constant, making it purely imaginary $\gamma(\omega) = j\omega\sqrt{LC}$ which, in this form, only represents the propagation coefficient. As a result, the characteristic propagation delay of the line can be calculated as $t_{pd} = \sqrt{LC}$ (s/m). According to ISO11898-2 [11] the nominal value for the propagation delay along a high speed CAN bus transmission medium is 5 ns/m (considering a homogeneous transmission medium). This offers a good approximation of propagation delay for a section of CAN bus not considering the presence of loads. However, for a better approximation of delays the loads along the bus must also be considered.

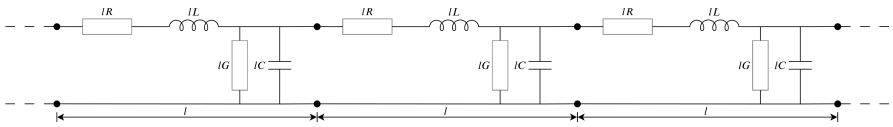


Fig. 5 Generic model of a lossy transmission line

In most cases, in transmission line models, loads (i.e., bus nodes participating in communication) are considered to be uniformly distributed along the transmission line. The resistive and capacitive loads are factored into calculations as additional distributed components per line unit length. However, when looking at in-vehicle networks, and the CAN bus in particular, there is considerable variability in the function and manufacturer of nodes sharing the same network which translates into the variability of bus interface circuitry. Moreover, bus nodes are not uniformly distributed along the line since their physical location is usually restricted to specific areas inside the vehicle.

Each CAN node connected to the bus behaves like a load connected in parallel to the bus lines. In addition, sender nodes act as voltage sources during transmission of dominant bits. The load represented by each CAN node has a resistive and a capacitive component. The resistive component mainly consists of the transceiver

differential input resistance R_{diff} which is expected to be in the 10-100 kΩ interval range according to the CAN specification. The capacitive load mainly consists of the transceiver internal differential capacitance C_{diff} which is expected to have a nominal value of 10pF [11] while the node is in the recessive state but should not exceed 50pF (measured with the node disconnected from the bus) [29]. The stub and connector used to link the node to the bus can also add to the capacitive load but this component is usually negligible. With this in mind, we can define the equivalent model of a loaded CAN bus as illustrated in Figure 6, where R_T are the bus termination resistors, each bus line segment connecting nodes is represented as a lossless transmission line component and nodes are represented as a parallel RC load, with an additional voltage source added as a component of a transmitting node.

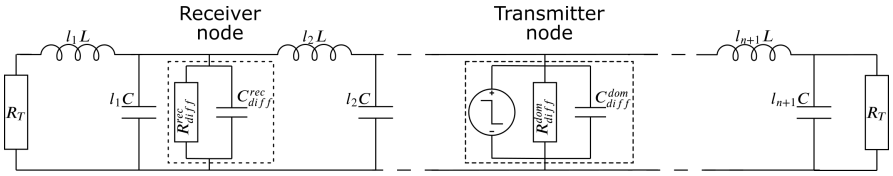


Fig. 6 Equivalent model of a CAN bus with receiver and transmitter nodes

To estimate the propagation delay based on this model, the work presented in [24] considers the loads caused by CAN network nodes to be mainly capacitive due to the reduced effect of the resistive load on propagation delays. Based on this assumption, the propagation delay is estimated as the sum of delays for each bus segment, where a bus segment is considered to span the distance between two nodes or between a node and the bus end. The differential capacitance of the node included in each segment is considered to be distributed along that line segment and factored in the calculations along with the characteristic line capacitance. Therefore, the estimated propagation delay on a segment of CAN bus can be calculated as

$$t_{pd} = \sum_{i=1}^{n} l_i \sqrt{L(C + C_{diff_i}/l_i)} \tag{1}$$

, where L and C are the characteristic line inductance and capacitance, C_{diff_i} is the differential capacitance of the CAN node included in segment i and l_i is the length of the ith segment.

3 Localization Methods based on Physical Layer Signals

In this section we discuss two intrusion detection mechanisms based on the differential propagation delays of the signals recorded at the physical layer which can be used to estimate the location of the transmitter node.

3.1 Transmitter identification by propagation delays

Based on the loaded CAN bus model discussed in the previous section it is evident that the propagation delays of CAN signals, as viewed from a fixed observation point, are directly influenced by the transmitter location on the bus. This suggests that the propagation delay of CAN signals could be used to identify transmitters and estimate their location on the bus. However, the problem comes down to how can these propagation delays be recorded. Using a single, fixed, observation point on the bus for measuring propagation delays would require additional information regarding the actual transmission time of the message which can only be recorded at the transmitter node location. To obtain this information from sender nodes, the receiver node, in charge with delay measurements, needs to be synchronized with the transmitters and trust the timing information they provide. Moreover, there should also be a way to determine if the transmission came from the left or right-hand side of the bus relative to the receiver location.

To alleviate these problems, the authors of [24] and [9] proposed a novel intrusion detection mechanism based on signal propagation delays that does not require knowledge about the message transmission time which also eliminates the need for time synchronization between nodes. They achieve this by measuring the differential propagation time, that is, the difference between the time required for a signal to reach one end of the bus and the time required for it to reach the other end. This requires monitoring CAN signals at the two ends of the CAN bus (considering the network is based on a bus topology) and recording signal arrival time at each end, as suggested in Figure 7. The differential propagation time can then be computed as $\delta = t^{N_i}_{right} - t^{N_i}_{left}, i = 1, n$, where $t^{N_i}_{right}$ and $t^{N_i}_{left}$ are the arrival times of the signal generated by node N_i at the right and left end of the bus respectively. This is equivalent to measuring the propagation time on the two signal propagation paths relative to the transmission point and calculating $\delta = t_{pd_{right}} - t_{pd_{left}}$.

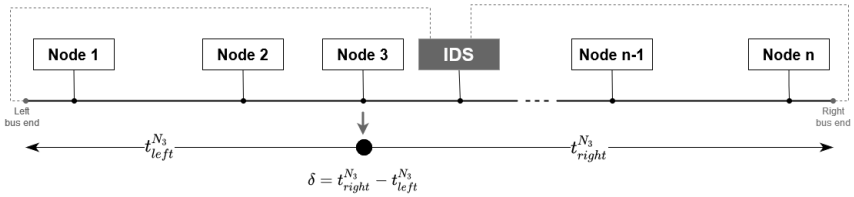


Fig. 7 Concept for recording differential propagation delays on a CAN bus

Like propagation delays, the differential propagation time is directly influenced by the location of the transmitter node on the bus. The absolute value of δ increases as the transmission node location is farther away from the point which represents the bus center of mass with respect to propagation delay. The sign of the differential propagation time indicates the bus end closer to the node location (i.e., a negative

value suggests a transmitter closer to the right bus end while a positive value indicates a transmitter closer to the left bus end).

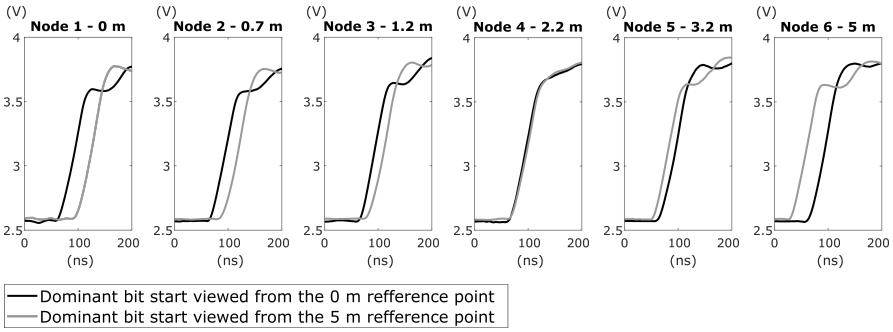


Fig. 8 Difference in dominant bit arrival time at CAN bus ends depending on transmitter location

The relation between differential propagation time and transmitters' location on the bus is illustrated by Figure 8 which shows dominant bit arrival times recorded on the CAN-High line at the two bus ends of a 5 meter long transmission line having 10 nodes distributed unevenly along the CAN bus. As expected the difference in signal arrival time is close to zero for transmitter located toward the center of the bus and it increases as senders are located closer to either end of the bus.

3.2 Signal acquisition

For intrusion detection based on voltage-related signal characteristics, samples must be recorded via a direct connection to the CAN bus physical layer. Since CAN uses differential signalling for increased noise immunity, it would be preferable to sample both the CAN-High and CAN-Low lines and use the resulting differential signal. Existing physical layer CAN IDS proposals use either the differential CAN signal for sampling or only one of the two CAN bus lines.

As discussed in the previous section an intrusion detection mechanism based on differential propagation delays requires the ability to sample physical layer signals as they are seen at the bus ends. This involves connecting the IDS sampling circuitry to the physical bus lines at both of the bus ends. Sampling the differential CAN signal would require a two wire connection to each bus end which increases wiring complexity. TIDAL-CAN [24] and CAN-SQUARE [9] proved to be efficient in extracting the differential propagation delay using a single CAN wire connection at each bus end (either CAN-High or CAN-Low). An alternative would be to use extra circuitry that connects to the two bus lines and outputs the differential signal. For example, PLI-TDC [25], a time-based physical layer intrusion detection mechanism, uses samples from the Rx pin of a transceiver connected to the bus for time measurements instead of directly sampling the bus lines. This approach would only

be appropriate for timing based intrusion detection mechanisms since physical layer voltage characteristics are not transmitted through the transceiver.

Many of the more recent voltage-based IDSs use rising and falling edges as a main target area for sampling due to the presence of transients that may reveal more unique features. The use of rising or falling edges is also required for measuring differences in signal arrival times at the bus ends since they provide clear indication for the start of a bit. Therefore, the signal areas targeted by sampling should be recessive-to-dominant and dominant-to-recessive transitions. Moreover, sampled signals must be the result of a single target node actively driving the bus. As discussed in previous sections, it is expected to have multiple nodes transmitting during the *arbitration* and *acknowledge* fields of a CAN frame. Therefore, the CAN frame areas targeted for sampling should be the *control*, *data* and *CRC* fields. In addition to the physical layer signal, the IDS also needs to capture the actual frame content. This is required for extracting information needed to identify the expected frame sender which is compared against the actual transmitter as inferred by the IDS.

The datasets used in the intrusion detection mechanisms discussed in this section were obtained from experimental models of a CAN bus. A PicoScope device from the 5000 series, along with the associated PC application, were used for sample acquisition.

3.3 The TIDAL-CAN methodology

TIDAL-CAN [24] introduces the concept of using differential propagation delays for intrusion detection and transmitter location estimation. By using the TIDAL-CAN mechanism it is possible to detect and distinguish between various attack strategies, i.e., compromised nodes, replaced nodes and node insertion. Sender location estimation is possible with an accuracy of several tens of centimeters, depending on the attack strategy employed. The CAN-TIDAL methodology is evaluated on an experimental setup comprising a 5 meter CAN bus with 10 nodes as illustrated in Figure 9 represented by up to 5 different device types.

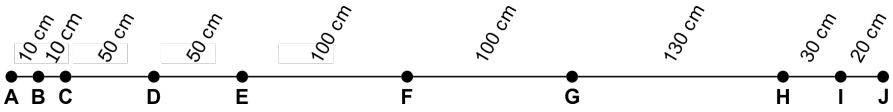


Fig. 9 Node positioning along the experimental bus model employed for evaluating TIDAL-CAN and CAN-SQUARE

Differential delays are measured from the rising/falling edges of the signals captured at the ends of the bus. Since the shapes of the rising/falling edges are not ideal, a threshold is used to specify the voltage level at which the differential delay δ should be measured. A common threshold is established for all transmitters so that it assures the best separation accuracy of differential delays from known network

nodes. The differential delays of known network nodes are prerecorded in a training phase and associated with frame IDs which are usually uniquely assigned to specific senders. Data recorded in the training phase is then used during normal run-time to identify transmitter nodes. Failure to correlate a newly recorded differential delay with prerecorded values expected for a specific frame triggers an intrusion alarm. This will happen when a compromised node attempts to transmit a frame associated with a different node as well as when the network structure is altered by removing or inserting a node since this will alter the characteristic propagation delay behavior of the network which is reflected in the propagation delays of other networks nodes.

Sender location estimation is made based on differential delays using simple linear interpolation considering two nodes with known locations. While identifying transmitter location is straightforward for attacks that do not lead to the alteration of the network, attacks involving node replacement or node insertion pose more challenges. This is caused by the fact that altering the network structure results in changes in the characteristic propagation behavior of signals sent along the bus. As a result, differential delays, including those produced by legit nodes, will be affected. Therefore, locating transmitters in this case cannot rely on prerecorded fingerprints. Node legitimacy must be reassessed at the bus level in order to allow sender location estimation and this requires processing multiple transmissions from all network nodes.

3.4 The CAN-SQUARE methodology

CAN-SQUARE [9] proposes a simple algorithm that improves on the threshold based separation of differential delays that is presented in TIDAL-CAN [24]. The separation accuracy is in the range of 10 centimeters which proves good localization accuracy while requiring only elementary arithmetic operations, i.e., additions and multiplications of the sampled amplitudes. What is notable about the methodology is that it proves resilience against replacement and insertion attacks as well as against temperature variations in the range of $0 - 60°C$. Previous works on physical layer identification of in-vehicle ECUs had a hard time with environmental variations under which the fingerprint of each ECU drastically changes. Apparently, the propagation time of the signal is far less influenced by environmental changes and localization is still sufficiently accurate even when the geometry of the cable changes as well when the temperature changes. The authors from [9] present experiments with the cable heated in enclosed box or cooled down in a fridge using four temperature check points $0°C$, $24°C$, $50°C$ and $60°C$.

How the methodology works is quite easy to explain. Two sampling points v_i, v_{i+w} are selected, separated by a window of size w, and using the sampling period δ multiplied by the size of the window w, the slope of the line that leads through the two sampling points is extracted, i.e.,

$$s[i] = \frac{v[i+w] - v[i]}{w\delta} \tag{2}$$

When the slope exceeds a fixed threshold τ the point is marked as the start time of the bit. The start time of the bit is computed both to the left and right side of the bus allowing to extract the exact location of the ECU from which the bit originates as:

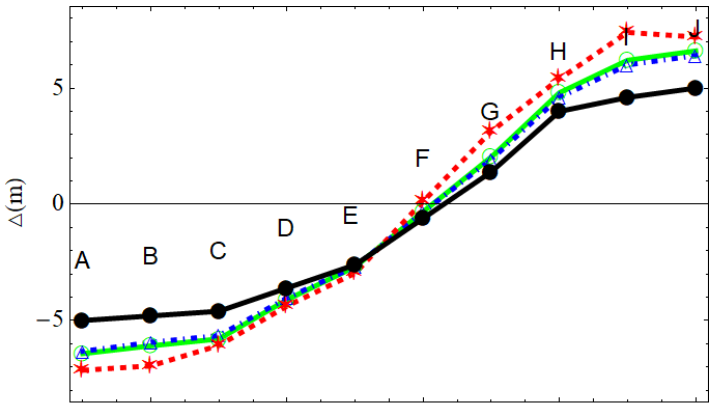
$$\pi = \frac{(\lambda_l - \lambda_r)\delta}{5 \times 10^{-9}} \tag{3}$$

Here λ_l and λ_r are the recorded indexes of the sample when the angle s reaches threshold τ at the left and right sides of the bus respectively, δ is the sampling time while 5×10^{-9} is a constant representing the default propagation time of the signal (representing the nominal propagation speed on a CAN bus which is 5 ns/m).

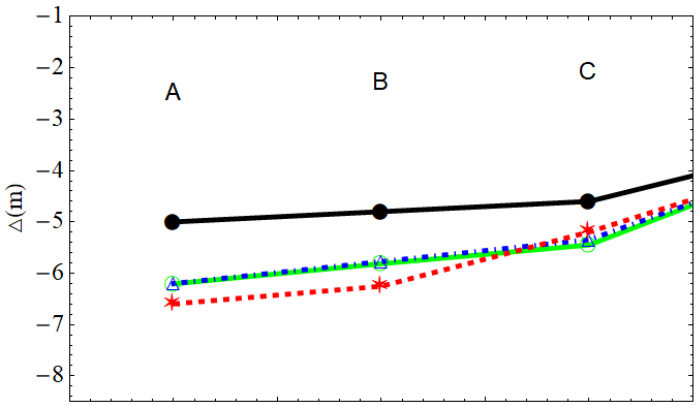
The paper presents two algorithms, the forward-square and backward-square algorithms, which parse the signal from the left-to-right or right-to-left respectively [9]. The backward algorithm gives slightly better results than the forward algorithm. As a suggestive depiction for the accuracy, we graphically show localization results in Figure 10 for the case of ECU replacements in the 10 ECU network configuration that is also used in CAN-TIDAL [24]. When replacements are done with identical ECUs (dashed-dotted line marked with triangles in Figure 10) the determined distances by the BCQ-SQUARE method generally fluctuate under 1 decimeter precision. Due to impedance changes, when replacements are done with distinct ECUs (dotted line marked with stars in Figure 10) the determined distances by the BCQ-SQUARE method may deviate by 2-3 decimeters but the nodes are still easy to locate. Figures 10s (ii) and (iii) give a detailed view of the distances at left and right sides of the bus. Notably, in case of replacements with distinct ECUs, only a single node out of the 10 nodes, i.e., ECU I, appears to be further than its real position, the rest of the 9 ECUs come close to their original location. The black line marked with filled circles denotes the actual physical position of the ECUs. We also note that in case of replacements with distinct ECUs, 6 out of the 10 ECUs were replaced with distinct ECUs, which is quite an extreme case and not an usual situation for a real in-vehicle network. For more results, in case of ECU insertions as well as temperature changes, we refer the reader to the original work [9].

4 Machine Learning on Physical Layer Signals

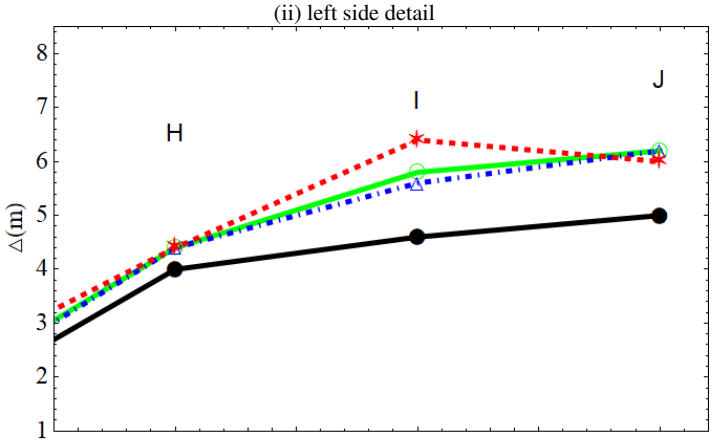
In this section we present concrete experimental results on ECU identification by using physical layer data. The results show that using single voltage features, like the maximum or minimum voltages, may help for smaller pools of ECUs but are insufficient as the pool becomes larger. In this case traditional machine learning algorithms give better results but only neural networks seem to separate between samples with excellent accuracy.



(i) complete view



(ii) left side detail



(iii) right side detail

Fig. 10 Localization with BCW-SQUARE of the 10 EUCs in case of replacements with identical ECUs (blue line) and distinct ECUs (red line): (i) complete view, (ii) left side, (iii) right side

4.1 The ECUPrint Dataset

A recently published work has released in the public domain a comprehensive physical layer dataset, ECUPrint [26] that contains data collected from 10 vehicles, i.e., nine passenger vehicles and one heavy-duty vehicle compliant to the J1939 standard. It is also relevant to note that the ECUPrint paper [26] advocates the use of physical fingerprints for forensics purposes, i.e., identification of vehicles that may be subject to theft or VIN cloning or the illegal replacement/modification of in-vehicle ECUs, an application of physical fingerprints which has not been previously considered. The authors propose the use of four features: 1) mean voltage, 2) maximum voltage, 3) bit time and 4) plateau time, each of them being extracted from isolated bits, i.e., a dominant bit between two recessive bits. The paper acknowledges that the use of a single feature out of the four leads to great overlaps between ECUs and multiple features should be combined. The use of all four features results in a sufficiently good identification of the ECUs with only slight overlaps. As we will show later, the use of machine learning algorithms allow an identification with a very high accuracy, above 99.9%.

Each car from the dataset has between 3 and 9 ECUs and each ECU uses distinct IDs. For each ID several bits are extracted leading to between 20 and 20187 sample files for each ECU in the dataset. One measurement represents 2000 sampling points for the nine passenger cars and 2700 sampling points in case of the measurements from the heavy-duty vehicle. This is due to the distinct data-rate of the bus from the heavy-duty vehicle. In this work we will use only measurements from the nine passenger vehicles which contain a total of 51 ECUs. The number of measurements is not equal for each ECU, but for each measurement there are exactly 2000 sampling points which allow us to use the same architecture for the classification algorithms.

4.2 Results with Traditional Classifiers and Neural Networks

We evaluate the detection performance based on the ECUPrint dataset for 5 machine learning algorithms, i.e., Decision Trees (Tree), Linear Discriminant (LD), K-Nearest Neighbors (KNN), Support Vector Machines (SVM) and a simple neural network (NN) available in the Matlab toolset [21]. A few words on these algorithms may be in order. Here is a short description based on the Matlab documentation [21]:

- *Decision Trees (Tree).* Decision tree is a supervised classification algorithm which organizes data as a tree to provide fast and easy to visualize classification results.
- *Linear Discriminant (LD).* Discriminant analysis is a classification algorithm based on the Gaussian distribution. The linear discriminant creates linear boundaries between classes.

- *K-Nearest Neighbors (KNN).* KNN is a commonly used classifier based on distances (in our case Euclidean distances) between the training samples and the test samples.
- *Support Vector Machines (SVM).* Support Vector Machines is used to train binary or multiclass models. SVM is a supervised machine learning algorithm commonly used to solve distinct classification problems. Matlab offers support for 6 types of SVM classifiers: Linear SVM, Cubic SVM, Quadratic SVM, Fine Gaussian SVM, Coarse Gaussian SVM and Medium Gaussian SVM. In this work we use the Fine Gaussian SVM classifier, which uses a Gaussian kernel function.
- *Neural Network (NN).* The Neural Network (NN) that we use is a simple wide neural network available in Matlab. It contains an input layer, one fully connected layer with 100 neurons, a rectified linear unit (ReLU) a final fully connected layer with 51 outputs (corresponding to the 51 ECUs used in our evaluation) and a Softmax function.

To evaluate the performance of the employed classifiers we used the following metrics, which derive from the true positives *TP*, false negatives *FN*, true negatives *TN* and the false positives *FP* rates:

1. *Accuracy* which is computed using the *kfoldLoss* classification error using k-fold cross validation (for our dataset we used 5-fold cross validation)

$$Accuracy = 1 - kfoldLoss,$$

2. *False Acceptance Rate (FAR)* and *False Rejection Rate (FRR)* which give a better understanding of the success and failure rate in identifying a node and are computed as:

$$FAR = \frac{FP}{TN + FP}, \qquad FRR = \frac{FN}{TP + FN},$$

3. The traditionally used *Precision*, *Recall* and *F1-score* computed as

$$Precision = \frac{TP}{TP + FP}, \qquad Recall = TPR = \frac{TP}{TP + FN},$$
$$F1 - score = \frac{2 \times precision \times recall}{precision + recall}.$$

We evaluated the performance of the 5 machine learning algorithms mentioned above using various percentages of the dataset for training and testing. We employ the implementation of these classifiers provided in Matlab 2021a. The tests were performed on a laptop equipped with an Intel Core i7-9850H processor and 32Gb RAM.

To begin with, we show that the use of two voltage features, i.e., two sampling points from a single bit, is insufficient when using machine learning classifiers. The authors in [26] already argued that four features (the mean voltage, maximum voltage, bit time and plateau time) would be required for such separation. By using 2 voltage

features, the maximum and minimum voltage, the KNN classifiers failed at least half of the times when identifying a node, leading to an accuracy of only 48.21%, mean values for FAR, FRR, precision and recall were 1.06%, 57.29%, 42.71% and 45.75% respectively, when using 80% of the data for training. In Figure 11 we depict the confusion matrix for KNN when 80% of the data is used for training and 20% used for testing in the case of using 2 features, i.e., maximum and minimum voltage. The true and the predicted class axis represent the ECU classes, with the letter indicating a distinct vehicle and the number denoting a particular ECU inside a vehicle, e.g., A1 is ECU1 from car A, B2 is ECU2 for car B, etc. The correctly identified ECUs which are marked on the main diagonal and misidentifications are highlighted outside the main diagonal. Locally, inside a single car, the ECUs may be correctly identified but there are clear overlaps between ECUs from different vehicles from the pool of 51 ECUs. The KNN classifier gave better results when compared to the rest of the classifiers on these two features alone. Still, two voltage features are insufficient for separation.

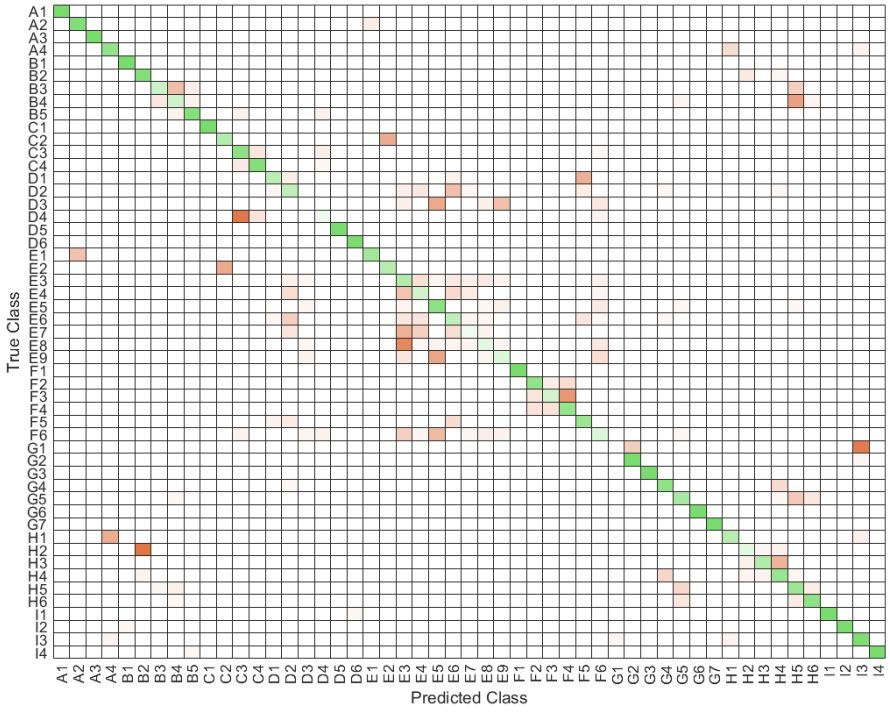


Fig. 11 Confusion matrix for KNN at 80% training on 2 features (max and min value) for the 51 ECUs

We now extend the classifiers over all the 2000 sampling points for each bit extracted from the ECUPrint dataset [26]. Firstly we split the dataset from each ECU

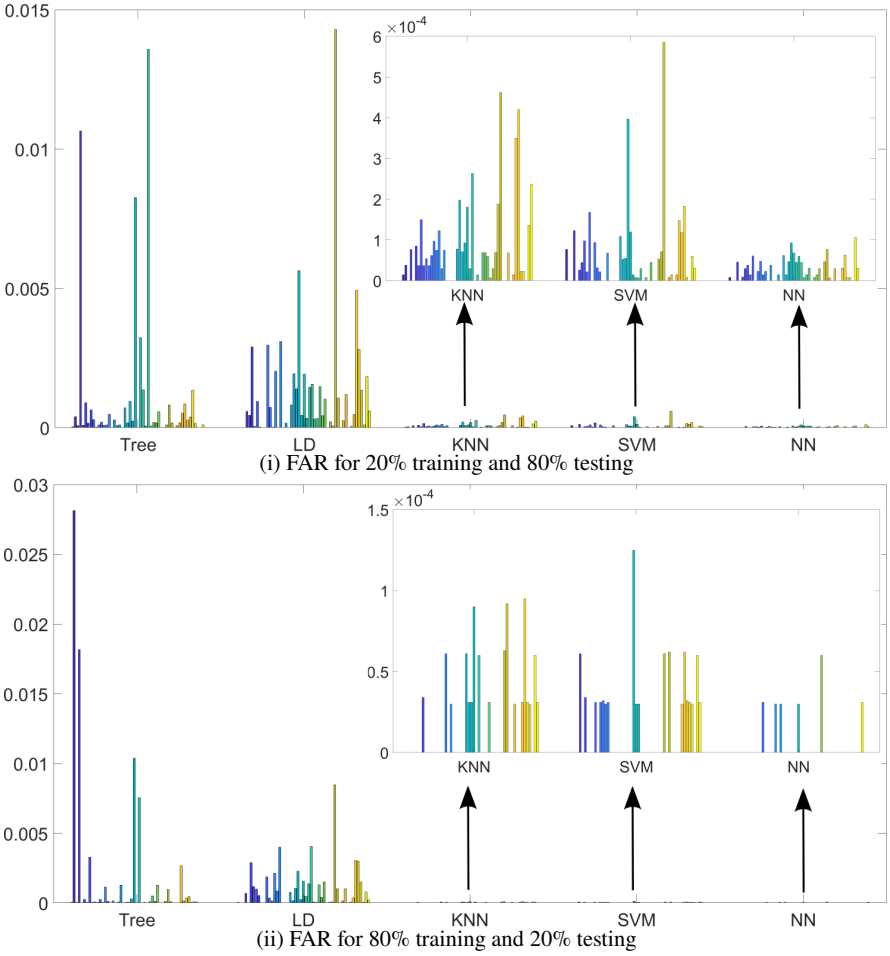


Fig. 12 FAR for 20% and 80% training for 51 ECUs

randomly in 20% training data and 80% as testing data and then increase the training percentages up to 80% in steps of 20%. The bar charts shown in Figure 12 depict the FAR obtained when using 20% (i) and 80% (ii) as training data for each of the 51 ECUs while applying all 5 machine learning algorithms selected for evaluation. When using 20% of the dataset as training data, the Tree and LD classifiers show poor performance, with the FAR reaching 1.3% in case of Tree and 1.4% in case of LD. The KNN, SVM and NN exhibit better results. For KNN, the FAR is up to 0.046% while in case of SVM, the FAR goes up to 0.05%. In the case of NN the results are slightly better in terms of FAR with the values falling in the 0 to 0.01% range. When increasing the training data percentage to 80% the FAR values increase for two ECUs in the case of the Tree classifier reaching 0.28% for one of them.

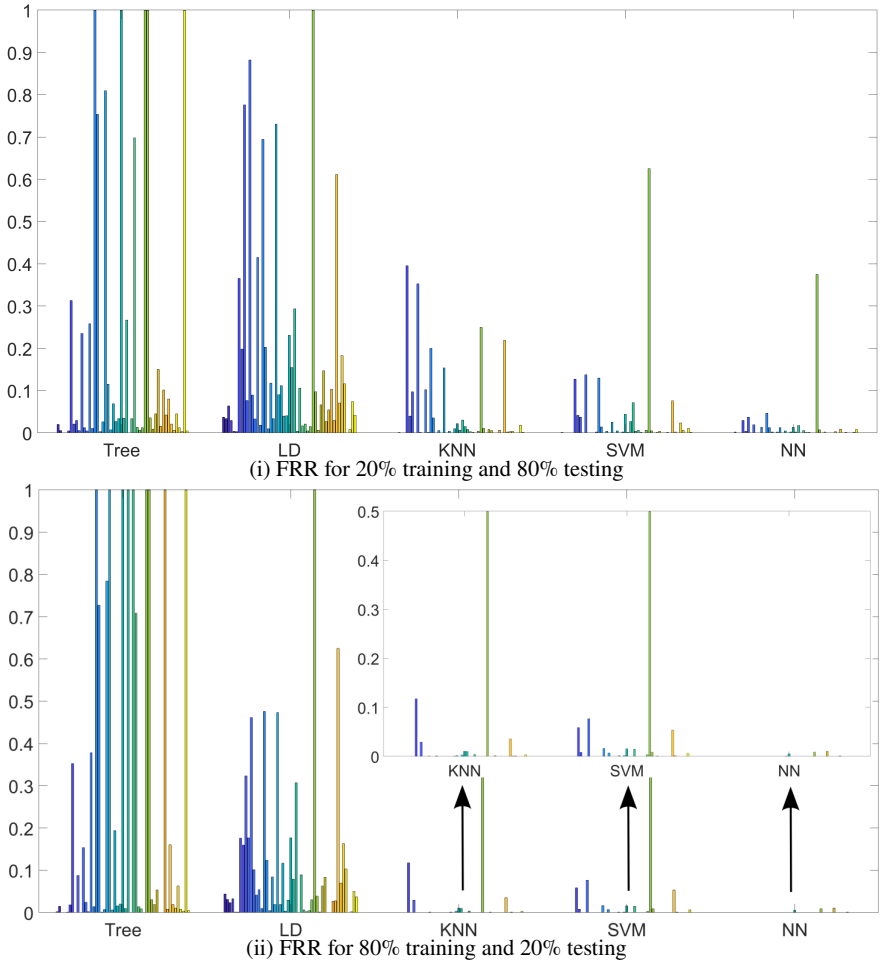
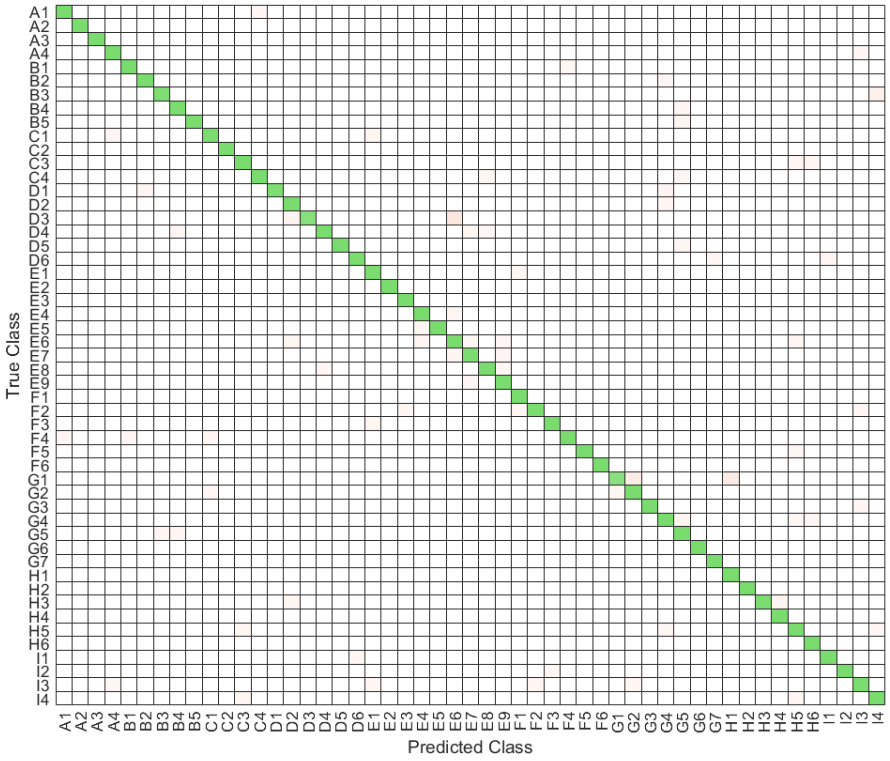


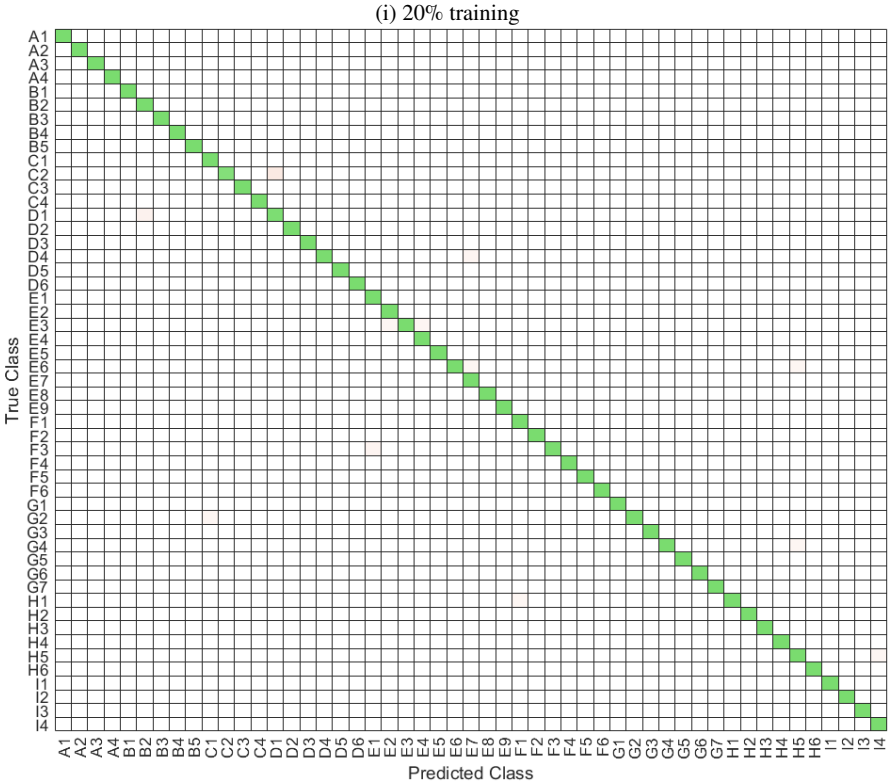
Fig. 13 FRR for 20% and 80% training for 51 ECUs

Improvements can be observed for the rest of the classifiers with the FAR going up to 0.006% in the case of NN.

In Figure 13 we depict the FRR for 20% (i) and 80% (ii) of the dataset used as training data. The FRR for the Tree and LD classifiers, when using 20% of the dataset for training, is far from acceptable with the FRR reaching 100% for several ECUs. Results obtained for KNN, SVM and NN look more promising with a FRR of up to 39% in the case of KNN, below 20% in the case of SVM (with the exception of one node for which we get a FRR of 62%) and below 10% for NN (except for one ECU with a FRR of 37.5%). In the case of 80% of the data used for training, the FRR for the Tree classifier reaches 100% for even more ECUs, while in case of LD the results show slight improvements. The KNN and SVM algorithms show



(i) 20% training



(ii) 80% training

Fig. 14 Confusion matrix for NN with (i) 20% and (ii) 80% training for the 51 ECUs

Table 2 Results for the five classifiers at 20% – 80% training

Trn.	Alg.	Acc.	FAR			FRR			Precision			Recall			F1-score		
			Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
20%	Tree	0.949	0	$< 10^{-3}$	0.013	0	0.184	1.0	0	0.815	1.0	0.418	NaN	1.0	0.319	NaN	0.999
	LD	0.940	0	0.001	0.0143	0	0.168	1.0	0	0.831	1.0	0.161	NaN	1.0	0.201	NaN	0.995
	KNN	0.994	0	$< 10^{-4}$	$< 10^{-3}$	0	0.039	0.395	0.604	0.960	1.0	0.868	0.985	1.0	0.736	0.971	1.0
	SVM	0.995	0	$< 10^{-4}$	$< 10^{-3}$	0	0.0280	0.625	0.375	0.971	1.0	0.951	0.994	1.0	0.545	0.980	1.0
	NN	0.998	0	$< 10^{-4}$	$< 10^{-4}$	0	0.012	0.375	0.625	0.987	1.0	0.833	0.992	1.0	0.714	0.989	1.0
40%	Tree	0.951	0	$< 10^{-3}$	0.013	0	0.194	1.0	0	0.805	1.0	0.279	NaN	1.0	0.229	NaN	1.0
	LD	0.948	0	0.001	0.0128	0	0.132	1.0	0	0.867	1.0	0.394	NaN	1.0	0.434	NaN	1.0
	KNN	0.997	0	$< 10^{-4}$	$< 10^{-3}$	0	0.023	0.250	0.750	0.976	1.0	0.939	0.992	1.0	0.841	0.983	1.0
	SVM	0.997	0	$< 10^{-4}$	$< 10^{-3}$	0	0.018	0.416	0.583	0.981	1.0	0.921	0.994	1.0	0.736	0.987	1.0
	NN	0.999	0	$< 10^{-4}$	$< 10^{-4}$	0	0.001	0.031	0.968	0.998	1.0	0.857	0.996	1.0	0.923	0.997	1.0
60%	Tree	0.941	0	0.001	0.018	0	0.236	1.0	0	0.763	1.0	0.269	NaN	1.0	0.205	NaN	0.999
	LD	0.953	0	$< 10^{-3}$	0.008	0	0.127	1.0	0	0.872	1.0	0.127	NaN	1.0	0.142	NaN	1.0
	KNN	0.998	0	$< 10^{-4}$	$< 10^{-3}$	0	0.0169	0.192	0.807	0.983	1.0	0.898	0.992	1.0	0.875	0.987	1.0
	SVM	0.998	0	$< 10^{-4}$	$< 10^{-3}$	0	0.011	0.375	0.625	0.988	1.0	0.928	0.996	1.0	0.769	0.991	1.0
	NN	0.999	0	$< 10^{-4}$	$< 10^{-4}$	0	0.002	0.027	0.972	0.997	1.0	0.973	0.998	1.0	0.923	0.997	1.0
80%	Tree	0.928	0	0.001	0.028	0	0.253	1.0	0	0.746	1.0	0.343	NaN	1.0	0.265	NaN	1.0
	LD	0.951	0	$< 10^{-3}$	0.008	0	0.117	1.0	0	0.882	1.0	0.275	NaN	1.0	0.360	NaN	1.0
	KNN	0.998	0	$< 10^{-4}$	$< 10^{-3}$	0	0.0141	0.500	0.500	0.985	1.0	0.984	0.998	1.0	0.666	0.990	1.0
	SVM	0.999	0	$< 10^{-4}$	$< 10^{-3}$	0	0.015	0.500	0.500	0.984	1.0	0.973	0.998	1.0	0.666	0.989	1.0
	NN	0.999	0	$< 10^{-4}$	$< 10^{-4}$	0	$< 10^{-3}$	0.010	0.989	0.999	1.0	0.666	0.993	1.0	0.800	0.995	1.0

now considerable improvements with FRR values below 12% with the exception of a node that exhibits a FRR of 50%. The use of NN proves to be the most reliable with FRRs below 1% at 80% training.

In Figure 14 (i) we illustrate the confusion matrix obtained when using NN with 20% of the data employed for training. Occasionally, the ECUs may be misidentified, at this lower training rate, but this is a rare event in general. To complete the image, in Figure 14 (ii) we illustrate the confusion matrix for NN when of 80% of the data is used for training and the remaining 20% for testing. In this case misidentifications are very rare, for example C2 is rarely misidentified as D1 and D1 is rarely misidentified as B2. Since this identification is based on samples from a single bit and multiple bits are available in each frame, the misidentification rate will essentially drop to 0 when multiple bits are used.

In Table 2 we summarize as numerical values the results obtained for all metrics, i.e., minimum, mean and maximum value of FAR, FRR, precision, recall and F1-score, for the 5 classifiers when using 20%, 40%, 60% and 80% of the dataset for training. The value *NaN* means division by zero, i.e., for some ECUs the sum between true positive and false negative is zero. It can be easily seen that the results do improve when increasing the training percentage, but not as significantly as one would expect, which suggests that a small pool of data should be sufficient. Also the KNN, SVM and NN are the classifiers which give the best results with NN clearly outperforming the rest with an accuracy over 99.9%.

5 Discussion and Conclusion

Physical layer based intrusion detection mechanisms show promising results and, as suggested by the results above, neural networks seem to perform much better compared to traditional machine learning algorithms at such tasks. But there are still some challenges that remain to be addressed before such approaches can be included in real-life in-vehicle networks. We now discuss some limitations. The first challenge comes from the high sampling rate required by most of the proposed approaches and the costs involved by integrating the needed HW in a vehicle. The ECUPrint dataset [26] contains data collected at a 500Ms/s sampling rate which is available on high-end oscilloscopes but not on regular microcontrollers. While some works investigate the use of lower sampling rates [4] or cost-effective HW solutions [15] such approaches have yet to be validated as real-life implementation inside vehicles.

Another challenge comes from the effect of environmental factors on physical layer characteristics. Temperature is one of the factors known to influence the characteristic signalling behavior of electronic circuits. A follow-up paper investigating the robustness of the Scission IDS [14] against environmental factors illustrates the importance of considering these elements in the design of physical layer based CAN IDSs. The authors consider the effect of temperature variations and improve their initial proposal to improve detection accuracy in such circumstances. The only methodology that proved surprisingly good resilience against environmental changes is the localization methodology from CAN-SQUARE [9]. Another IDs proposal entitled SIMPLE [7] accounts for both temperature and voltage variations and their effects. To compensate for these effects the authors of SIMPLE implement a secure update procedure for node fingerprints. And this leads us to the third challenge which is updating fingerprints to account for legit variations in physical layer signalling behavior while reliably protecting the IDS from malicious attempts at compromising the procedure to evade IDS detection.

It is no doubt that voltage information can be used to separate between ECUs, but having in mind the previous challenges, there is still room for further investigations. It is also worth noting that other physical features can be used to fingerprint ECUs. For example, clock skews which were commonly used to fingerprint computers [16] have been also recently used to fingerprint [3] or map ECUs inside a car [18]. Unfortunately, clock skews are very easy to clone [30] by adjusting the local clock of the controller, a reason for which clock skews do not seem to be as secure as voltage fingerprints.

References

1. AUTOSAR. *Specification of Secure Onboard Communication*, 4.3.1 edition, 2017.
2. S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno, et al. Comprehensive experimental analyses of automotive attack surfaces. In *USENIX Security Symposium*. San Francisco, 2011.

3. K.-T. Cho and K. G. Shin. Fingerprinting electronic control units for vehicle intrusion detection. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 911–927, 2016.
4. K.-T. Cho and K. G. Shin. Viden: Attacker identification on in-vehicle networks. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pages 1109–1123, New York, NY, USA, 2017. ACM.
5. W. Choi, H. J. Jo, S. Woo, J. Y. Chun, J. Park, and D. H. Lee. Identifying ecus using inimitable characteristics of signals in controller area networks. *IEEE Transactions on Vehicular Technology*, 67(6):4757–4770, June 2018.
6. W. Choi, K. Joo, H. J. Jo, M. C. Park, and D. H. Lee. Voltageids: Low-level communication characteristics for automotive intrusion detection system. *IEEE Transactions on Information Forensics and Security*, 13(8):2114–2129, Aug 2018.
7. M. Foruhandeh, Y. Man, R. Gerdes, M. Li, and T. Chantem. Simple: Single-frame based physical layer identification for intrusion detection and prevention on in-vehicle networks. In *Proceedings of the 35th Annual Computer Security Applications Conference*, ACSAC '19, page 229–244, New York, NY, USA, 2019. Association for Computing Machinery.
8. B. Groza and P. Murvay. Security solutions for the controller area network: Bringing authentication to in-vehicle networks. *IEEE Vehicular Technology Magazine*, 13(1):40–47, March 2018.
9. B. Groza, P.-S. Murvay, L. Popa, and C. Jichici. Can-square-decimeter level localization of electronic control units on can buses. In *European Symposium on Research in Computer Security*, pages 668–690. Springer, 2021.
10. ISO. *11898-1, Road vehicles - Controller area network (CAN)–Part 1: Data link layer and physical signalling*, 2015.
11. ISO. *11898-2, Road vehicles - Controller area network (CAN) Part 2: High-speed medium access unit*, 2016.
12. ISO/SAE. *21434, Road vehicles - Cybersecurity engineering*, 2021.
13. M. Kneib and C. Huth. 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, pages 787–800, New York, NY, USA, 2018. ACM.
14. M. Kneib, O. Schell, and C. Huth. On the robustness of signal characteristic-based sender identification. *arXiv preprint arXiv:1911.09881*, 2019.
15. M. Kneib, O. Schell, and C. Huth. EASI: Edge-Based Sender Identification on Resource-Constrained Platforms for AutomotiveNetworks. In *Proceedings of the 2020 Network and Distributed System Security Symposium*, San Diego, CA, 2020.
16. T. Kohno, A. Broido, and K. C. Claffy. Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, 2(2):93–108, 2005.
17. K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, et al. Experimental security analysis of a modern automobile. In *2010 IEEE Symposium on Security and Privacy*, pages 447–462. IEEE, 2010.
18. S. Kulandaivel, T. Goyal, A. K. Agrawal, and V. Sekar. Canvas: Fast and inexpensive automotive network mapping. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pages 389–405, 2019.
19. T. Limbasiya, K. Z. Teng, S. Chattopadhyay, and J. Zhou. A systematic survey of attack detection and prevention in connected and autonomous vehicles. *arXiv preprint arXiv:2203.14965*, 2022.
20. M. Marchetti and D. Stabili. Read: Reverse engineering of automotive data frames. *IEEE Transactions on Information Forensics and Security*, 14(4):1083–1097, 2019.
21. Mathworks. Choose classifier options. `https://www.mathworks.com/help/stats/choose-a-classifier.html`. [Online; accessed 1-April-2022].
22. C. Miller and C. Valasek. Remote exploitation of an unaltered passenger vehicle. *Black Hat USA*, 2015:91, 2015.
23. P. Murvay and B. Groza. Source identification using signal characteristics in controller area networks. *IEEE Signal Processing Letters*, 21(4):395–399, April 2014.

24. P.-S. Murvay and B. Groza. Tidal-can: Differential timing based intrusion detection and localization for controller area network. *IEEE Access*, 8:68895–68912, 2020.
25. S. Ohira, A. K. Desta, I. Arai, and K. Fujikawa. Pli-tdc: Super fine delay-time based physical-layer identification with time-to-digital converter for in-vehicle networks. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, pages 176–186, 2021.
26. L. Popa, B. Groza, C. Jichici, and P.-S. Murvay. Ecuprint—physical fingerprinting electronic control units on can buses inside cars and sae j1939 compliant vehicles. *IEEE Transactions on Information Forensics and Security*, 17:1185–1200, 2022.
27. Robert Bosch GmbH. *CAN Specification, Version 2.0*, 1991.
28. M. Rumez, J. Dürrwang, T. Brecht, T. Steinshorn, P. Neugebauer, R. Kriesten, and E. Sax. Can radar: Sensing physical devices in can networks based on time domain reflectometry. In *2019 IEEE Vehicular Networking Conference (VNC)*, pages 1–8. IEEE, 2019.
29. SAE International. *J1939-11 – Physical Layer, 250K bits/s, Twisted Shielded Pair*, Sept. 2006.
30. S. U. Sagong, X. Ying, A. Clark, L. Bushnell, and R. Poovendran. Cloaking the clock: emulating clock skew in controller area networks. In *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, pages 32–42. IEEE, 2018.
31. UNECE. *WP.29 Addendum 154 – UN Regulation No. 155, Uniform provisions concerning the approval of vehicles with regards to cyber security and cyber security management system*, March 2021.
32. M. Wolf, A. Weimerskirch, and C. Paar. Security in automotive bus systems. In *Workshop on Embedded Security in Cars*, pages 1–13. Citeseer, 2004.