vProfile: Voltage-Based Anomaly Detection in Controller Area Networks

Nathan Liu
Electrical and Computer Engineering
University of Waterloo
Waterloo, Canada
ndliu@uwaterloo.ca

Carlos Moreno
Electrical and Computer Engineering
University of Waterloo
Waterloo, Canada
cmoreno@uwaterloo.ca

Murray Dunne
Electrical and Computer Engineering
University of Waterloo
Waterloo, Canada
mdunne@uwaterloo.ca

Sebastian Fischmeister
Electrical and Computer Engineering
University of Waterloo
Waterloo, Canada
sfischme@uwaterloo.ca

***Abstract*—Modern cars are becoming more accessible targets for cyberattacks due to the proliferation of wireless communication channels. The intra-vehicle Controller Area Network (CAN) bus lacks authentication, which exposes critical components to interference from less secure, wirelessly compromised modules. To address this issue, we propose vProfile, a sender authentication system based on voltage fingerprints of Electronic Control Units (ECUs). vProfile exploits the physical properties of ECU output voltages on the CAN bus to determine the authenticity of bus messages, which enables the detection of both hijacked ECUs and external devices connected to the bus. We show the potential of vProfile using experiments on two production vehicles with precision and recall scores of over 99.99%. The improved identification rates and more straightforward design of vProfile make it an attractive improvement over existing methods.**

***Index Terms*—Anomaly detection, automotive, Controller Area Network (CAN), intrusion detection, security, side-channel**

I. Introduction

Modern vehicles are becoming more connected, with newer models including cellular network interfaces to augment their comfort and convenience. With more connectivity, however, comes new attack vectors [1]. These new entry points allow attackers to remotely infiltrate systems without physical access, which is particularly concerning when considering the fundamentally insecure yet widely-used Controller Area Network (CAN) bus for intra-vehicle communication.

CAN is an unauthenticated channel, designed in an era where cybersecurity was of little concern for automobiles due to their limited connectvitiy [2]. The lack of authentication means that a node on the bus, called an Electronic Control Unit (ECU), can imitate any other ECU without raising suspicion. This deficiency is a significant flaw since ECUs can control critical system aspects, such as the brakes and airbags in a car. The work by Miller and Valasek on a Jeep Cherokee exploited this and allowed them to send arbitrary CAN messages by infiltrating the infotainment system through the car's cellular connection [3]. They were able to remotely control the engine and brakes of the vehicle, among other components. With fully autonomous vehicles on the horizon, the human override disappears, making compromised vehicles no more than remote-controlled toys.

Authentication methods based on cryptographic schemes, such as message authentication codes, are quick to present themselves but are difficult to implement in CAN communications. The issue with using cryptography is that CAN has a low bandwidth, allowing only 8 bytes of data per message. Furthermore, additional computational power would be needed to achieve the recommended security levels while maintaining real-time constraints [4]. Instead of an active approach, one can passively monitor the bus and related side-channels to look for unusual behavior. These methods are typically known as Intrusion Detection Systems (IDS). Research on IDSs exists in more than just the automotive industry [5], [6].

This work proposes vProfile, a system for verifying the origin of CAN messages by using the distinct electrical properties of a transmitting ECU. Specifically, we use the CAN bus voltage as a reputable side-channel. vProfile integrates into an IDS and enables message sender identification and the ability to detect unauthorized messages. Compared to similar, existing methods that use slow and complex operations, vProfile boasts simplicity while exhibiting low misclassification rates. Because of the versatility of CAN, with use in medical equipment, marine electronics, and factory automation [7], vProfile's applicability exceeds the vehicular domain.

A. Paper Organization

Following the introduction in Section I, Section II presents some prior work related to our proposed method and Section III has background information on the CAN protocol. Section IV then explains vProfile's implementation, Section V contains experimental results from running vProfile in a commercial truck, and Section VI summarizes a case study where we integrate vProfile into an industry partner's vehicle. Next, Section VII describes threats to vProfile's validity, possible

978-3-9819263-5-4/DATE21/©2021 EDAA

solutions, and future work. Finally, we present some closing remarks in Section VIII.

II. Related Work

These works share characteristics with vProfile, in that they also sample analog CAN voltages and exploit some inherent, immutable properties: a method by Murvay and Groza [8], Kneib and Huth's Scission [4], Cho and Shin's Viden [9], and a system by Choi, Jo, Woo, Chun, and Park [10].

Murvay and Groza use a sampling rate of 2 GS/s at 12 bits for a 10 kb/s bus. They first remove noise from the input voltage data with a low pass filter and then store the output as a fingerprint for each ECU. They investigate three signal processing techniques: mean square error (MSE), convolution, and mean-value. This method's disadvantages are the high sampling rate and the high false classification rate. Murvay and Groza used an oscilloscope in their work. However, an analog to digital converter (ADC) with the specifications required for this method is expensive (e.g., the Texas Instruments ADC12D1000). They performed the MSE and convolution experiments on two sets of ten CAN transceivers and yielded an average false positive rate of 3.1% (with a standard deviation of 3.6%) and an average false negative rate of 6.0% (with a standard deviation of 17.9%).

Scission samples a 500 kb/s CAN bus at a rate of 20 MS/s. The message is split into bits and binned into one of three groups based on certain criteria. Scission uses the Relief-F algorithm supplied by the Weka 3 toolkit for feature selection. The system in [4] used the mean, standard deviation, variance, skewness, kurtosis, root mean square, maximum value, and energy as features for each of the three groups. Scission then uses the logistic regression machine learning algorithm for training and classification. The high sampling rate and elaborate pre-processing and feature selection process are some downsides to this approach.

Viden is not so much an IDS but rather a method to enhance an existing IDS by providing the ability to identify the attacking device. It uses a sampling rate of 50 kS/s for a 500 kb/s CAN bus. Viden creates multiple sets of tracking points from *non-ACK* voltage samples (voltages not taken from the acknowledgment (ACK) slot of a message) and uses them to create a *voltage profile* where each profile is unique to an ECU. When the IDS detects an intrusion, Viden must first create an attack profile using multiple malicious messages before identification can occur. This method's drawbacks include the complicated voltage profile creation process, the need to create an attack profile before classification, and the requirement of an underlying IDS.

Choi, Jo, Woo, Chun, and Park's method uses a sampling rate of 2.5 GS/s, a resolution of 12 bits, and considers only the 18-bit extended identifier (ID) of each CAN message. They use LibXtract to calculate 40 features in both the time and frequency domain and keep the best eight time domain and best nine frequency domain features as ranked by the FEAST toolbox. Model training uses a supervised learning algorithm. This method must first extract its 17 features from a message before it can be classified. This approach's two main disadvantages are the high sampling rate and how it can easily miss messages due to its long processing time. As mentioned above, an ADC with the required specifications is cost-prohibitive. During the 1.02 ms required to extract all 17 features, the authors found that two messages can be transmitted on average, assuming a 500 kb/s bus at 50% load.

III. CAN Background

We briefly describe some important aspects of the CAN protocol that relates to vProfile's operation.

A. Architecture

Robert Bosch GmbH released CAN in 1986 [11], creating a protocol that is now ubiquitous not only in the automotive industry but also in aerospace and factory control, among others [12]. The most recent revision of CAN is version 2.0, which specifies two different message formats: the standard frame format with an 11-bit ID and the extended frame format with a 29-bit ID [13]. Because vProfile testing occurred on vehicles that employ the SAE J1939 protocol [14] which uses solely extended frames, this paper explains only the extended frame format and all future mentions of data fields are with respect to J1939.

1) Bus Values: A CAN bus can take one of two complimentary values: *dominant* or *recessive*. If an ECU sends dominant while another sends recessive, then the bus will be dominant. For example, given a wired-AND implementation (assumed for the rest of the paper), a dominant bit is represented by logical '0' and the recessive bit by logical '1'. The bus holds the recessive value when idle [13].

2) Extended Frame Format: CAN specifies four frame types: data frame, remote frame, error frame, and overload frame. We focus on the data frame as it is usually the target of attacks [15]. Fig. 1 shows the format of an extended data frame.

Each ID can only map to a single ECU, but each ECU can send multiple IDs. Thus, the ID can uniquely identify the sender of a legitimate CAN message. The source address (SA) in the last 8 bits of the 29-bit ID exhibits this property, so vProfile needs only the SA to detect intrusions.

IV. Overview of vProfile

We now explain vProfile, including some required background information and its four functional stages.

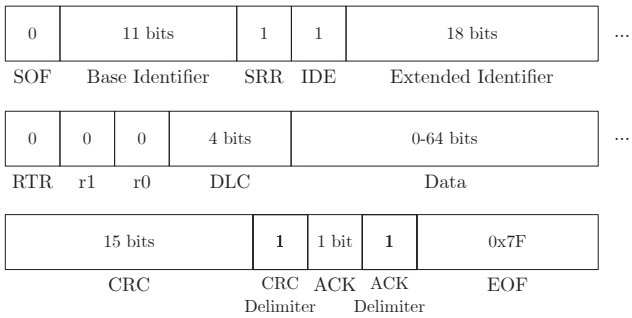

Fig. 1: CAN extended frame format.

A. Preliminaries

This section explains the technologies and techniques behind vProfile.

1) Immutable ECU Property: Minute inconsistencies in manufacturing introduce random physical differences in each ECU that are unpredictable and uncontrollable. These variations suggest that each ECU exhibits unique electrical characteristics. Thus, we can not only uniquely identify ECUs based on these electrical properties, but they are practically impossible for an adversary to imitate [16]. Fig. 2 shows the unique properties of ECUs. We plot the rising and falling edges (together with the recessive state in-between, these form an *edge set*) of 200 voltage traces from multiple SAs mapping to two ECUs on a commercial Sterling truck. Though the traces originate from multiple SAs, it is clear that there are two distinct waveforms, one for each ECU. The units of measurement are irrelevant, so the values have been left as digitized values.

2) Mahalanobis Distance: Mahalanobis distance measures the similarity between an observation and a distribution. For an N-dimensional dataset, it is defined by

$$D(\mathbf{x}) = \sqrt{(\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu})} \tag{1}$$

where $\mathbf{x} = (x_1, x_2, ..., x_N)^T$ is the observation, $\boldsymbol{\mu} = (\mu_1, \mu_2, ..., \mu_N)^T$ is the mean of the distribution, and Σ is the covariance matrix [17]. Note that (1) reduces to Euclidean distance if Σ is the identity matrix. For vProfile, the number of samples in an edge set defines the dimensionality. Taking Fig. 2 as an example where each edge set is 100 samples, the dataset would have 100 dimensions.

The advantage of using the Mahalanobis distance for CAN voltage traces over other distance metrics comes from its use of a covariance matrix. Although each waveform from the same ECU has similar overall characteristics, they are not identical; the variance can differ significantly across samples, even for neighboring samples and especially at the rising and falling edges.

The covariance matrix captures the correlation between a sample and its neighbors, which plays a vital role in distinguishing traces from different sources. This results in a more accurate clustering metric for this use case. To visualize the effect of the matrix, we compare Mahalanobis distance to Euclidean distance. Consider Fig. 3, which includes traces from two different ECUs (ECU 0 and ECU 1) and a test trace originating from ECU 0. Table I shows the Euclidean and Mahalanobis distances from the test trace to the traces from both ECUs. We observe that both distances correctly indicate that the test trace is more similar to messages sent by ECU 0. However, the Mahalanobis distances' quotient is over two orders of magnitude larger than the Euclidean quotient. This difference signifies that Mahalanobis distance sees the test trace as considerably more similar to ECU 0 than ECU 1. In contrast, Euclidean distance sees that it is only slightly more similar to ECU 0.

B. Threat Model

Based on how an adversary could gain access to the CAN bus, we consider two types of intruders.

Hijack Intruder: This adversary gains bus access by hijacking an existing ECU and can then send arbitrary messages using that ECU.

Foreign Intruder: This adversary gains access by connecting a new device to the bus, which they can then use to send crafted messages. We assume that this new device did not exist during model training.

Both intruders can craft messages that appear legitimate at the binary level, including the SA and other metadata bits. Due to the broadcast nature of CAN, there is no need for an intruder to identify themselves with a unique SA to receive responses. Intruders that transmit using a unique SA can be trivially detected, but those that transmit under the ID of an existing, legitimate ECU cannot be detected within the confines of the CAN protocol; this necessitates a side-channel approach.

C. Functionality

The four main stages of vProfile include: *Sampling*, *Preprocessing*, *Training*, and *Detection*. Both Training and Detection require the Sampling and Preprocessing stages.

1) Sampling: There is a point of diminishing returns for high sampling rates and resolutions. The drawback of increased detail is an additional computational cost, which may be infeasible on embedded processors. A good choice for sampling

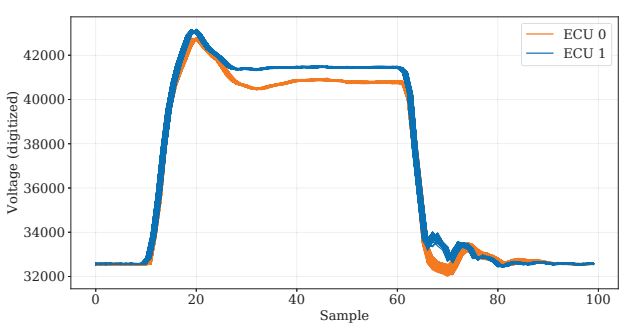

Fig. 2: Voltage differences of messages from two different ECUs and the similarities of messages from the same ECU.

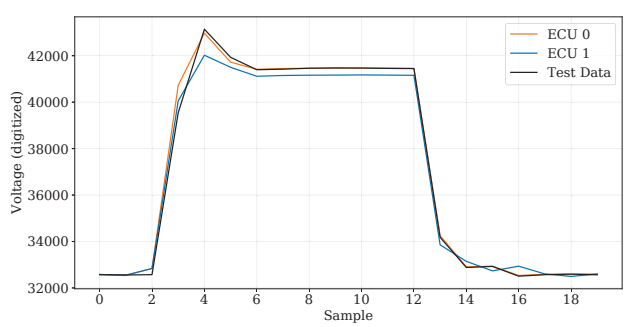

Fig. 3: Traces from two different ECUs and a test trace from ECU 0.

TABLE I: Mahalanobis and Euclidean Distance from Test Data to the two ECUs in Fig. 3

Metric	Distance to ECU 0	Distance to ECU 1	Quotient
Euclidean	1245.35	1665.84	1.34
Mahalanobis	6.32	856.45	135.42

rate and resolution depends on the ECU voltage characteristics of the system on which vProfile runs. For example, a system with many ECUs where some have similar waveforms would benefit from a higher sampling rate and resolution, while a system with fewer ECUs and distinct waveforms might be able to tolerate a lower rate and resolution. Experiments on the vehicles used in Sections V and VI showed that a sampling rate of 10 MS/s and a resolution of 12 bits were sufficient.

2) Preprocessing: Per Fig. 2, the interesting signal characteristics reside in edge sets. In the absence of ECU malfunctions, these characteristics do not change throughout a transmission, so only one edge set is needed for every message [10]. Minimizing the delay from sampling to detection requires that the edge set should be extracted as early in the transmission as possible. Because of the possibility of collisions during arbitration, any bits within the arbitration field are considered unstable. Thus, vProfile uses the first edge set after the arbitration field. The message's SA is also extracted and kept with its edge set.

3) Training: The result of preprocessing is a dataset of edge sets and their corresponding SAs. Since an ECU can send messages using multiple SAs, the dataset needs to be clustered by SA, where a cluster represents all of the messages sent by an ECU. If one is fortunate enough to be provided with a database containing the target system's ECUs and their valid SAs, it is simple to cluster using the database as a lookup table (LUT). If one does not have access to such information, then the following can be done.

Group the data by SA and then calculate the Mahalanobis distance between the edge sets of every pair of SAs and cluster those with the smallest distance. This can be done manually or with a clustering algorithm and some fine-tuning may be required if the physical characteristics of the ECUs are similar.

The model contains each cluster's mean, inverse covariance matrix, and a LUT that maps valid SAs to their cluster. The model also needs a detection threshold so that vProfile can handle traces that do not belong to any cluster. We determine this threshold by finding the largest distance from an edge set to its cluster's mean over all clusters. Algorithm 1 summarizes the training procedure.

4) Detection: Given an edge set to classify, vProfile uses the cluster-SA LUT to first check if the observed SA exists in the system. If it exists, vProfile then uses the LUT to get the cluster based on the reported SA (the expected cluster), predicts the cluster based on the edge set's Mahalanobis distance to the cluster means (the predicted cluster), and compares the two results. If the expected and predicted clusters differ, vProfile raises an alarm. If they match, vProfile compares the minimum calculated Mahalanobis distance to the detection threshold with some configurable margin added to account for additional deviation. If the distance is greater than the sum of the threshold and margin, vProfile raises an alarm. Otherwise, the message is considered legitimate. For messages from ECUs in the training dataset, vProfile can also determine the attack's origin from the predicted cluster. Algorithm 2 summarizes the detection procedure.

Algorithm 1: vProfile training procedure.

```
TRAIN(edgeSets) begin
    if fortunate then
        clustSaLut = CLUSTERBYLUT(edgeSet);
    else
        saLut = GROUPBYSA(edgeSet);
        saMeans = GETMEANS(saLut);
        saInvCovs = GETINVCOVS(saLut);
        clustSaLut = CLUSTERBYMAHDIST(saLut,
          saMeans, saInvCovs);
    end
    clustMeans = GETMEANS(clustSaLut);
    clustInvCovs = GETINVCOVS(clustSaLut);
    clustMaxDists = Array[clustSaLut.size];
    foreach clust ∈ clustSaLut do
        clustMaxDists[clust] = max(MAHDIST(clust,
          edgeSets, clustMeans, clustInvCovs));
    end
    model = (clustSaLut, clustMeans,
      clustInvCovs, clustMaxDists);
    return model;
end
```

V. Experiment

A. Setup

Testing was performed on a commercial Sterling truck (henceforth known as *vehicle A*) with a 250 kb/s CAN bus and two ECUs where the first can send one of two SAs and the other sends only one. Vehicle A was stationary with its engine running during the test. Custom hardware samples (at 10 MS/s and 12 bits), pre-processes, and streams the CAN bus data to a laptop running vProfile. A Kvaser USB to CAN device attached to the bus acts as a third ECU to inject arbitrary messages for testing. Fig. 4 shows the means of 100 edge sets from each ECU and how each waveform is easily distinguishable. For this vehicle, we used a margin of 50.

B. Procedure

To test vProfile's hijack intruder detection capabilities, we added messages sent using the Kvaser to the training set to simulate a third ECU with its own unique SA. We then ran vProfile and used the Kvaser to inject 350 messages imitating ECU 0 and 350 messages imitating ECU 1.

Algorithm 2: vProfile detection procedure.

```
DETECT(testEdgeSet) begin
    sa = testEdgeSet.sa;
    if sa ∉ clustSaLut then
        return ANOMALY;
    end
    expClust = clustSaLut.getCluster(sa);
    minDist = ∞;
    foreach clust ∈ clustSaLut do
        dist = MAHDIST(testEdgeSet, clust,
          clustMeans, clustInvCovs);
        if dist < minDist then
            predClust = clust;
            minDist = dist;
        end
    end
    if (expClust ≠ predClust) || (minDist >
      (clustMaxDists[predClust] + margin)) then
        return ANOMALY;
    end
    return OK;
end
```

For foreign intruders, we trained vProfile with data from only the two existing ECUs. The Kvaser then injected 350 messages imitating ECU 0 and 350 messages imitating ECU 1. We ignore the case where a message has an unknown SA since detection is trivial.

C. Results

Table II shows the confusion matrix for the hijack intruder test and Table III shows the confusion matrix for the foreign intruder test. This test environment produced a detection rate of 100% with no false positives or false negatives.

Running the hijack intruder experiment with 5 MS/s and 2.5 MS/s sampling rates at 12 bits exhibited the same perfect detection rate, indicating that this vehicle can tolerate a lower sampling rate.

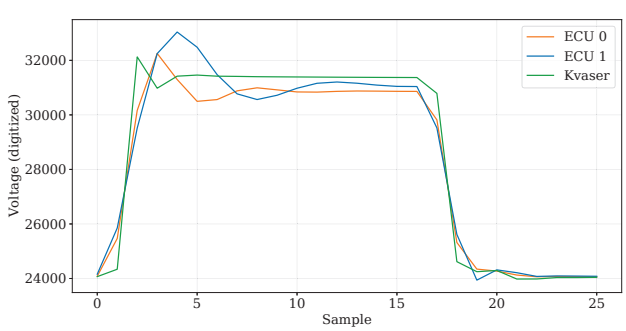

Fig. 4: Means of 100 edge sets from the two ECUs in the Sterling truck and the Kvaser.

VI. Case Study

In collaboration with an industry partner, we evaluated vProfile on one of their vehicles (henceforth known as *vehicle B*). Due to our partner's confidentiality requirements, we omit vehicle-specific details and are limited in what we can disclose besides the final test results.

A. Setup

Vehicle B has a 250 kb/s CAN bus. The partner did not disclose the exact number of ECUs present, but the trained model shows nine clusters, so we can infer that there are likely nine ECUs. We also used a margin of 50 for this vehicle.

B. Procedure

We wanted to conduct the tests while a driver performed various maneuvers, such as hard acceleration and braking, gear shifting, and turning to exercise the vehicle as much as possible. However, we were not allowed to send anything on the bus while vehicle B was in motion, so instead, we performed ECU imitation in software to evaluate vProfile's efficacy. We recorded the bus traffic while the vehicle was driven and then replayed the data into vProfile while changing each message's SA to one that belongs to another cluster with a 20% chance. This experiment simulates a test where every ECU can imitate every other ECU.

C. Results

Table IV shows the confusion matrix and additional metrics are provided in Table V.

This experiment's results further demonstrate vProfile's detection capabilities, exhibiting high scores across all metrics, even in more challenging conditions.

Table VI presents metrics from experiments with lower sampling rates and 12 bits of resolution. Repeating the test four times per sampling rate shows consistent results with no impact on performance. We conclude that a lower sampling rate is feasible for this vehicle.

TABLE II: Hijack Intruder Confusion Matrix

		Predicted	
		Anomaly	Normal
Actual	Anomaly	700	0
	Normal	0	16287

TABLE III: Foreign Intruder Confusion Matrix

		Predicted	
		Anomaly	Normal
Actual	Anomaly	700	0
	Normal	0	16287

TABLE IV: ECU Imitation Confusion Matrix

		Predicted	
		Anomaly	Normal
Actual	Anomaly	65541	0
	Normal	2	261272

TABLE V: Performance Metrics

Metric	Score
Precision	0.99997
Recall	1.00000
F1 Score	0.99998
Informedness	0.99999
Markedness	0.99997

VII. Limitations and Future Work

Despite vProfile's astounding detection capabilities, there exist some limitations. Variances in battery voltage and ECU temperatures can affect the ECU voltages [9]. Though we did not consider these factors in our experiments, one can reduce their effect on vProfile by expanding the training set with data from a wide variety of operating conditions. A more adaptable solution would be an algorithm to update the model with live, streaming data. Next, we conducted experiments on vehicles that use only extended CAN frames, whereas most consumer vehicles use the standard format. However, we do not anticipate many changes to adapt vProfile for standard frames. Furthermore, the current implementation of vProfile cannot detect when a hijacked ECU sends messages with SAs within its normal operating set. For additional coverage, we recommend using vProfile in an IDS that can detect anomalies based on other message properties, such as the period and payload.

Given the above, future work includes using vProfile with standard CAN frames and analyzing the degree to which battery voltage and ECU temperature variances affect vProfile's operation.

VIII. Conclusion

An IDS is an effective way to add security to inherently insecure CAN networks. To this end, we proposed vProfile, a sender authentication system based on voltage fingerprints that enhances IDSs. Experiments show that vProfile achieves high detection rates for both hijacked and foreign ECUs while boasting a more straightforward implementation and lower sampling rates than many existing works. vProfile is a promising method to increase the security of systems that use CAN.

TABLE VI: Results with Downsampled Data

Sampling Rate (MS/s)	Precision	Recall
2.5	0.99997	1.00000
5	1.00000	1.00000

References

[1] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces," in *20th USENIX Security Symposium*, 2011. [Online]. Available: http://static.usenix.org/events/sec11/tech/full_papers/Checkoway.pdf

[2] T. Hoppe, S. Kiltz, and J. Dittmann, "Security threats to automotive CAN networks - practical examples and selected short-term countermeasures," in *Computer Safety, Reliability, and Security, 27th International Conference*, 2008, pp. 235–248.

[3] C. Miller and C. Valasek. (2015) Remote exploitation of an unaltered passenger vehicle. Black Hat USA. [Online]. Available: http://illmatics.com/Remote%20Car%20Hacking.pdf

[4] 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*, ser. CCS '18. New York, NY, USA: ACM, 2018, pp. 787–800.

[5] I. Butun, S. D. Morgera, and R. Sankar, "A survey of intrusion detection systems in wireless sensor networks," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 266–282, 2014.

[6] J. Veeramreddy, V. Prasad, and K. Prasad, "A review of anomaly based intrusion detection systems," *International Journal of Computer Applications*, vol. 28, pp. 26–35, 08 2011.

[7] H. F. Othman, Y. R. Aji, F. T. Fakhreddin, and A. R. Al-Ali, "Controller area networks: evolution and applications," in *2006 2nd International Conference on Information Communication Technologies*, vol. 2, 2006, pp. 3088–3093.

[8] P. Murvay and B. Groza, "Source identification using signal characteristics in controller area networks," *IEEE Signal Processing Letters*, vol. 21, no. 4, pp. 395–399, 2014.

[9] K. 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*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 1109–1123.

[10] 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*, vol. 67, no. 6, pp. 4757–4770, 2018.

[11] History of CAN technology. CAN in Automation. [Online]. Available: https://www.can-cia.org/can-knowledge/can/can-history/

[12] M. D. Natale. (2008, 10) Understanding and using the controller area network. [Online]. Available: https://inst.cs.berkeley.edu/~ee249/fa08/Lectures/handout_canbus2.pdf

[13] (1991) Can specification. BOSCH. [Online]. Available: http://esd.cs.ucr.edu/webres/can20.pdf

[14] J1939 introduction. Kvaser. [Online]. Available: https://www.kvaser.com/about-can/higher-layer-protocols/j1939-introduction/

[15] M. Bozdal, M. Samie, and I. Jennions, "A survey on CAN bus protocol: attacks, challenges, and potential solutions," in *2018 International Conference on Computing, Electronics Communications Engineering*, 2018, pp. 201–205.

[16] R. M. Gerdes, M. Mina, S. F. Russell, and T. E. Daniels, "Physical-layer identification of wired ethernet devices," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 4, pp. 1339–1353, 2012.

[17] R. D. Maesschalck], D. Jouan-Rimbaud, and D. Massart, "The Mahalanobis distance," *Chemometrics and Intelligent Laboratory Systems*, vol. 50, no. 1, pp. 1–18, 2000. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0169743999000477