Towards Viable Intrusion Detection Methods For The Automotive Controller Area Network

Andrew Tomlinson
Systems Security Group, Institute for Future Transport and Cities, Coventry University, UK
tomlin25@uni.coventry.ac.uk

Jeremy Bryans
Systems Security Group, Institute for Future Transport and Cities, Coventry University, UK
jeremy.bryans@coventry.ac.uk

Siraj Ahmed Shaikh
Systems Security Group, Institute for Future Transport and Cities, Coventry University, UK
s.shaikh@coventry.ac.uk

ABSTRACT

The Controller Area Network (CAN) in cars is critical to their safety and performance and is now regarded as being vulnerable to cyberattack. Recent studies have looked at securing the CAN and at intrusion detection methods so that attacks can be quickly identified. The CAN has qualities that distinguish it from other computer networks, while the nature of car production and usage also provide challenges. Thus attack detection methods employed for other networks lack appropriateness for the CAN. This paper surveys the methods that have been investigated for CAN intrusion detection, and considers their implications in terms of practicability and requirements. Consequent developments that will be needed for implementation and research are suggested.

KEYWORDS

intrusion detection, controller area network, automotive cybersecurity

ACM Reference Format:
Andrew Tomlinson, Jeremy Bryans, and Siraj Ahmed Shaikh. 2018. Towards Viable Intrusion Detection Methods For The Automotive Controller Area Network . In *2nd Computer Science in Cars Symposium - Future Challenges in Artificial Intelligence Security for Autonomous Vehicles (CSCS 2018), September 13–14, 2018, Munich, Germany*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3273946.3273950

1 INTRODUCTION

The possibility of cyberattack on the Controller Area Network (CAN) in cars has led to an interest in methods to detect intrusions within the CAN bus traffic [37]. Reviews of methods have considered intrusion detection on computers in general (e.g. [3, 11, 33]), but the nature of the automobile CAN warrants specific considerations. Differentiating factors include the low processing power available to the Electronic Control Units (ECUs) on the CAN; the lifecycle, maintenance and usage patterns of the car; the rapid generation of CAN data records; quick decision requirements, and the limited CAN dictionary sharing by automotive component manufacturers. Also, the safety critical nature of the CAN bus and the criticality of decisions made whilst driving would elevate the legal connotations of a CAN intrusion detection system (IDS).

This paper reviews recent published studies into intrusion detection methods for the CAN bus. The results of those studies are discussed, and the potential for adoption of the methods considered. We also consider additional areas of research that would be warranted before the methods are adopted.

Contributions of this paper:

(1) Review of recent studies into analysis methods for CAN intrusion detection.
(2) Consideration of the implications for implementing those methods in a CAN IDS, and the research challenges for fully evaluating them.

The rest of this paper is structured as follows: Section 2 provides a summary of the CAN. The challenges due to the nature of the automotive CAN and car usage are discussed in Section 3. Section 4 outlines the categories of intrusion detection methods that have been studied, with the studies discussed more fully in Sections 5 and 6. The implications for adoption of those methods are considered in Section 7, with concluding remarks in Section 8.

2 CAN OVERVIEW

The automotive CAN is an in-vehicle network used by components that control vital car functions, including braking, steering and engine control. CAN data has a fixed structure. A few CAN packet types can be broadcast (such as data, remote request, error, and overload), though CAN cybersecurity research, including the studies discussed here, often focuses on the data packet, since this is used to convey information between the ECUs and consequently might be altered to affect some component of the car's functionality. The data component of the CAN data packet comprises 8 one-byte fields, though these might not all be used by the broadcasting ECU. Amongst the other fields in the packet is the identifier field, which contains an ID that ECUs use to determine whether the packet is relevant to them. An ID should be unique to the broadcasting ECU, however, the CAN protocol has no authentication, so any node could conceivably broadcast using an ID belonging to another ECU, or broadcast an ID that is fully invented. Usually, broadcasting packets with invented IDs would not be useful to an attacker, since other nodes would not recognize the ID and ignore the packet. However CAN arbitration dictates that when two ECUs attempt to broadcast packets at the same time, the packet with the lowest ID has priority. Thus, packets with a very low fabricated ID value might be used to dominate the network in a Denial of Service (DoS) attack.

CSCS 2018, September 13–14, 2018, Munich, Germany

ACM ISBN 978-1-4503-6616-8/18/09…$15.00
https://doi.org/10.1145/3273946.3273950

3 CHALLENGES FOR CAN INTRUSION DETECTION

Intrusion detection is based on the assumption that the behaviour of intrusions will differ from the behaviour of legal activity, and a few analytical methods have been proposed as potential candidates for a CAN IDS. For this review we categorise them broadly as signature based and anomaly based, which is a distinction made by researchers across the computing field [26, 29, 30, 33]. Anomaly based approaches are further categorised into their type of method employed, such as statistical, knowledge based, clustering/density based, support vector machines, neural networks, and Markov models. The latter four categories are examples of Machine Learning (ML), discussed later.

The accuracy of an IDS is determined by the number of false positives (wrongly classifying a legitimate event as an attack) and false negatives (failure to detect an attack) it produces. High rates of either are detrimental. Failure to spot attacks is obviously dangerous, but the nature of driving also makes it important to reduce false positives, which could cause distraction to the driver, unwarranted intervention, and future warnings to be ignored. A false positive rate of 0.0001% could still result in five expected false positives per hour in a CAN broadcasting 1500 packets per second. Clearly the IDS would need mechanisms to verify its decisions.

Once an intrusion has been detected, some action would need to be taken. This is not a trivial problem and would need thorough analysis, beyond the scope of this paper. Driver intervention and levels of alert have been considered [14], as have delegation to a central portal, though network factors would be problematical [19]. Whatever the mode of intervention, the driver's peace of mind, avoidance of repeated interaction procedures, and automated support for intervention choices will need addressing [15]. Hoppe, Kiltz and Dittman [15] point out the legal requirement in many countries that automotive systems should not take safety-relevant decisions autonomously, which would imply the IDS triggers only a warning. However, this would assume the driver has sufficient knowledge and time to act appropriately; consequently the inclusion of an autonomous reaction component might be necessary [27]. Either way, update mechanisms would need to be available to tailor the IDS response as threats evolve. Checking that the IDS responses are being interpreted correctly or amended if necessary, could be facilitated through feedback mechanisms and, again, implies the need for updates to the IDS.

The variety of car makes and models, and their varied life-cycles and dispersed location, makes reliable and regular updating a challenge, and it is perhaps unwise to assume owners would be able, or motivated, to update their systems outside the annual service [15, 27]. An option might be over-the-air updating, though this adds a potential attack route and logistical challenges. Of relevance here are proposals to make automotive ECU reflashing faster, simpler and lighter, such as those made by Bogdan et al. [2].

A system for detecting CAN attacks would need to run in real time, detecting intrusions quickly, and rapidly eliciting an appropriate response without disrupting or delaying the broadcast of legitimate packets. Compared to PC systems, an IDS for the automotive CAN is likely to be constrained by limited processing power, limited multi-tasking capability and limited storage capacity [15]. The micro controller commonly used in car ECUs has a memory of 40-50 KB and the computing power is less than 10 MHz [32]. In spite of the low processing power, Hoppe, Kiltz and Dittmann [16] propose that each ECU might be given some processing ability, such as checking packet IDs against its own IDs, thus detecting intrusion attacks. Though this would not work in masquerade attacks that disable the mimicked ECU, and would be difficult to roll-out to existing cars. A retrospectively fitted, or more substantial, IDS would probably need creating as a dedicated unit, either attached to the CAN or attached externally, such as on the ODB-II port.

Research and development into CAN cyberattacks is hindered by two major challenges. First, although the CAN protocol is openly documented [17], the meaning of the data transmitted, and specifics pertaining to any car make or model, such as the expected frequency of broadcasts, or the matching of packet IDs to ECUs or functions, is not available to most researchers. This makes the expected behaviour of the CAN traffic difficult to comprehensively map, and as a consequence, the boundary between a normal signature and an attack signature may be difficult to determine. In addition, the behaviour of an ECU might not be fully known [36].

Second, obtaining labelled attack data is difficult. Again, this is not published by manufacturers, and generating it is difficult because: a) testing attack scenarios using cars is dangerous, costly, potentially damaging to the car, and may require ECU programming knowledge; and b) the nascent nature of CAN cybersecurity means the invented attack scenarios are speculative, incomplete and embryonic.

4 CAN IDS RESEARCH

Table 1 presents a general summary of the methods that have been proposed for network intrusion detection in terms of overheads, advantages and disadvantages. The Set-up overhead column lists the relative machine effort to establish or up-date the decision algorithm, and together with the runtime processing and memory columns is based on opinions expressed in published sources [3, 8, 13]. As discussed below, we acknowledge that determining such implementation constraints is simplistic and error prone without a detailed analysis of the algorithms and their specific configurations. Even so, the table highlights that some methods, such as clustering and density based methods, might be too hungry to run on the limited in-vehicle resources. Likewise, the high training requirements of some methods makes training *in situ* problematical.

The setting up of clustering algorithms can be particularly processing intensive since they may involve nested iterations through the training data-set to identify neighbours. That said, the training overhead across all the methods is high, especially considering the low processing available on ECUs. Training therefore might not be viable in-line. Signature, Statistical and Knowledge-Based methods employ disparate processes and human interventions in their development, which are unlikely to be amenable to automated training, so the overheads are left blank in our table. Although the training overheads are high, the methods have lower resource usage once trained, making them generally amenable to in-line detection [3].

Of course, such general estimates of the overheads are not definitive since variations in the implementations will affect processing

and workload, and would require a thorough analysis of the training algorithms. Applying a density function or grouping the data points as dense and connected regions can reduce the overhead in Clustering methods [1, 3]; likewise, the adjustment of parameters for the SVM [13]. Never-the-less, the overheads would need to be considered and minimized if in-line CAN training is considered.

Table 2 summarises the CAN studies that have tested the IDS methods. Those studies and the methods they employed are discussed in the next sections. The table also shows the attack manifestation that the studies sought to identify: packet flow (changes in the timing or number of packets), and payload manipulation (changes to the values in the packet data fields). Many demonstrated attacks require the injection of packets at sufficient frequency to over-ride legitimate information [36, 41]. However, an attack might conceivably result in little or no change to packet broadcast frequencies, such as the case where a legitimate ECU is disabled while a masquerading component manages to perfectly spoof its broadcast rates [10, 16]. Therefore detection based on packet payload data vales, in addition to detection based on packet flow, is likely to be warranted in a CAN IDS.

Attacks might involve multiple components, defining a convoluted scenario. For example, Valasek and Miller [41] demonstrate an attack to alter the steering. This first requires the car to be fooled into thinking it was in reverse gear (enabling the autoparking functionality to operate), then steering information is broadcast to get the car to steer whilst in drive. The authors found that this would only work at low speed unless the network was also supplied with bogus speed packets. This use of multiple data fields to stage an attack suggests a role for detecting contextual discrepancies; for example, a gear value incongruous with speed or odometer readings. Though the reverse engineering and background analysis conducted by the attack researchers when devising specific attacks on defined car models is not an option for a generic IDS product. Therefore detection methods that are agnostic to the data fields' purpose might be needed.

5 SIGNATURE DETECTION

Signature based intrusion detection assumes that the signature patterns of the attack are known. The discriminatory features from the signatures are stored in a database, against which the subsequent packets are monitored. Packets with features that match the database signatures are flagged as violations.

Since they do not need to adapt to the deployed environment, signature based methods can be easy to deploy initially [33]. They can detect reliably and generate a very low false positive rate when the attack signature is known, and they can determine which type of attack the system is experiencing, which is useful for assessing the epidemiology of attacks [30]. However, given the still emerging mode of automotive cyber-attacks, and the disparate life-cycle of vehicles, signature detection techniques have sometimes been rejected for CAN anomaly detection (e.g. [15] [27]).

With regard to attack signatures, the newness of the automotive cybersecurity field, and the diversity of potential attack modes being uncovered (e.g. [6]), together with the ingenuity of hackers, suggests that it is unwise to assume all attack scenarios will be predicted. A further hindrance is that the efficiency of signature based detection may be degraded when the attack activity spans multiple packets [30]. Additional complications stem from the frequent updates required to keep the signature databases up to date.

In spite of the drawbacks, Studnia et al. [35] proposed a formal language based approach to model both normal ECU behaviour and the behaviour of known attack signatures. Their proposed system first makes a formal check as to whether the packet was compliant with the protocol, followed by a signature-based consistency check against the current system state and the ECU's model. For devising the models, Studnia et al. had access to the specifications of the CAN network and the ECUs of the sampled car. Though they acknowledge that parts of the actual system might not adhere to those specifications, in which cases they suggest the adoption of machine learning approaches for detection. Their system could detect intrusions in real-time, but it sometimes failed to detect an attack if it missed the first packets of the attack broadcast.

6 ANOMALY DETECTION

Anomaly based IDS detect behaviour that deviates from the normal. The assumption is that the normal behaviour can be sufficiently profiled and thresholds established to enable the meaningful identification of deviations. Because anomaly detection is not based on set signatures, these methods have the potential to generalise, making it possible to detect previously unknown attacks [30][29]. Further, the profiles used by the system are customised by the learning algorithm, which makes it difficult for the attacker to know what activity might trigger an alarm [30].

However, such systems require training and the collection of data that is sufficiently representative of normal traffic to enable accurate, reliable anomaly boundaries to be determined. As discussed earlier, acquiring such data for the CAN is problematical. Also, because anomaly detection looks for anomalous events, rather than a specified attack, its rates of false positives and false negatives can be high. Even so, the unpredictability of attack scenarios, together with the difficulties in determining comprehensive robust signatures, has led to many studies adopting the anomaly detection approach. Broadly, these approaches can be categorised into statistical based approaches, knowledge based approaches, and machine learning, which can be further categorised according to the machine learning model applied.

6.1 Statistical

Statistical methods compare the currently observed statistical profile of the system against a previously determined statistical profile. For example, the system might assess deviations in properties such as mean, median and mode. In the case of time series data, such as the CAN bus traffic, statistical methods might employ a rolling window which reduces the size of the immediate data set analysed.

Statistic based techniques can be univariate or multivariate. Univariate techniques model each variable independently. Some factors regarding the manner of broadcast, such as timing intervals between CAN ID broadcasts, might be amenable to univariate analyses, but they are likely to be less useful for applying to the data contents of the CAN messages. Here for example, what appears to be a normal profile (for example, for the steering position) might actually be malicious in the given context. Therefore, multivariate methods

Table 1: Computer network intrusion detection method overheads, advantages and disadvantages. The Overhead columns list the relative machine effort to initially set-up the decision algorithm, and the subsequent relative runtime processing and memory requirements, expressed in published papers [3, 8, 13].

Method Category	Overhead [3, 8, 13]			Advantage	Disadvantage
	Set-up	Processing	Memory		
Signature Based		Low	Medium	Low false positive rate for known attacks.	Assumes the attack signature is already known. Easily bypassed by modified attacks [33].
Statistical		Medium	Low	Requires no prior knowledge about normal activity, and can identify attacks evolving over a long period [11, 30].	Can be retrained by attacker; and not all behaviours can be stochastically modelled [11, 30]. Thresholds may be difficult to determine [30]. Unlikely to capture the interactions between attributes [5].
Knowledge Based		Medium	Medium	Low false positives.	Knowledge may be difficult to develop. May be difficult to maintain [30].
Clustering or Density Based	$O(n^3)$	High	High - may require matching against full training data set.	Simplicity and understandability [3, 29].	Needs feature reduction to reduce high dimensionality [3]. Classification time can become large as the number of neighbour comparisons increases.
Support Vector Machines	$O(n^2)$	Medium	Low	Can deal with high-dimensional data and small samples [3, 33].	Resource-hungry training [29]. Prone to high false positive rate [30]
Neural Networks	$O(emnk)$ where: e epochs, m features, n observations, k hidden neurons.	High	Low	Adaptability to environmental changes [11]. Ability to generalise and cope with noise [20].	Lack of descriptive model explaining decision [11]. Prone to high false positive rate [30].
Hidden Markov Models	$O(nc^2)$ where: c number of states	Medium	Low	Used extensively in IDS [11]. Suitable for assessing transition sequences [26].	Results are dependent on assumptions about the system behaviour[11].

might be better suited for detecting intrusions affecting CAN ECU sensor data.

The complexity of the automobile network is cited as making it difficult to make a precise statistical model that would allow rare but legitimate patterns whilst still maintaining a sufficiently low false positive rate [35]. Even so, statistical models have been considered for CAN intrusion detection, and some researchers proposing other methods have suggested them as a supplemental check [37].

Ling and Feng [22] proposed a system using resetable counters and thresholds to detect any ID broadcast consecutively beyond a threshold-defined number of times. Such broadcasts might be indicative of a DoS attack. However, the authors' reliance on known thresholds has been challenged [4].

Gmiden, Gimden and Trabelsi [12] considered attacks involving the injection of malicious packets. They selected IDs broadcast at fixed time intervals, and used the ID's own regular frequency to monitor for increased broadcast frequencies. Song, Kim and Kim [34] detected attacks where an ID's packets were injected at half the normal time interval. For IDs where the normal interval occasionally fell below 0.2 milliseconds, they determined that this would not happen for more than three consecutive broadcasts, unless an injection attack was taking place.

Cho and Shin [7] considered the timing intervals between packet broadcasts, and proposed a clock-based IDS to detect fabrication, suspension and masquerade attacks. In fabrication attacks, a compromised ECU injects forged packets with a view to distracting receiver ECUs. Suspension attacks entail an ECU being compromised so as to suspend its transmissions. The masquerade attack is potentially harder to detect since it does not necessarily alter the frequency of any ID broadcast, and involves an ECU learning the

Table 2: Intrusion detection methods used in automotive CAN research. The Intrusion Detection column shows the attack manifestation that the studies sought to identify: packet flow (changes in the timing or number of packets), and payload manipulation (changes to the values in the packet data fields).

Category	Study	Intrusion Detection	
		Packet Flow	Payload Manipulation
Signature Based	Studnia et al. [35]: attack signatures modelled using language theory.	X	X
Statistical	Ling and Feng [22]: Consecutively broadcast ID threshold count.	X	
	Cho and Shin [7]: Model of clock behaviours using Recursive Least Squares, with Cumulative Sum analysis to detect anomalies.	X	X
	Song, Kim and Kim [34]: Attack score based on message frequency.	X	
	Gmiden, Gimden and Trabelsi [12]: Time intervals between matching CAN IDs.	X	
	Taylor, Japkowicz and Leblanc [36]: Time interval between messages assessed statistically using T-Tests.	X	
	Lee, Jeong and Kim [21]: Time interval of a remote request broadcast and the received response compared against mean.	X	
	Tomlinson et al. [40]: Time intervals between packets of matching IDs compared with window means using ARIMA and Z-scores.	X	
Knowledge Based	Marchetti and Stabili [24]: Transition matrix.	X	
	Studnia et al. [35] Legitimate state transitions.	X	X
Clustering or Density Based	Martinelli et al. [25]: Four nearest neighbour classifiers tested against an open-sourced CAN dataset.	X	X
	Tomlinson, Bryans and Shaikh [39]: Detection of packet data anomalies using one-class Compound Classifier.		X
Support Vector Machines	Taylor, Japkowicz and Leblanc [36]: Detection of manipulated packet frequencies using One Class Support Vector Machine.	X	
Neural Networks	Kang and Kang [18]: Supervised Deep NN trained using normal and attack CAN packets.	X	
	Taylor, Leblanc and Japkowicz [37]: Anomaly detection in data bit words using Long Short-Term Memory recurrent neural network.		X
	Wasicek and Weimerskirch [42]: Root-Mean-Square function to decide the degree of anomaly from trained Bottleneck NN output.		X
Hidden Markov Models	Narayanan, Mittal and Joshi[28]: Hidden Markov Model detecting anomalous changes to speed and RPM.		X

broadcast patterns of another ECU, which it then forces to suspend broadcasting and itself takes over the broadcasting. Cho and Shin proposed that their method could pinpoint the attacking ECU, even in messages broadcast periodically. Their system uses the receiver ECUs to fingerprint transmitting ECU clock behaviours by using message periodicity to estimate the transmitter's clock skews (the difference in its frequency compared to clocks in other ECUs).

Hoppe, Kiltz and Dittmann [15] devised an attack in which a malicious component broadcasts a mimic packet to turn off the lights whenever it detects a legitimate packet calling for them to be activated. Modelling the flow of packets using an attack simulation, they observed that the frequency of broadcasts for the packet rose beyond the maximum 22(±1) per second seen in normal operation. In addition, the semantic meaning of the packet's data ("on" or "off") in the previous 8 packets switched more frequently than during normal operation. Thus, they treated more than 28 packet broadcasts per second, combined with more than 4 inversions in the previous 8 packets, as signifying an anomaly.

Lee, Jeong and Kim [21] considered the time intervals between the broadcast of a remote request and response packets as a means

of detecting three types of attack: a) DoS by packet injection; b) fuzzy attack by packet injection; c) masquerade attack. The authors reasoned that these attacks would change the intervals between the remote request and a response by affecting packet arbitration sequences, or, in the case of masquerade attack, changes in the distances between nodes, the clock cycles, the processing mechanisms or the physical ECU qualities. They measured: 1) offset (number of messages on the CAN between the remote request and the next message with the matching ID); 2) the proportion of requests with an offset of one; 3) the proportion of requests not responded to in the next six messages (an ECU was deemed to be out of action in this case), and 4) time interval between the request and the matching ID'd message. Conducting DoS and fuzzy attacks that injected records they observed a decrease in offsets of one message, and an increase in none-responses. The masquerade attack did not change the offset rates, since the impersonating node was programmed to respond in the same manner as the impersonated node, but it did lead to a longer delay and greater dispersion in the response times, due, the authors conclude, to the different clock frequency and location of the impersonating node.

The methods discussed above require prior analysis of the traffic on the specific CAN instance to decide the values for the discrimination factors. In contrast, we have looked at methods that might be used unsupervised and would not need prior analysis of the traffic [40]. In simulated attacks, two methods, ARIMA and Z-score were, using broadcast time-intervals, able to detect injected and dropped packets in the highest priority, most regularly broadcast CAN packets, with results similar to a supervised method using prior determined average times. Initial results, however, showed the performance degraded for detection in lower priority packets; though we are looking at ways this might be improved, and also exploring ways to generate more realistic attack data.

6.2 Knowledge based

Knowledge based techniques deduce a set of rules from the training data. Subsequent events are then classified against these rules. These techniques can produce a high capture rate and low false positive rate if the knowledge base is comprehensive enough [11]. However, they rely on knowledge which is difficult to acquire in the case of CAN traffic.

Marchetti and Stabili [24] proposed building a transition matrix, which contained the legitimate paired sequences of CAN IDs. Consecutive messages broadcast onto the CAN would then be validated against the matrix; and messages broadcast out of sequence, flagged as anomalies. Such an approach would require very little storage and have low computational costs. They concluded that this approach gave no false positives, and reliably detected simulated attacks where random sequences of IDs were inserted into the readings. However, the method was less successful at detecting replay attacks wherein known sequences are rebroadcast.

Whilst such a matrix approach based on the next legitimate ID might usefully detect mismatches where an ID is known to be followed by an ID from a very small subset, it would be less successful where an ID could be followed by an ID from a large subset. Our analyses of CAN data from a popular family car shows this to be the case for many of the IDs (Fig. 1).

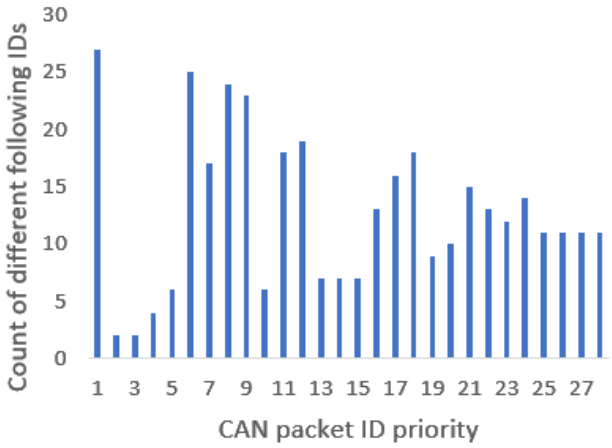

Figure 1: Count of different packet IDs observed to follow each CAN ID in a popular UK family car. The CAN ID rank, rather than ID value, is represented on the x axis.

6.3 Machine Learning

Machine Learning (ML) methods typically attempt to learn patterns in the data, and can be supervised or unsupervised. Supervised learning uses training data containing labelled data, whereas unsupervised learning uses training data in which the labelling of the data is unknown; the machine learning algorithm will then attempt to derive the classes based on some similarity. Some unsupervised machine learning methods offer variations that seek to determine outliers after being trained just using data from known normal instances. During such one-class classification training, data taken from normal operation is analysed to determine a threshold boundary that encompasses the normal instances. Subsequent instances are then classified as normal if they reside within the boundary, or anomaly if they are beyond. One-class classification has been proposed as a likely CAN intrusion detection mechanism [23, 36, 38]. It is, for example, suitable where training data instances of attack behaviour are difficult to generate or predict, or where the simulated attack class risks being too narrowly defined to enable generalisation [43].

6.3.1 Clustering and Outlier Detection. Clustering ML techniques adopt the assumption that instances of a particular class have data profiles that group them into clusters around an archetype. Each new instance can then be classified according to its distance from that archetype. Recent approaches to outlier detection have employed multidimensional distance calculations, enabling multivariate analysis, often combined with density calculations [30].

Clustering approaches can be applied to unsupervised learning [30]. For unsupervised training on data that includes multiple classes, the assumption is that the anomalies will form the smaller of the clusters. For unsupervised training that contains only normal data, a density measure and distance calculation could determine the edge of the normal class boundary.

Martinelli et al [25] tested four nearest neighbour classifiers in an attempt to distinguish between normal CAN packets and simulated

attack CAN packets, solely from the data fields. The classifiers were evaluated according to a range of scores that assess the rates of false and true classifications. All the classifiers were able to classify inserted packets containing gear and RPM data with full accuracy, though were not as accurate on the DoS and fuzzy attacks.

Our own research has investigated a one-class nearest neighbour classifier developed primarily for rapid image processing, which offers comparatively low processing overheads once trained [39]. The Compound Classifier grows a set of hyperspheres to consume the normal instance data-point clusters. Subsequent instances that fall outside all hyperspheres are designated as anomalies. Three CAN data fields comprising continuous sensor-data were used in the classifier. However, it gave a false positive rate on the training data that was high, making it unsuitable in its original configuration, though changes to that configuration might improve its performance.

6.3.2 Hidden Markov Models. Hidden Markov Models attempt to predict the subsequent state of a system from considering the current state. Although they can't be applied to some processes; for example, where the future state is independent of the current state (such as predicting the results in a series of coin tosses), they have been suggested as having potential in CAN anomaly detection [37].

Narayanan, Mittal and Joshi [28] used gradient data derived from CAN packets, rather than absolute values, to implement a Markov-based anomaly detector. From the CAN readings, they built a time-series vector containing speed and RPM data, which they were able to identify in the CAN logs they obtained from three different model cars. They used a Hidden Markov Model to generate transition probabilities (i.e. probability of changing to the next state) and emission probabilities (i.e. the probability of the observed output for the given state). During anomaly detection, the posterior probability of the next read observation was determined using a sliding window of immediately prior observations. Their test data comprised CAN logs in which they changed the data to show anomalies representative of specific behaviours, e.g. sudden increase in speed. The authors report that the system successfully detected the anomalies they generated covering speed and RPM data, both individually, and in combination.

6.3.3 Support Vector Machines. Support Vector Machines (SVMs) seek to separate classes using a hyperplane that follows the centre of the largest margin between classes. The instances on the edge of each class that determine the margin become the support vectors. SVMs have the ability to learn from small samples; can cope with high-dimensional data, and can self-learn [33]. A variant of the SVM particularly relevant to anomaly detection is the One-Class SVMs (OCSVMs), which attempts to create a suitable boundary when trained using only normal instances.

Observing that many IDs seem to be broadcast at fixed frequencies, Taylor, Japkowicz and Leblanc [36] restricted their study to examples of these, and analysed flows measuring broadcast frequencies and average data-content changes represented by hamming distance. They compared these with historical values to determine anomalies. Their study concentrated on attacks that might alter the ID frequency, either by injecting extra packets or erasing packets, and compared a statistical approach (T-tests) against a OCSVM. They found that the Hamming distance added no discriminatory power, and dropped it from their subsequent comparisons. The statistical method was able to usefully detect attacks of one second, though not shorter ones. In comparison, the OCSVM was able to perfectly detect attacks lasting 0.5s or higher, and reliably detected short attacks lasting 0.2s in which just three packets were inserted.

6.3.4 Neural Networks. Inspired by biological neural functioning, a neural network comprises many simple processing nodes working in parallel to produced a decision based on their cumulated outputs. The functioning of the neural network is determined by the network structure, the processing carried out in each node, and the adjustment of connection strengths (or weights) applied to the connections between the nodes.

Kang and Kang [18] looked at packet injection attacks, and tested a neural network on data from three CAN IDs generated using a simulator. Attack data was generated by manipulating the data in the packets to deceive the system and, in addition, Gaussian noise was added to the value information to add natural randomness.

Taking only packets that used all the available data fields, they used the neural network to determine the probability of bit values in any position of the data field. Vectors fed into the neural network comprised the probability that the bit-symbol at a given position was 1, and the "attack" or "normal" label assigned to the training sample. Weights in the neural network were tuned using back-propagation to minimize the mean squared error between the predicted value and the output. They concluded that compared to conventional feedforward neural networks, theirs had a more accurate and consistent detection performance and the speed of testing suggested real-time decision making was viable.

Taylor, Leblanc and Japkowicz [37] considered an attack that transmits legitimate packets that are anomalous only in terms of the context of recent packets on the bus. Such an attack might occur, for example, where malicious Keep Lane Assist messages falsify the steering requirement. They used a Long Short-Term Memory (LSTM) recurrent neural network (RNN) for the anomaly detection. The LSTM component is added to the RNN, offering the capacity to forget, or to remember, more distant events, thus enabling consideration of contextual information.

For each ID, the authors trained a detector to predict the next packet data bit values, the output being predicted values of between 0 and 1 for each of the 64 bits in the data fields. They found that the LSTM worked well for detecting unusual bit patterns (such as when bit values are randomly flipped). However, there were three IDs for which the anomaly detection was poor; the reasons were not readily apparent to the authors. They also point out that the IDs were assessed independently, so any anomalies in the interactions between them would not necessarily be picked up, although this is common across many methods tested.

Wasicek and Weimerskirck [42] considered options for detecting chip tuning. This involves an attacker deliberately changing the software or parameters of an ECU, or adding new hardware that acts in an abnormal manner. The authors chose three features, speed, RPM and torque, which they felt would characterise an engine's power behaviour in different traffic situations. Data was collected from simulated CAN readings from the TORCS racing simulator. Vectors comprised mean value, standard deviation, variance, skewness and kurtosis for the features for a time period

and for a shifted time period. These were recorded for each of the three features, giving 30 input values. Including a time-shift period, they propose, would reduce the impact of any noise in the data. The training vectors were fed into a bottleneck artificial neural network, trained using back-propagation. The outputs from the neural network were combined using a Root-Mean-Square function to generate an anomaly score. Although they found some success in that the true positive rate was higher than the false positive rate, the authors concluded that there was room for improvement, a conclusion borne out in the close proximity of their presented Receiver Operating Characteristic curves to the random guess curve.

7 RESEARCH AND DEVELOPMENT IMPLICATIONS

The development of a viable CAN IDS will require trade-off choices with regard to the detection methods chosen. Signature based methods that offer the lowest rate of false positives, require the careful and comprehensive encoding of the attack signatures, which are still emerging. They have a higher need for frequent updating, which is potentially problematical for the dispersed and varied CAN instances. Anomaly detection methods are prone to higher levels of false decisions, but might capture unforeseen attacks and require fewer updates. They also seem more attuned to the limited CAN packet information available to the researcher and IDS developer.

ML based anomaly detection might afford a means to adapt in some situations, though even in these occasional updates will be inevitable. The frequency, content and magnitude of updating needs researching, in addition to the options for its implementation. For example, it should be established whether CAN traffic could be regarded as stationary over months or years, and if it is not, what changes might be expected. If the IDS is implemented *in situ*, on an ECU or on a CAN gateway, processing capacity is likely to be too small to exploit ML retraining which might be useful occasionally. Dedicated ML GPUs might be considered here, though could be too expensive. It is likely, therefore that algorithm training and updating would be done centrally, and distributed to the car's IDS via some over-the-air mechanism.

Given safety-criticality and legal impacts of driving decisions, the decisions of the CAN IDS are likely to have important ramifications. Retraining in methods such as ML, as well as system updates, pose a challenge for repeatability since it means their parameters and decision paths might not be static. Reliable and secure checkpointing or logging of decisions and parameters, will probably be needed if updates are made.

An IDS could present a complex decision path, particularly with ML, which might employ complex calculations and hidden layers. Explainability, and hence trust, is important where the driver has to act on the decision of the IDS, or might even call the decision's validity into question. Likewise developers, as well as the designers of any interfacing systems (such as automated responses), will also need to understand the reasons of each detection decision. Perhaps useful here are methods such as the open source package LIME [31], which seek to uncover and clarify the decision structure. Though of course, such additions have overheads and might add extra vulnerabilities.

Many of the discussed studies considered methods that would be used in specific attacks, so it is likely an IDS would need to employ multiple methods to cover the range of potential attacks. Taylor, Leblanc and Japkowicz [37] consider that CAN anomaly detection will require a number of methods, such as frequency-based detectors, in addition to their studied neural network methods. Similarly, Mutter, Groll and Freiling [27] propose eight detectors covering the legitimacy of the packet and its structure; its plausibility, based on data from other packets or earlier broadcasts; its frequency, and even its consistency with data from sources external to the CAN network. Attacks that change the frequency of packets, might be the most straightforward to contemplate. However, attacks that involve replacing legitimate packets with malicious ones, or forging packets that usually only broadcast in response to an event, are more challenging [35]. These might be detected through anomalies in the payload (such as the methods represented in the Payload column in Table 2), or some measurable change resulting from the properties of the masquerading ECU.

The generation of labelled data for training and testing is another hurdle to be overcome. Some of the studies fabricated data by manipulating extracted CAN logs (e.g. [37, 40]), though these might be less realistic and miss knock-on effects of the attack, such as packets usurping lower ID packets in chain. Unsurprisingly given the risks and costs, very few studies used data from attacks to actual cars. Perhaps the most suitable option (adopted by a few of the studies, e.g. [18, 42]) is to use simulators, such as the testbed proposed by [9], which could provide a set of standardised benchmark tests.

8 CONCLUDING REMARKS

A range of analysis methods have been proposed for detecting intrusions on the CAN. These are targeted at specific attacks, consequently an IDS might need to adopt an ensemble of them to cover the likely range of attacks.

Methods for implementing, managing and updating the IDS have received less research coverage. These will be impacted by the analysis methods chosen, and will need careful consideration given the constraints and unpredictabilities of automotive lifecycles and usage. We hope the issues outlined in this paper will spark more debate in this area. The safety critical nature of driving suggests, also, that trust, clarity and auditing need consideration. Also to be resolved are the problems of acquiring representative attack data. This requires continued, comprehensive research into probable attack scenarios, as well as methods to generate realistic training and test data.

9 ACKNOWLEDGEMENTS

We would like to thank Dr Harsha Kumara Kalutarage, Queen's University, Belfast, for advice and for assistance in data collection.

REFERENCES

[1] Bruce G Batchelor. 2012. Colour Recognition. In *Machine Vision Handbook*, Bruce G Batchelor (Ed.). Springer-Verlag, London, Chapter 16, 665–694. https://doi.org/10.1007/978-1-84996-169-1

[2] Daniel Bogdan, Razvan Bogdan, and Mircea Popa. 2016. Delta flashing of an ECU in the automotive industry. *SACI 2016 - 11th IEEE International Symposium on Applied Computational Intelligence and Informatics, Proceedings* (2016), 503–508. https://doi.org/10.1109/SACI.2016.7507429

[3] A Buczak and E Guven. 2015. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials* PP, 99 (2015), 1. https://doi.org/10.1109/COMST.2015.2494502

[4] Paul Carsten, Todd R Andel, Mark Yampolskiy, and Jeffrey T McDonald. 2015. In-Vehicle Networks: Attacks, Vulnerabilities, and Proposed Solutions. In *Proceedings of the 10th Annual Cyber and Information Security Research Conference*. https://doi.org/10.1145/2746266.2746267

[5] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection. *Comput. Surveys* 41, 3 (2009), 1–58. https://doi.org/10.1145/1541880.1541882 arXiv:arXiv:1011.1669v3

[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 *20th USENIX Security Symposium*, Vol. August. The USENIX Association, San Francisco, 77–92. 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 *Proceedings of the 25th USENIX Security Symposium*, Thorsten Holz and Stefan Savage (Eds.). USENIX Association, Austin, TX, 911–927. https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/cho

[8] Gary Ericson and William Anton Rohm. 2017. How to choose algorithms for Microsoft Azure Machine Learning. (2017). https://docs.microsoft.com/en-us/azure/machine-learning/studio/algorithm-choice

[9] Daniel S. Fowler, Madeline Cheah, Siraj Ahmed Shaikh, and Jeremy Bryans. 2017. Towards a Testbed for Automotive Cybersecurity. *Proceedings - 10th IEEE International Conference on Software Testing, Verification and Validation, ICST 2017* (2017), 540–541. https://doi.org/10.1109/ICST.2017.62

[10] Sibylle Fröschle and Alexander Stühring. 2017. Analyzing the capabilities of the CAN Attacker. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 10492 LNCS (2017), 464–482. https://doi.org/10.1007/978-3-319-66402-6-27 arXiv:9780201398298

[11] Pedro Garcia-Teodoro, J. Diaz-Verdejo, Gabriel Maciá-Fernández, and Enrique Vázquez. 2009. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security* 28, 1 (2009), 18–28. https://doi.org/10.1016/j.cose.2008.08.003

[12] M Gmiden, M H Gmiden, and H Trabelsi. 2016. An intrusion detection method for securing in-vehicle CAN bus. *2016 17th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA)* (2016), 176–180. https://doi.org/10.1109/STA.2016.7952095

[13] Markus Goldstein, Markus Goldstein, and Seiichi Uchida. 2016. A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data. *PLoS ONE* April (2016), 1–31. https://doi.org/10.7910/DVN/OPQMVF

[14] Tobias Hoppe, Stefan Kiltz, and Jana Dittmann. 2008. Adaptive dynamic reaction to automotive it security incidents using multimedia car environment. *Proceedings of the 4th International Symposium on Information Assurance and Security* (2008), 295–298. https://doi.org/10.1109/IAS.2008.45

[15] Tobias Hoppe, Stefan Kiltz, and Jana Dittmann. 2009. Applying Intrusion Detection to Automotive IT – Early Insights and Remaining Challenges. *Journal of Information Assurance and Security* 4, May (2009), 226–235.

[16] Tobias Hoppe, Stefan Kiltz, and Jana Dittmann. 2011. Security threats to automotive CAN networksPractical examples and selected short-term countermeasures. *Reliability Engineering and System Safety* 96, 1 (2011), 11–25. https://doi.org/10.1016/j.ress.2010.06.026

[17] International Organization for Standardization. 2015. *ISO 11898-1:2015 Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signalling*. Technical Report. https://www.iso.org/standard/63648.html

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

[19] Pierre Kleberger, Tomas Olovsson, and Erland Jonsson. 2011. Security aspects of the in-vehicle network in the connected car. *IEEE Intelligent Vehicles Symposium, Proceedings* Iv (2011), 528–533. https://doi.org/10.1109/IVS.2011.5940525 arXiv:1204.3968

[20] David Kriesel. 2007. *A Brief Introduction to Neural Networks*. http://www.dkriesel.com

[21] Hyunsung Lee, Seong Hoon Jeong, and Huy Kang Kim. 2017. OTIDS : A Novel Intrusion Detection System for In-vehicle Network by using Remote Frame. *Privacy, Security, and Trust 2017* (2017). http://ocslab.hksecurity.net/Dataset/CAN-intrusion-dataset

[22] Congli Ling and Dongqin Feng. 2012. An Algorithm for Detection of Malicious Messages on CAN Buses. In *National Conference on Information Technology and Computer Science*. Atlantis Press. https://doi.org/10.2991/citcs.2012.161

[23] Leandros A Maglaras. 2015. A Novel Distributed Intrusion Detection System for Vehicular Ad Hoc Networks. *International Journal of Advanced Computer Science and Applications* 6, 4 (2015), 1–6. https://doi.org/10.14569/IJACSA.2015.060414

[24] Mirco Marchetti and Dario Stabili. 2017. Anomaly detection of CAN bus messages through analysis of ID sequences. *IEEE Intelligent Vehicles Symposium, Proceedings* Iv (2017), 1577–1583. https://doi.org/10.1109/IVS.2017.7995934

[25] Fabio Martinelli, Francesco Mercaldo, Vittoria Nardone, and Antonella Santone. 2017. Car Hacking Identification through Fuzzy Logic Algorithms. In *IEEE International Conference on Fuzzy Systems*. Naples.

[26] A Murali and M Rao. 2005. A survey on Intrusion Detection Approaches. In *International Conference on Information and Communication Technologies: 2005*. IEEE, Karachi, 233–240. https://doi.org/10.1109/ICICT.2005.1598592

[27] Michael Müter, André Groll, and Felix C. Freiling. 2010. A structured approach to anomaly detection for in-vehicle networks. *2010 6th International Conference on Information Assurance and Security, IAS 2010* (2010), 92–98. https://doi.org/10.1109/ISIAS.2010.5604050

[28] Sandeep Nair Narayanan, Sudip Mittal, and Anupam Joshi. 2016. OBD_SecureAlert: An Anomaly Detection System for Vehicles. *2016 IEEE International Conference on Smart Computing (SMARTCOMP)* (2016), 1–6. https://doi.org/10.1109/SMARTCOMP.2016.7501710

[29] Salima Omar, Asri Ngadi, and Hamid H Jebur. 2013. Machine Learning Techniques for Anomaly Detection: An Overview. *International Journal of Computer Applications* 79, 2 (2013), 975–8887. https://doi.org/10.5120/13715-1478

[30] Animesh Patcha and Jung Min Park. 2007. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks* 51, 12 (2007), 3448–3470. https://doi.org/10.1016/j.comnet.2007.02.001

[31] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016)*. ACM, San Francisco, California, USA, 1135–1144. https://doi.org/10.1145/2939672.2939778

[32] Sandro Schulze, Mario Pukall, Gunter Saake, and T Hoppe. 2009. On the need of data management in automotive systems. *BTW* 144, C (2009), 217–226. http://wwwiti.cs.uni-magdeburg.de/{~}sanschul/papers/BTW2009autodama{_}final.pdf

[33] Jayveer Singh and Manisha J Nene. 2013. A Survey on Machine Learning Techniques for Intrusion Detection Systems. *International Journal of Advanced Research in Computer and Communication Engineering* 2, 11 (2013), 4349–4355.

[34] 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. *International Conference on Information Networking* 2016-March (2016), 63–68. https://doi.org/10.1109/ICOIN.2016.7427089

[35] Ivan Studnia, Eric Alata, Vincent Nicomette, Mohamed Ka^, Ivan Studnia, Eric Alata, Vincent Nicomette, Mohamed Ka^, and Youssef Laarouchi A. 2015. A language-based intrusion detection approach for automotive embedded network. In *21st IEEE Pacific Rim International Symposium on Dependable Computing*, IEEE (Ed.). Zhangjiajie, China.

[36] Adrian Taylor, Nathalie Japkowicz, and Sylvain Leblanc. 2015. Frequency-based anomaly detection for the automotive CAN bus. *2015 World Congress on Industrial Control Systems Security, WCICSS 2015* (2015), 45–49. https://doi.org/10.1109/WCICSS.2015.7420322

[37] Adrian Taylor, Sylvain Leblanc, and Nathalie Japkowicz. 2016. Anomaly Detection in Automobile Control Network Data with Long Short-Term Memory Networks. *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)* (2016), 130–139. https://doi.org/10.1109/DSAA.2016.20

[38] Andreas Theissler and Ian Dear. 2012. Detecting anomalies in recordings from test drives based on a training set of normal instances. In *Proceedings of the IADIS International Conference Intelligent Systems and Agents and European Conference on Data Mining 2012*. IADIS Press, 124–132.

[39] Andrew Tomlinson, Jeremy Bryans, and Siraj Ahmed Shaikh. 2018. Using A One-Class Compound Classifier To Detect In-Vehicle Network Attacks. In *GECCO '18 Companion: Genetic and Evolutionary Computation Conference Companion*. ACM, Kyoto. https://doi.org/10.1145/3205651.3208223

[40] Andrew Tomlinson, Jeremy Bryans, Siraj Ahmed Shaikh, and Harsha Kumara Kalutarage. 2018. Detection of Automotive CAN Cyber-Attacks by Identifying Packet Timing Anomalies in Time Windows. In *4th Workshop on Safety and Security of Intelligent Vehicles (SSIV 2018)*. Luxembourg. https://doi.org/10.1109/DSN-W.2018.00069

[41] Chris Valasek and Charlie Miller. 2013. Adventures in Automotive Networks and Control Units. *Technical White Paper* (2013), 99. http://www.ioactive.com/pdfs/IOActive{_}Adventures{_}in{_}Automotive{_}Networks{_}and{_}Control{_}Units.pdf

[42] Armin Wasicek and Andre Weimerskirch. 2015. Recognizing Manipulated Electronic Control Units. *SAE International Journal of Passenger Cars - Electronic and Electrical Systems* (2015), 1–8. https://doi.org/10.4271/2015-01-0202.Copyright

[43] Ian H Witten, Eibe Frank, and Mark A Hall. 2011. *Data Mining: Practical Machine Learning Tools and Techniques* (3rd ed.). Morgan Kaufmann Publishers.