Correlation-based Anomaly Detection for the CAN Bus*

András Gazdag[1][0000−0002−4481−3308], György Lupták[1][0000−0002−6491−2266], and Levente Buttyán[2][0000−0003−4233−2559]

[1] Laboratory of Cryptography and System Security (CrySyS Lab)
Deptartment of Networked Systems and Services
Budapest University of Technology and Economics
{agazdag,gyorgy.luptak}@crysys.hu
[2] Ukatemi Technologies
buttyan@ukatemi.com

Abstract. Previous attacks have shown that in-vehicle networks have vulnerabilities and a successful attack could lead to significant financial loss and danger to life. In this paper, we propose a Pearson correlation based anomaly detection algorithm to detect CAN message modification attacks. The algorithm does not need a priori information about the communication: it identifies signals based on statistical properties, finds the important correlation coefficients for the correlating signals, and detects attacks as deviations from a previously learned normal state.

Keywords: Controller Area Network · Anomaly Detection · Correlation

1 Introduction

Modern vehicles are equipped with a large number of embedded controllers (ECUs) and other embedded computing devices, which are interconnected with networks internal to the vehicle (e.g., CAN buses, Ethernet), and some of these devices also have interfaces to external networks (e.g., Bluetooth, WiFi, 4G). The ECUs are responsible for various functions of the vehicle, some of which are safety critical. This setup makes vehicles subject to cyberattacks, whereby malicious actors may try to interfere with the behavior of the vehicle by accessing its internal components via its aforementioned external interfaces. The feasibility of such attacks on road vehicles have been demonstrated by researchers as a proof-of-concept [10], and there have been some real cases as well [16]. Unfortunately, attacks affecting safety critical functions may result in potentially fatal

* The presented work was carried out within the MASPOV Project (KTI_KVIG_4-1_2021), which has been implemented with support provided by the Government of Hungary in the context of the Innovative Mobility Program of KTI. The research presented in this paper have also been supported by the NRDI Office, Ministry of Innovation and Technology, Hungary, within the framework of the Autonomous Systems National Laboratory Programme, and the NRDI Fund based on the charter of bolster issued by the NRDI Office.

accidents. Hence, it is clear that vehicles must be protected against cyberattacks, and the first step of this is to make them capable of detecting when they are attacked.

An overwhelming majority of the demonstrated attacks rely on injecting fake messages into the CAN bus. Some ECUs can be easily misled by the fake information in those injected illegitimate messages if they overweight the legitimate ones carrying the correct information. These injection attacks, however, are not so difficult to detect: one can observe the timing statistics of different types of messages and detect deviations from expected values by simple heuristic rules. For instance, [15] introduced the idea of analyzing the rate of messages for in-vehicle attack detection. In the normal state of vehicle operation, most message IDs appearing on the CAN bus have a regular frequency. When an attacker injects messages to achieve some malicious goal, the frequency of some message IDs will unexpectedly increase, as the legitimate ECUs will still send messages periodically with those message IDs besides the attacker's injected messages. Moreover, the frequency may be increased by a factor ranging from 2 to 100, as pointed out in [15]. Hence changes in frequencies can be detected quite easily by simple comparison to some fixed thresholds within a certain size of observation window. Equivalently, increased frequency of a message ID can be translated to decreased inter-arrival times for that message ID, and hence, changes in the statistics of inter-arrival times of certain message types can also serve as the basis for attack detection.

While the majority of attacks on the CAN bus indeed relies on message injection, this is not the only technique to achieve malicious goals. The predictability of message ID frequencies alone is not sufficient for detecting attacks in case of irregular or unpredictable CAN messages and in case of attacks that do not inject new messages on the CAN bus. In this paper, we address the latter problem: we propose a method to detect message modification attacks. Message modification attacks are more difficult to carry out, but they are also much more difficult to detect, therefore, attackers may consider them in the future, especially, given that message injection attacks will likely be detected by future vehicles. Achieving a message modification attack is difficult because the built in safety features of the CAN bus prevent a deliberate modification of a message on the fly. Hence, three options remain: (1) either the attacker compromises the relevant ECU to modify the message before it is even transmitted; (2) or a CAN gateway between two segments is compromised to modify a message during the hand-off between segments; (3) or the CAN bus is divided into two segments with a new malicious gateway inserted, allowing for message modifications on the gateways between the segments. The first two can be performed purely with software exploits, while the last requires physical modification of the network. Despite the increased complexity, the first and the last approaches to message modification attacks have already been demonstrated in [16] and in [3], respectively.

Our anomaly detection solution utilizes the fact that vehicle signals are correlated. The speed, the engine revolution, the current fuel consumption and many other values change together, representing the physical changes of the vehicle

state. The solution proposed in this paper can represent these high level relationships between the vehicle signals with the correlation values that correspond to the normal state of the vehicle. During an attack, if a vehicle signal is overwritten by the attacker in a CAN message, some of the measured correlation values may deviate from those of the normal state, and this can be detected as an anomaly. An advantage of this approach, compared to previously proposed algorithms where only a single signal value is used in the anomaly detection, is that if the attacker does not modify all the correlating signals precisely at the same time, the attack can likely be detected.

The rest of the paper is structured as follows. Section 2 presents the related work. Section 3 introduces our proposed anomaly detection method and Section 1 summarizes its testing and validation. Finally, Section 5 concludes the paper.

2 Related work

Academic papers proposing solutions for securing in-vehicle networks can be divided into three groups: (1) a relatively large body of work (e.g., [25, 5, 18]) is concerned with adding extensions to the CAN protocol, and by doing so, fixing its inherent security weaknesses; (2) a few papers (e.g., [14]) discuss intrusion prevention either by introducing firewalls or other techniques; and (3) another set of papers (which we discuss in more details below) deal with intrusion detection on the CAN bus using various approaches. Given the considerable amount of work done in the field, a few surveys have also been published: [19, 9] have the broad scope of in-vehicle security as a whole, and [11, 27, 26] are more focused on in-vehicle intrusion detection. As our work falls in the domain of anomaly-based intrusion detection, we focus on that domain in the rest of this section.

Anomaly-based intrusion detection can take two approaches: *specification-based* and *model-based* anomaly detection. In case of the former, the normal behavior of the monitored system is explicitly specified. For instance, in the automotive case, the car manufacturer can have specifications for the normal frequency of different periodic messages at which they appear on the CAN bus. Deviations from the specification can easily be identified as signs of attack. In the case of model-based anomaly detection, on the other hand, no explicit specification of normal behavior is given, but instead, a model of the normal behavior is somehow obtained (e.g., learned by observing the system in the non-attacked state), and deviations from this model are detected as attacks. Different academic proposals differ in what modelling technique they use and what features of the system are modelled.

As for the modelling technique, many papers propose to use some statistical model (e.g., mean, variance, or entropy) of some parameter, with simple heuristic rules (e.g., comparison to a threshold) [15, 21, 20, 4, 17] or more sophisticated statistical hypothesis testing methods [12, 24] to detect deviations from the model. Other papers (e.g., [23, 22, 13, 7]) use some machine learning model (e.g., classifier or neural network), with the corresponding model specific method to decide if some input deviates from the learned model. Regarding the features

that are modelled, many papers consider properties of the network traffic itself, such as packet timing features (e.g., frequency of certain types of packets) [21, 20, 17, 24] and features related to the content of the packets (e.g., the time series of packet IDs or certain signal values) [13, 8, 12], whereas some papers use physical characteristics as features, such as voltage level and clock drift of physical signals on the CAN bus [2, 6].

In this paper, we propose a model-based anomaly detection method for the CAN bus that uses correlations across different types of messages as features. To the best of our knowledge the only other paper using correlation-based anomaly detection is [1] therefore, we make a more in-depth comparison here. The method proposed in [1] computes the correlation between two specific signals, velocity and RPM, and detects attacks where extra messages containing incorrect values of these signals are injected into the bus. In contrast to this, our method computes the correlation between all pairs of signals, and identifies those pairs that have high correlation without identifying the actual signals. In addition, we detect message modification attacks, which are more difficult to detect than injection attacks. We simulate seven different types of modification attacks and evaluate the performance of our method for each of them. Otherwise, both [1] and our work use the Pearson correlation function, while in [1], the authors applied other analysis techniques (such as K-Means and Hidden Markov Models) as well.

3 Anomaly Detection algorithm

3.1 Attacker model

Our anomaly detection algorithm focuses only on the detection of message modification attacks, where the original repetition times of the messages are unchanged. As a result only the content of the messages can be used for anomaly detection.

3.2 Overview

We propose an anomaly detection algorithm that uses the correlation between signals encoded in CAN messages. Under normal conditions, the correlation between different signal pairs stays within a (signal pair specific) interval. In case of an attack where the attacker modifies only one member of a correlating signal pair, the resulting correlation may no longer stay within the interval, and this can be detected as an anomaly.

In the training phase, the correlation values between signals has to be determined. We measured multiple times the pairwise Pearson correlation between every signal pair in a one minute long time window and in a three minutes long time window. Next, based on these measurements, we decided whether the values are produced by an actual correlation. We achieved this by fitting different continuous probability distribution functions onto the measured correlation values. When we found a proper fit, we added the signal pair to our model. For

every signal pair, we also calculated four thresholds to identify the boundaries of normal behaviour: (1) two thresholds define a narrow normal interval, such that measurement outside of this interval are considered potential anomalies for further analysis; (2) and another two thresholds define a wider interval, such that measurements outside of this interval are considered anomalies immediately.

In the detection phase, correlation values are determined in both a one minute long and a three minutes long window. Then the measured values are compared to the previously defined threshold for anomaly detection.

3.3 Data preprocessing

In the training and testing phases we used a 31 minutes long CAN traffic log captured from a middle class vehicle. The traffic contains both periodic and non-periodic messages. This means that some messages arrive regularly with a fix repetition times, while others are only transmitted upon specific events. Before the training, we filtered out the non-periodic messages and those periodic messages that appear less than once per minute. After the filtering step, 92% of the original data remained.

The next step was the signal extraction from the traffic log. For this we used an algorithm from the Automated CAN Payload Reverse Engineering[3] project. This algorithm separates the bits of the CAN data field into signals based on bit flip frequencies, called Transition Aggregation N-Grams. The method builds on the property that the MSB bits of a signal change less frequently than the LSB bits. If there is a significant change in the bit flip frequencies of two neighboring bits that shows the boundary between two signals. The same statistical information can also be used to determine the signal encoding.

We calculated correlation values pairwise for the identified signals in different time windows. We tested window sizes from 1 to 8 minutes. For every interval, we re-sampled the signals to have two signal values per second within the chosen window. This rarefying speeds up the correlation calculation. Our measurements showed that the best results can be achieved by choosing 1 minute and 3 minutes as final time windows. This allows us to detect significant anomalies fast and smaller anomalies in a reasonable time.

3.4 Model training

Five matrices are calculated for both time windows for the purpose of anomaly detection. A matrix C contains the correlation values and there are four additional threshold matrices that store two upper ($C_{th+,1}$ $C_{th+,2}$) and two lower thresholds ($C_{th-,1}$ $C_{th-,2}$) for the detection.

Each cell $c_{i,j}$ in matrix C contains the Pearson correlation value calculated between signals i and j in the given time window. The correlation value is stored the following way:

[3] https://github.com/brent-stone/CAN_Reverse_Engineering

- if the calculated coefficient indicates constant values then a *NaN* value is stored.
- otherwise if the two-tailed p-value of the Pearson correlation coefficient is less than or equal to 0.05, then the calculated value is stored.
- otherwise a 0 is stored.

During the training phase, we randomly select a starting point in the CAN log and calculate the correlation values for all signal pairs for both time intervals. Selecting the starting point randomly allows us to use the original trace multiple times generating a correlation matrix with small differences for every starting point. With this method, we created 300 training matrices for the threshold calculations.

These training matrices gave us a good representation of the typical correlation value for all pairs of signals. In order to find thresholds characterizing the normal behaviour, we fitted different continuous probability distribution functions onto the $c_{i,j}$ values of every training matrix. For every distribution, we performed a Kolmogorov-Smirnov (K-S) test to find the distribution that fits best the correlation values. The K-S test gave us two results: a D statistics and a p-value. For the former, we calculated[4] a critical value at significance level $\alpha = 1\%$ using (1) (where n is the number of samples in the dataset):

$$d_{1\%} = \frac{1.6276}{\sqrt{n}} \tag{1}$$

Those distributions, where the resulting D statistic was less than or equal to the critical value and the resulting fitted probability distribution's standard deviation was greater than 0 and less than 0.2, we accepted the distribution as a potential candidate. Then, for all these candidates, we calculated the probability distribution's percent-point (or quantile) function value for the $10^{-3}, 1-10^{-3}, 10^{-6}, 1-10^{-6}$ probabilities. These gave us candidates for the minimum ($min_{i,j}$), maximum ($max_{i,j}$), significant minimum ($sigmin_{i,j}$) and significant maximum ($sigmax_{i,j}$) thresholds.

To choose the best option from the candidates, a scoring technique was used, which is based on the length of the normalized significant min-max interval ($sigmax_{i,j} - sigmin_{i,j}$) and the normalized min-max intervals ($max_{i,j} - min_{i,j}$). The candidate with the minimum final score was selected as the final probability distribution. We used the following function (Figure 1 and Equation 2) with a minimum value of 0.6 for the significant min-max, and a minimum value of 0.5 for the min-max intervals:

$$score(x) = \begin{cases} (x - min)^2, \text{if } x \leqq min \\ \frac{x - min}{min * x}, \text{if } x > min \end{cases} \tag{2}$$

Then, for the final candidate score we used formula (3):

[4] https://www.real-statistics.com/statistics-tables/kolmogorov-smirnov-table/

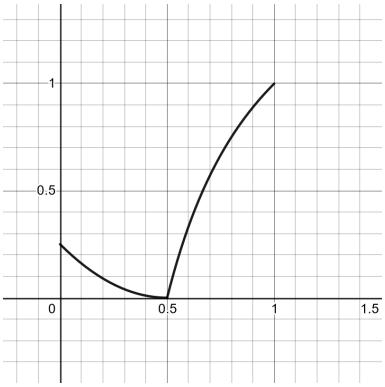

Fig. 1. Visualization of the function used to score the normal min-max threshold intervals.

$$final\_score = 0.65 * minmax\_score + 0.35 * significant\_minmax\_score \quad (3)$$

The used scoring system assigns the smallest score to the candidates with a min-max interval length closest to 0.5 and a significant min-max interval length closest to 0.6. This approach prefers the candidates where the intervals are relatively small but not too tight. Our measurements showed that these typically used statistical constants in the calculations with this weighting gives the best trade-off between false positive and false negative results. We used a larger weight for the min-max score to reflect that it is more important to detect and attack the speed of the detection.

These final chosen threshold values are stored in the threshold matrices, given that there was at least one candidate distribution, in the following way: the minimum values are stored in matrix $C_{th-,1}$; the maximum values are stored in matrix $C_{th+,1}$; the significant minimum values are stored in matrix $C_{th-,2}$, and finally, the significant maximum values are stored in matrix $C_{th+,2}$. If there was no candidate probability distribution found, the signal pair was excluded from the study.

In Figure 2, we present an example of the measured correlation values of a signal pair and the fitted probability distribution function ('loggamma'). The vertical blue and red lines show the determined minimum and maximum, and significant minimum and maximum values, respectively.

3.5 Detection

In the detection phase, the current correlation values are calculated for the last 1 and 3 minutes of the network traffic, and then, the results are compared to the threshold matrices in the following way:

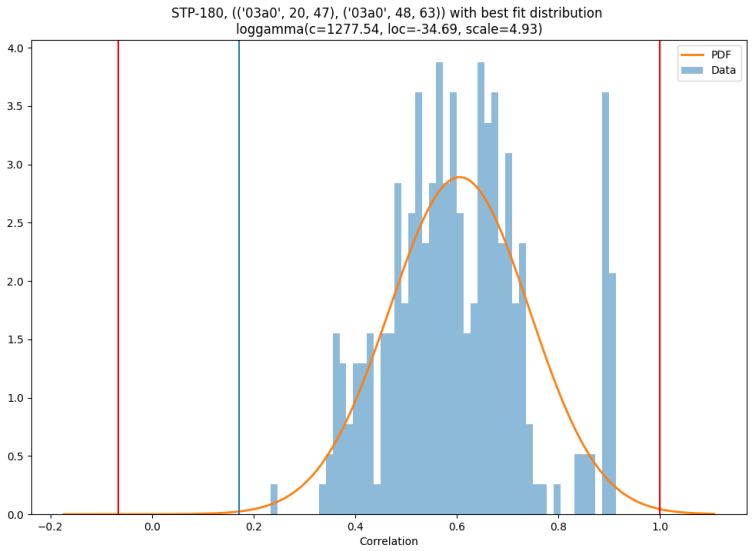

Fig. 2. One example of a signal pair with a fitted 'loggamma' probability distribution

- if a correlation value is outside the significant min-max interval, it is immediately detected as an attack;
- if the value is only outside the normal min-max interval, but not outside the significant one, we consider it a potential attack only, which will be a real attack if our model with the other time interval also gets a similar result.

4 Evaluation of the algorithm

4.1 Testing

In the testing phase, we took 270 1-minute long samples of the original trace, which we also used for training. We performed the detection step of the algorithm on these samples with a previously trained model and found the following result: the model signaled an attack in 14 out of 270 cases, resulting in a false positive rate of 5.2%.

4.2 Validation

Infected dataset The validation of the proposed algorithm was performed on an infected dataset. We simulated different message modification attacks (no new message is added to the log) with a previously developed attack simulator[5]

[5] https://github.com/CrySyS/can-log-infector

that can take a clean CAN log and modify a selected subset (specified by ID and time interval) of its messages according to 7 different attack scenarios:

1. **const:** the original data value is replaced by a given attack data.
2. **random:** the original data value is replaced by a new random value.
3. **delta:** a given attack data is added to the original data value.
4. **add_incr:** an increasing value is added to the original data value.
5. **add_decr:** an increasing value is subtracted from the original value.
6. **change_incr:** the original data value is replaced by an increasing value.
7. **change_decr:** the original data value is replaced by a decreasing value.

Measurements In order to evaluate the performance of the algorithm in more details, we divided the signals into three different groups and validated the algorithm in each group separately. The first group contain signals that strongly correlate with multiple other signals. Typically, the most important signals of a vehicle belong to this group. The second group contains signals that have a strong correlation with one other signal, and the third group contains signals with only weak correlation values. We considered a correlation strong between two signals if the mean of the absolute correlation value was above or equal to 0.9 for all the 300 training data samples.

We chose from each group 4 or 5 signal pairs for the validation. For these signals, we simulated all previously mentioned 7 attacks on 15 randomly chosen segments of the original trace. Each attack was performed multiple times. First, only 8 bits was modified according to the attack description in one of the target signals, than the number of affected bits in the upcoming test was increased by 4 until the signal length was reached. All of the attacks were theoretical, but based on previous real life attack descriptions. We did not check whether a specific attack would actually have any impact in real life.

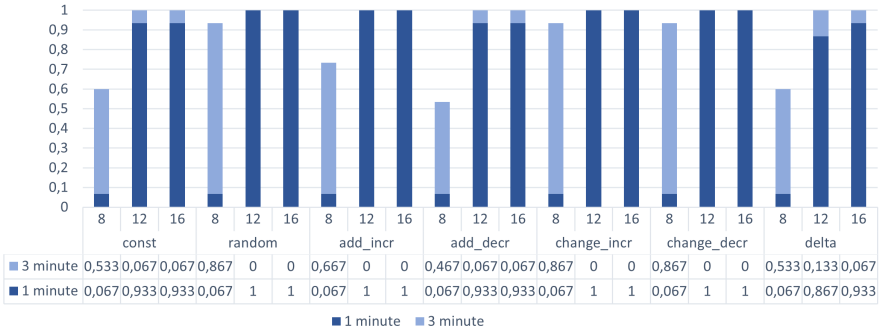

Fig. 3. Testing results for 16 bit long signal with strong correlations.

Figure 3 shows detailed results for a signal with strong correlations. The 16 bit long signal was attacked with all attack types. For each type, 3 attacks were

performed where the affected number of bits increase from 8 to 16. The two colors of the columns indicate which time window was successful for the attack detection. The detection rate varies between 55% and 100% with an above 90% result for attacks modifying more than 12 bits.

The results found in the others groups, as expected, are less accurate. The average detection accuracy of attacks of signals with one strong correlation is 58% while this falls to 20% for the third group where the signals only have weak correlations.

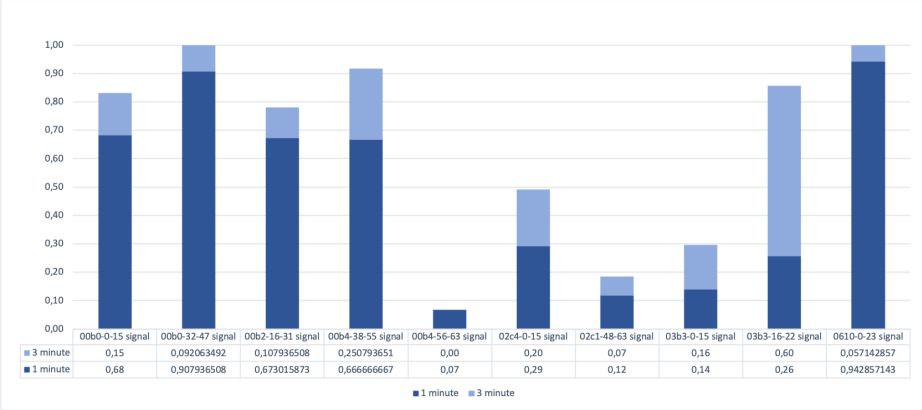

Fig. 4. Detection accuracy of high priority signals.

Figure 4 shows a summary of the results for messages with the highest priority (based on the CAN ID field). It can be seen, that most messages with the highest priority contain signals with high correlation, making them ideal candidates for a correlation based anomaly detection.

5 Conclusion and future work

In this paper, we proposed a novel correlation based anomaly detection method for the CAN bus with a focus on message modification attacks. We showed, that our solution efficiently detects most of the attacks, making it a promising candidate for real life anomaly detection. Furthermore, a significant advantage of this correlation based detection is that it can detect even the most sophisticated attacks, assuming that the attacker does not modify every related signals consistently.

In our future work, we plan to investigate if the proposed threshold based detection mechanism can be replaced with other potential solutions that increase accuracy. A potential option for this is a machine learning based classification. Moreover, the efficiency of the correlation calculation could also be increased

with better data preprocessing, in order to further improve the applicability of our solution in real life scenarios.

References

1. Ben Othmane, L., Dhulipala, L., Abdelkhalek, M., Multari, N., Govindarasu, M.: On the performance of detecting injection of fabricated messages into the can bus. IEEE Transactions on Dependable and Secure Computing pp. 1–1 (2020). https://doi.org/10.1109/TDSC.2020.2990192
2. Choi, W., Joo, K., Jo, H.J., Park, M.C., Lee, D.H.: VoltageIDS: Low-level communication characteristics for automotive intrusion detection system. IEEE Transactions on Information Forensics and Security **13**(8), 2114–2129 (2018). https://doi.org/10.1109/TIFS.2018.2812149
3. Gazdag, A., Ferenczi, C., Buttyán, L.: Development of a man-in-the-middle attack device for the can bus. In: Proceedings of the 1st Conference on Information Technology and Data Science Debrecen, Hungary, November 6–8, 2020. pp. 115–130 (2020)
4. Gmiden, M., Mohamed, H., Trabelsi, H.: An intrusion detection method for securing in-vehicle CAN bus. In: Proceedings of the 17th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA) (2016). https://doi.org/10.1109/STA.2016.7952095
5. Groza, B., Murvay, S., van Herrewege, A., Verbauwhede, I.: LiBrA-CAN: a lightweight broadcast authentication protocol for Controller Area Networks. In: Proceedings of the International Conference on Cryptology and Network Security (CANS). pp. 185 – 200 (2012)
6. Ji, H., Wang, Y., Qin, H., Wu, X., Yu, G.: Investigating the effects of attack detection for in-vehicle networks based on clock drift of ECUs. IEEE Access **6**, 49375–49384 (2018). https://doi.org/10.1109/ACCESS.2018.2841884
7. Kang, M.J., Kang, J.W.: Intrusion detection system using deep neural network for in-vehicle network security. PLoS One **11**(6) (2016)
8. Kang, M.J., Kang, J.W.: A novel intrusion detection method using deep neural network for in-vehicle network security. In: Proceedings of the 83rd IEEE Vehicular Technology Conference (VTC Spring). pp. 1–5 (2016). https://doi.org/10.1109/VTCSpring.2016.7504089
9. Kim, K., Kim, J.S., Jeong, S., Park, J.H., Kim, H.K.: Cybersecurity for autonomous vehicles: Review of attacks and defense. Computers & Security **103** (2021). https://doi.org/10.1016/j.cose.2020.102150
10. Koscher, K., Czeskis, A., Roesner, F., Patel, S., Kohno, T., Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S.: Experimental security analysis of a modern automobile. In: 2010 IEEE Symposium on Security and Privacy. pp. 447–462 (2010). https://doi.org/10.1109/SP.2010.34
11. Lokman, S.F., Othman, A.T., Abu-Bakar, M.H.: Intrusion detection system for automotive Controller Area Network (CAN) bus system: a review. EURASIP Journal on Wireless Communications and Networking **184** (2019). https://doi.org/10.1186/s13638-019-1484-3
12. Marchetti, M., Stabili, D.: Anomaly detection of can bus messages through analysis of id sequences. In: Proceedings of the IEEE Intelligent Vehicles Symposium (IV). pp. 1577–1583 (2017). https://doi.org/10.1109/IVS.2017.7995934

13. Markovitz, M., Wool, A.: Field classification, modeling and anomaly detection in unknown CAN bus networks. In: Proceedings of the Embedded Security in CARs (ESCAR) Conference (2015)
14. Matsumoto, T., Hata, M., Tanabe, M., Yoshioka, K., Oishi, K.: A method of preventing unauthorized data transmission in Controller Area Network. In: Proceedings of the 75th IEEE Vehicular Technology Conference (VTC Spring). pp. 1–5 (2012). https://doi.org/10.1109/VETECS.2012.6240294
15. Miller, C., Valasek, C.: A survey of remote automotive attack surfaces. Tech. rep., IOActive (2014), https://ioactive.com/pdfs/IOActive_Remote_Attack_Surfaces.pdf
16. Miller, C., Valasek, C.: Adventures in automotive networks and control units. Tech. rep., IOActive (2013)
17. Moore, M.R., Bridges, R.A., Combs, F.L., Starr, M.S., Prowell, S.J.: Modeling inter-signal arrival times for accurate detection of CAN bus signal injection attacks. In: Proceedings of the 12th Annual Conference on Cyber and Information Security Research (2017)
18. Nürnberger, S., Rossow, C.: vatiCAN – vetted, authenticated CAN bus. In: Proceedings of the International Conference on Cryptographic Hardware and Embedded Systems (CHES). pp. 106 – 124 (2016)
19. Sharma, C., Moylan, S., Amariucai, G.T., Vasserman, E.Y.: An extended survey on vehicle security. Computing Research Repository (CoRR) **abs/1910.04150** (2019), http://arxiv.org/abs/1910.04150
20. Song, H.M., Kim, H.R., Kim, H.K.: Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network. In: Proceedings of the International Conference on Information Networking (ICOIN) (2016). https://doi.org/10.1109/ICOIN.2016.7427089
21. Taylor, A., Japkowicz, N., Leblanc, S.: Frequency-based anomaly detection for the automotive CAN bus. In: Proceedings of the World Congress on Industrial Control System Security (WCICSS) (December 2015)
22. Taylor, A., Leblanc, S., Japkowicz, N.: Anomaly detection in automobile control network data with long short-term memory networks. In: Proceedings of the IEEE International Conference on Data Science and Advanced Analytics (DSAA). pp. 130–139 (2016). https://doi.org/10.1109/DSAA.2016.20
23. Theissler, A.: Anomaly detection in recordings from in-vehicle networks. In: Proceedings of the International Workshop on Big Data Applications and Principles (2014)
24. Tomlinson, A., Bryans, J., Shaikh, S.A., Kalutarage, H.K.: Detection of automotive CAN cyber-attacks by identifying packet timing anomalies in time windows. In: Proceedings of the 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W). pp. 231–238 (2018). https://doi.org/10.1109/DSN-W.2018.00069
25. Van Herrewege, A., Singelée, D., Verbauwhede, I.: CANAuth – a simple, backward compatible broadcast authentication protocol for CAN bus. In: Proceedings of the ESCAR Conference (2011)
26. Wu, W., Li, R., Xie, G., An, J., Bai, Y., Zhou, J., Li, K.: A survey of intrusion detection for in-vehicle networks. IEEE Transactions on Intelligent Transport Systems **21**(3) (March 2020). https://doi.org/10.1109/TITS.2019.2908074
27. Young, C., Zambreno, J., Olufowobi, H., Bloom, G.: Survey of automotive controller area network intrusion-detection systems. IEEE Design & Test (October 2019). https://doi.org/10.1109/MDAT.2019.2899062