Budapest University of Technology and Economics
Department of Networked Systems and Services

New Methods for Security and Privacy of CAN Bus Communication

Collection of Ph.D. Theses
of
András Gazdag

Supervisor: Levente Buttyán, PhD, DSc

www.crysys.hu

Budapest, Hungary
2023

1 Introduction

The last decades of the automotive industry have seen a significant change with the adoption of embedded controllers. Digital circuits and software components have taken control of processes previously controlled by analog methods. In addition to supporting more integrated functions and services, the primary motivation for this change was to reduce manufacturing costs. While delivering the expected results, this shift also created an undesirable problem: vehicles inherited the cybersecurity weaknesses of computers.

The emergence of cyber-physical systems (including vehicles) has brought new threats. If an attacker can take control of a computer-controlled process in an attack, it can cause physical damage. In the transportation industry, such an attack could endanger human lives or cause significant financial loss.

Fortunately, we are not yet aware any such attack happened in real life. However, the seriousness of the problem is illustrated by the millions of vehicles that manufacturers have recalled on several occasions to repair vulnerabilities found by security experts at considerable cost. The importance of the problem is further shown by the fact that several new regulations and standards, such as ISO/SAE 21434:2021 or UN Regulation No. 155, have been introduced in the European Union and elsewhere in the world to make cyber security a priority for new vehicles. Under UN Regulation No. 155, from 2024, only new vehicle types that meet cybersecurity requirements will be type-approved.

Researchers have also been working for years to address emerging cybersecurity issues. In my dissertation, I also address some of these challenges. I examine the entire cybersecurity problem set from several perspectives. I have conducted research on supporting long-term efficient traffic log storage and on attack detection. In addition to security threats, I have also worked on the privacy issues of vehicular data release.

A specific feature of cyber attacks is that there can be a significant time lag between the execution of an attack and its discovery or the signs of the damage it causes. With this in mind, the first area where I have achieved new results relates to recording data on the internal network of vehicles. I propose a new compression algorithm to facilitate efficient data storage over a long period, thus allowing its analysis long after the attack. My proposed method achieves significantly better compression ratio than widely used alternative compression methods.

I have also achieved new results in attack detection. First, I show that the previously described compression algorithm is suitable for detecting message injection attacks. This result further enhances the value of the compression method, as it allows us to investigate a subset of attacks on compressed data, saving significant computational resources. My following proposed detection method exploits correlations between transmitted signals and proves that attacks can be efficiently detected using models built on correlations. Finally, I propose a third detection method that can be applied to signals individually. The method uses machine learning to predict for each signal an expected future value. Measured values can be continuously compared to their prediction to identify any unexpected change caused by an attack.

The dissertation's final chapter focuses on the privacy issues of CAN data release. I show that the data transmitted over the bus can also be used to reconstruct the movement of vehicles for short and long distances. This result supports the assumption that the captured data can only be used with due care in order to avoid legal and ethical problems.

This document is a summary of the results that are discussed in details in my dissertation.

2 New Results

2.1 Semantic compression of CAN traffic

Historical analysis of CAN bus traffic logs, is only possible if data storage is solved efficiently. There are two possible approaches: (1) storing traffic logs locally or (2) offloading captured traffic to a remote server. Whichever option the manufacturer chooses, traffic compression can significantly improve the efficiency of the process. In this dissertation, I propose a compression method that allows for the lossless, yet efficient storage of data. I achieve this by performing semantic compression on the CAN traffic logs, rather than simple syntactic compression. The compression ratio that I achieve is better than the compression ratio of the state-of-the-art syntactic compression methods, such as zip.

THESIS 1.1: I proposed a semantic compression method for compressing CAN traffic and measured that this alone compresses data to 10% of the original size in [C2]. I showed that combining semantic and syntactic compression can reduce the required storage space to 5% of the original size in [J1]. This approach thus provides a significantly more efficient result than syntactic compression alone, which only reduces the size to 30% of the original.

I propose a compression algorithm that takes advantage of the largely periodic nature of the CAN traffic. The high level approach of my algorithm is to separate the traffic into message flows, containing only messages that have the same ID, and then, compressing each message flow separately leveraging the regular repetition times and repeating data contents of the communication. Algorithm 1 shows the pseudo code of my compression algorithm.

I defined two output formats for my algorithm. One is a text-based (ASCII) representation of the traffic log, while the other is a binary format. Both formats contain the same lossless information.

My algorithm significantly outperforms state-of-the-art syntactic compression methods (see Table 1 and Table 2). I was able to achieve compression ratios of less than 20% using an ASCII representation of the output of my algorithm. The binary representation shows an even more efficient compression with the results being around 10% of the original file size.

If I applied, as a hybrid approach, an additional syntactic compression to my semantic compression it resulted in the smallest file sizes I was able to achieve. In the ASCII representation scenario the combined result shows an approximate 6% compression ratio while the binary case shows an approximate 5% compression ratio.

```
Algorithm 1: Semantic compression
Input: raw CAN log
Output: compressed CAN log
messages ← read CAN traffic log;
flows ← separate Messages into message groups;
for flow in flows do
    calculate_average_inter_arrival_time(flow);
    group_messages_with_identical_data(flow);
    for message in flow.messages do
        compress_timestamp(message);
for flow in flows do
    write_compressed_flow_to_output(flow);
```

Table 1: Semantic compression ratio comparisons

Test case	Original trace	Text format		Binary format	
	file size	file size	file size	file size	file size
	(bytes)	(bytes)	percentage	(bytes)	percentage
1	10 095 971	1 710 920	16,94%	1 090 757	10,80%
2	7 040 165	1 334 902	18,96%	835 539	11,86%
3	19 143 383	3 747 229	19,57%	2 307 146	12,05%
4	21 936 245	4 233 994	19,30%	2 601 354	11,85%

Table 2: Semantic and Syntactic compression ratio comparisons

Test case	Original trace	Semantic and Syntactic compression combined			
	zip compressed	Text format		Binary format	
	file size	file size	file size	file size	file size
	(bytes)	(bytes)	percentage	(bytes)	percentage
1	1 291 315	546 725	5,41%	499 998	4,95%
2	937 319	429 234	6,09%	390 467	5,54%
3	2 569 118	1 194 758	6,24%	1 092 183	5,70%
4	2 895 039	1 332 585	6,07%	1 223 677	5,57%

2.2 Attack detection in compressed CAN traffic

In order to support forensics investigations in vehicles, CAN traffic must be logged continuously and stored efficiently for later analysis. My contribution to support this effort is a novel anomaly detection method to identify message injection attacks that works on compressed CAN traffic logs. The advantage of running anomaly detection on the compressed logs is that smaller amount of data needs to be analyzed, hence, the efficiency of forensic investigations can be increased.

THESIS 1.2: To support forensics analysis, I showed through measurements on two datasets that the compressed format is suitable for high-confidence identification of message injection attacks in [C4][1]. The previous results show that to implement a successful message injection attack, at least five times the normal message frequency is required during the attack. The proposed detection solution, on the other hand, can detect the anomaly from as low as twice the normal frequency.

My anomaly detection algorithm is based on analyzing the average frequencies of messages with given CAN IDs. The compression algorithm, presented in Section 2.1, preserves the number of messages per unit time in an easily analyzable form in the compressed CAN log, which makes it possible to use my anomaly detection algorithm on the compressed logs. I demonstrated that this approach works reliably in a range of scenarios, including using data sets captured in real vehicles and modified with synthetically generated attacks as well as data sets captured in real vehicles under real attacks.

On synthetic data I used 100-100 normal and attacked samples for attacks with different frequency. The histogram of the distribution of the attacks can be seen in Figure 1. It demonstrates that the attacked traffic is efficiently distinguishable from the normal traffic even when the attack frequency is as low as 2 times of the original.

On the data from the real world attacks I performed the same calculations. Figure 2 shows that my algorithm achieves the same reliable results in the real life scenarios.

[1] Dóra Neubrandt implemented the measurement algorithm.

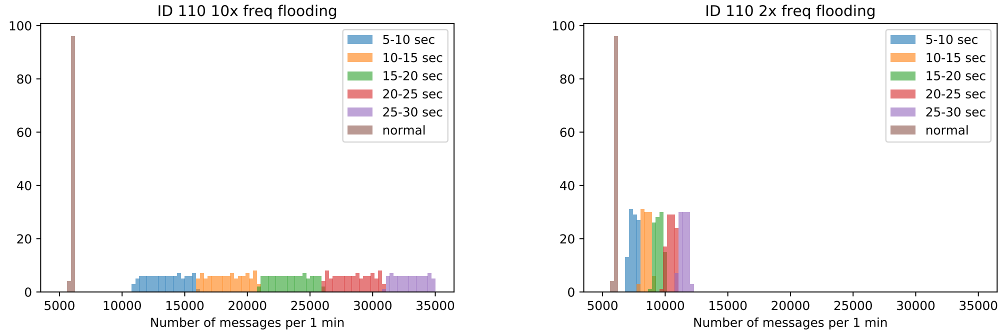


Figure 1: Comparison of the number of messages feature for 100 - 100 benign and synthetically attacked samples.

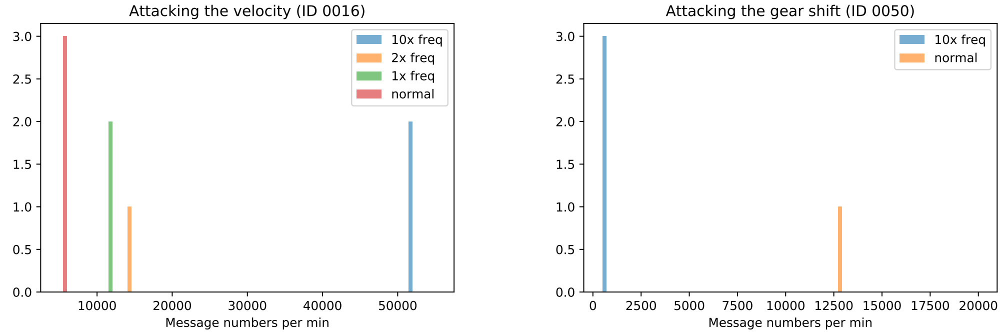


Figure 2: Comparison of the number of messages in normal and attacked scenarios during real attacks.

2.3 Correlation-based anomaly detection

While the majority of attacks on the CAN bus relies on message injection[5, 8], this is not the only technique to achieve malicious goals. The predictability of message ID frequencies alone is not sufficient for detecting attacks that do not inject new messages on the CAN bus, also called message modification attacks.

I 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.

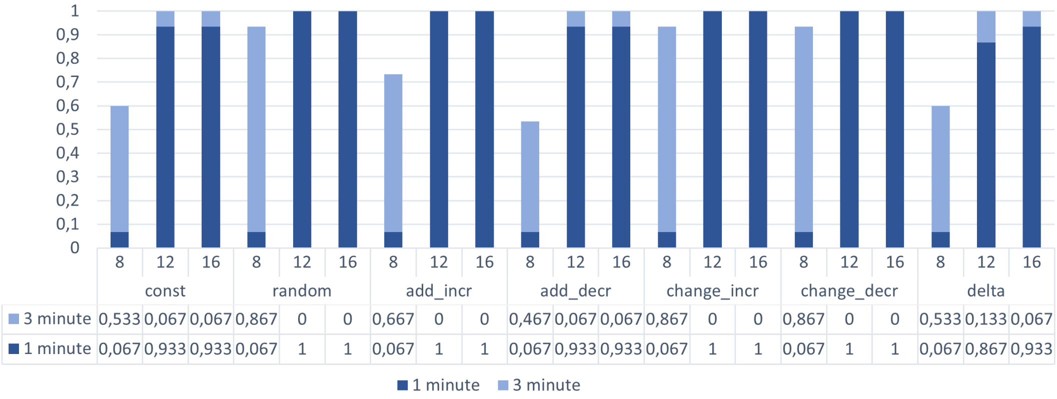
THESIS 2.1: I showed in [C6][2] that our model, built on the correlation measurements between CAN signals, can successfully detect message modification attacks. I tested the accuracy of the proposed method against seven different attack strategies. The results show that for attacks targetting signals strongly correlating with other signals, the accuracy of our detection is ∼90% with a 0% false positive rate due to applying a double threshold system. It is worth highlighting that for the RANDOM, ADD-INCR, and ADD-DECR attacks modifying at least 8 bits of a signal, we were able to achieve 95% accuracy. Similarly, for all attacks modifying at least 12 bits of a signal, the detection accuracy is 95%.

In the training phase, the correlation values between signals has to be determined. I 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, I decided whether the values are produced by an actual correlation. I achieved this by fitting different continuous probability distribution functions onto the measured correlation values. When I found a proper fit, I added the signal pair to my model. For every signal pair, I 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.

[2] György Lupták implemented the correlation calculation and statistical testing.

In order to evaluate the performance of the algorithm in more details, I divided the CAN 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.



	const			random			add_incr			add_decr			change_incr			change_decr			delta		
	8	12	16	8	12	16	8	12	16	8	12	16	8	12	16	8	12	16	8	12	16
■ 3 minute	0,533	0,067	0,067	0,867	0	0	0,667	0	0	0,467	0,067	0,067	0,867	0	0	0,867	0	0	0,533	0,133	0,067
■ 1 minute	0,067	0,933	0,933	0,067	1	1	0,067	1	1	0,067	0,933	0,933	0,067	1	1	0,067	1	1	0,067	0,867	0,933

Figure 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.

2.4 Signal anomaly detection with TCN

Detecting message modification attacks is a difficult task. Building any model of the CAN traffic based only on the message data is particularly challenging, as one does not know how to interpret the data[7]; therefore, one cannot exploit any semantic information. As I showed in Section 2.3, exploiting data correlations between messages can be a powerful detection mechanism. However, not every signal correlates strongly with others, so that approach is limited. In this section, I propose a new detection method that works on a signal-by-signal base to supplement my previous solutions.

THESIS 2.2: I proposed a TCN-based detection model that can detect CAN message modification attacks by predicting future values to CAN signals and then comparing the prediction with the actual values in [C7][3]. Based on measurements from two datasets, I demonstrated that my TCN-based detection method detects attacks with an accuracy between 83% and 99% while keeping a false positive rate below 0.2%. I compared the proposed method to the previously best-performing solution and showed that my detection algorithm performs better in 27 out of 30 cases.

I propose a TCN-based approach for detecting modified CAN bus messages. I construct and train the TCN in an unsupervised fashion, since, in practice, labelling CAN bus messages is a difficult task. In the training process, the TCN will learn to accurately reconstruct the individual signals of CAN bus messages through its causal convolution layers, which allows for information retention from past data samples. Finally, the classification of new data samples will resume to setting an appropriate threshold on their reconstruction loss value. The core idea here is that signals whose data have been altered will be poorly reconstructed by the model, and thus be easy to recognize. Note, that it is not a prerequisite for me to know CAN bus signal semantics which is usually kept confidential [7].

Table 3: Accuracy of the models on SynCAN dataset.

Model	Data	Normal	Cont.	Playb.	Flood.	Suppress	Plateau
TCN	ID 2	**0.9977**	**0.8660**	**0.8674**	**0.7678**	**0.8402**	**0.8336**
INDRA		0.9811	0.8584	0.8660	0.7600	0.8347	0.8133
TCN	ID 3	**0.9992**	**0.8664**	**0.8680**	**0.6422**	**0.8390**	**0.8394**
INDRA		0.9965	0.8653	0.8672	0.6420	0.8377	0.8386
TCN	ID 10	**0.9977**	**0.8637**	0.8577	0.7399	**0.8446**	**0.8282**
INDRA		0.9858	0.8546	**0.8638**	**0.7923**	0.8370	0.8100

[3] Irina Chiscop implemented the TCN network architecture.

Table 4: Results for the CrySyS dataset.

Model	Data	Acc.	FPR	Precision
TCN	ID 280	**0.8833**	0.0426	**0.7766**
INDRA		0.7989	**0.0000**	0.0000
TCN	ID 290	**0.9159**	0.0687	0.7701
INDRA		0.8617	**0.0378**	**0.7755**

To evaluate the performance of my proposed model, I compared its performance to the previously best-performing result from the literature. To the best of my knowledge, the most recent and suitable candidate is the INDRA framework [6]. It proposes a recurrent autoencoder network that is able to detect CAN messages in which signals have been tampered with. For each message ID one such recurrent autoencoder is trained such that it learns to reconstruct the signals within that particular message ID. This approach is shown to outperform other recent unsupervised methods such as Predictor LSTM [9], Replicator Neural Network [10], and CANet [4], on most attack classes of the SynCAN dataset, in terms of accuracy and false positive rate.

I first assessed the performance of my model and the INDRA model on the SynCAN dataset. The accuracy values, calculated for the normal test set and for each attack class, are shown in Table 3. A first observation is that TCN achieves a higher accuracy than INDRA in most cases, with the exception of playback and flooding attacks on ID 10. Moreover, the false positive rates are quite low for both models. Overall, there are large variations in the precision values across different message IDs which may be related to how the attacks were performed (target signals chosen, attack duration, etc.) and the different signal correlations. Also, the relatively low precision values show that the models manage to capture only a limited set of temporal characteristics of the SynCAN data. This is a direct consequence of the stopping mechanism implemented during training, and in the case of TCN, of the choices made to keep a lightweight architecture.

The message IDs in the SynCAN dataset contains signals that are physically interdependent, but are very weakly correlated; this also increases the difficulty of the detection task. In order to assess how the two models perform in a different setting, I evaluated the models on two message IDs of the CrySyS dataset which contains more signals with a strong correlation. Here, similarly to SynCAN, only one signal was attacked. The results are shown in Table 4. I notice that both models still achieve high accuracy and a low false positive rate, with TCN showing a high precision for both attacks, as opposed to INDRA, failing to detect the attack in message 280.

In conclusion, the simple TCN architecture achieves a slightly better accuracy compared to the INDRA model on both datasets. A remarkable achievement of TCN is the significant reduction of false positives (by a factor of 10) in nearly all cases: this translates to a more reliable detector in practice. Further advantages of the TCN are that it is quick to train, has a much smaller resource need, and achieves in general lower training and validation loss.

2.5 Privacy threat: macrotracking

Data is constantly being generated in vehicles. In cars with more and more controllers, the signal values measured while driving provide more detailed information about the vehicle and its driver. I have argued in previous sections that storing and processing this data is an important task in the future for identifying attacks. However, one can also realize that the same data can have a different value: it can be used in data-driven services. Some of those new services may raise concerns about the protection of personal data [1]. The manufacturer could use the data available in the vehicle to recommend services for the owner continuously. Furthermore, many companies could use the behavioral and location information of the driver to offer other services. Such use of data is only acceptable if the relevant laws are obeyed in the process.

In this section, I show that by releasing CAN data, vehicles and thus drivers become traceable. The tracing of the vehicle is achieved in two steps. In my dissertation, first, I show how a vehicle can be traced accurately over short distances based exclusively on CAN messages (microtracking). Second, I show how to extend tracing for longer trips using additional, publicly available information (macrotracking).

THESIS 3.1: I proposed a (macrotracking) algorithm that can reliably reconstruct the trajectory of a vehicle over longer trips only from raw CAN data and publicly available map information in [J2]. I have verified the method's accuracy with measurements: the algorithm was able to reconstruct all several kilometers-long test cases, consisting of at least 20 intersections, with just a few meters of inaccuracy.

To achieve the goal, I use some auxiliary information in order to mitigate the problem of error accumulation encountered in microtracking. The speed and steering wheel angle values extracted from the CAN messages are required for the reconstruction. Additionally, the starting position and the initial heading are also a prerequisite for my algorithm. Provided with these input data, I show that the trajectory of a vehicle can be effectively reconstructed revealing the destination of the drive, which constitutes a privacy breach with respect to the driver. This implies that CAN logs have to be handled or processed carefully to avoid this privacy issue and comply with data protection regulations.

The pseudo code to reconstruct the movement of a vehicle is presented in Algorithm 2. First, the next state is always predicted from the previous state with model-based prediction as in microtracking (Line 5-8), and then map-based correction (Line 9-14) is only performed if the distance from the last correction is sufficiently large (Line 10 in Alg. 2).

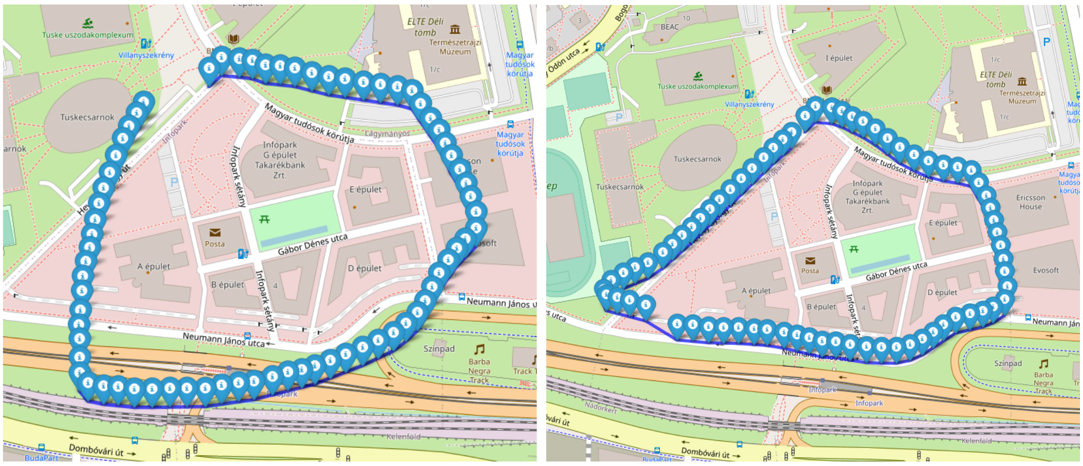
The accuracy of my algorithm depends on the correctness of model-based prediction and the density of the road network. On one hand, areas with many intersections does not allow the map based corrections to improve the model prediction as much, therefore the reconstruction error will dominate over longer distances. On the other hand, if the trajectory of the drive follows long sections without intersections, my algorithm will hardly suffer from any errors.

```
Algorithm 2: Macrotracking for CAN logs
Input: starting position and heading value, CAN log
Output: Reconstructed trajectory $\mathbb{T}$
initialize current state to starting position and heading;
load data from CAN log;
filter relevant messages;
while there is message to process do
    Model-based prediction:
        extract speed and steering wheel position from messages;
        compute heading from axle distance and steering wheel position;
        calculate next state from current state using heading and speed;
    Map-based correction:
        if distance from last correction > minimum required then
            find nearest road segment on map;
            project current position and heading to selected road segment;
            update map weight $w$ based on distance from closest intersection;
            update next state using the projected state with map weight $w$;
    append next state to reconstructed trajectory $\mathbb{T}$;
    update current state to next state;
```

Table 5 shows data about my test cases. In these tests, I drove along different circular trajectories to show the result of my algorithm. For example, during the C3 test case, I drove deliberatly with frequent movements of the steering wheel even on straight road segments to make the reconstruction harder. The corresponding row in the table shows that my algorithm was able to reconstruct the trajectory with only small errors in this case as well. Figure 4 shows the reconstructed trajectory.

Table 5: Summary of macrotracking test cases

Test case		Average trajectory reconstruction error (meter)	Std. deviation of error (meter)	Endpoint reconstruction error (meter)	Total distance travelled (meter)	Number of decision points on map
C1	without map	30.2	25.13	9.3	2025.17	20
	with map	9.37	8.99	4.2	2039.11	20
C2	without map	39.37	34.02	35.58	2139.03	18
	with map	9.13	8.74	41.12	2158.48	18
C3	without map	55.04	36.07	82.17	1751.07	19
	with map	7.45	6.05	6.05	1817.81	19



(a) C3 reconstruction without map

(b) C3 reconstruction with map

Figure 4: C3 test trajectories

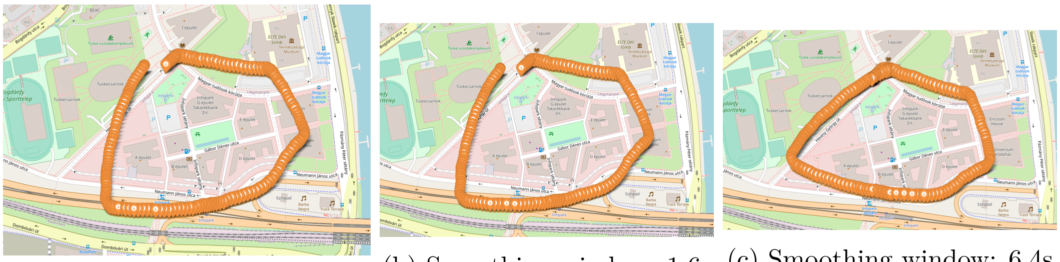
2.6 Mitigation options

Next, I considered some well established signal processing techniques including smoothing and low pass filtering which might be employed by a data controller or processor in order to distort CAN data so that unique features of traces are no longer recognizable. However, as I showed in my dissertation, such techniques fail to provide strong privacy guarantees, or introduce so much distortion to counter my attacks that renders the anonymized data practically useless.

THESIS 3.2: I have showed that the proposed macrotracking algorithm is robust to typical signal distortion techniques for protecting privacy in [J2]. The method's robustness has been verified by several measurements: even after applying a low-pass filter to achieve a 20% distortion, the inaccuracy remained below 8 meters. In the case of smoothing, the algorithm is even more robust: it accurately restored the original trajectory even after applying a 6.4s long smoothing window.

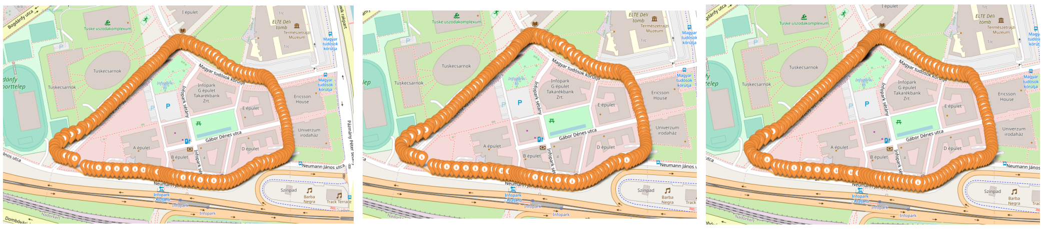
Smoothing

Smoothing is a type of downsampling technique. It is used to remove short-term fluctuations and highlight longer-term trends in the signal (or time-series). It has many variations, the main idea is to shift a fixed-size moving window through the signal and apply a transformation on each window, then publish the transformed signal. I apply a time-based moving window, i.e. the average of the data points that is in a fixed time frame (called *smoothing window*), whose size in time is given as a parameter w, is calculated and used as a single replacement of all points within the given time frame. As



(a) Smoothing window: 0.2s (b) Smoothing window: 1.6s (c) Smoothing window: 6.4s

Figure 5: C1 test case macrotracking result on smoothed data without map



(a) Smoothing window: 0.2s (b) Smoothing window: 1.6s (c) Smoothing window: 6.4s

Figure 6: C3 test case macrotracking result on smoothed data with map

the mean is reported per window, consecutive windows do not overlap. This technique is expected to smooth out local variations of every signal within w seconds. Therefore, as long as such local variations correspond to unique features of a trace, the transformed signal should mitigate tracking.

Figure 5 shows the effect of smoothing on the model-based trajectory reconstruction without map correction (i.e., microtracking) for the C3 case. Although smoothing negatively impacts reconstruction, applying smoothing even with the largest window size still allows a relatively accurate reconstruction of many parts of the trajectory. Interestingly, due to an encoding of the steering wheel value in the messages, applying smoothing on this signal has an inverse effect: it results in sharper turns than in the original trace.

My macrotracking algorithm with map correction is significantly more accurate than only model-based reconstruction and can successfully reconstruct the original traces even after smoothing is applied. Although the C3 test case is driven with frequent steering wheel changes, the effect of these changes are reduced by smoothing, and therefore the reconstruction results are actually improved with a larger window size (Figure 6). Table 6 contains the trajectory reconstruction errors with their standard deviation and the endpoint reconstruction error for all test cases with three different window sizes.

Table 6: Effects of smoothing on the location tracking algorithm.

Test case	Smoothing windows size (second)	Average trajectory reconstruction error (meter)	Std. deviation of error (meter)	Endpoint reconstruction error (meter)
C1	0.201	32.2	26.4	22.76
	1.608	33.51	26.97	33.02
	6.4	38.58	27.97	132.94
C2	0.201	37.75	28.68	75.74
	1.608	38.92	32.83	76.72
	6.4	42.22	32.47	126.84
C3	0.201	50.78	32.74	64.71
	1.608	47.53	29.6	66.52
	6.4	20.47	18.15	70.09

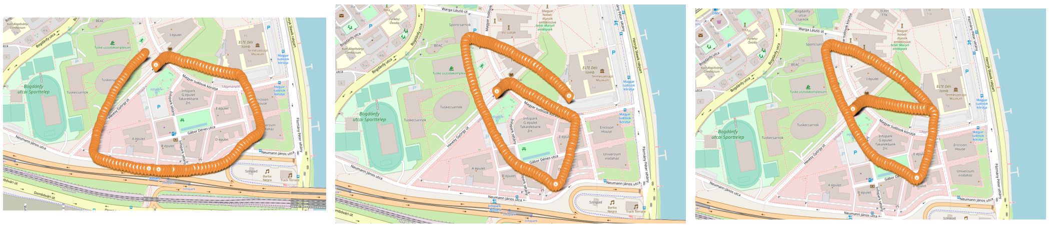
Low-pass filtering

Low pass filtering is a common technique not only to compress signals, but to reduce noise, eliminate aliasing, or attenuate resonances [3] without heavily degrading utility. Moreover, with the growing need for data privacy, low pass filtering has been used for signal anonymization as well [2]. Low-pass filters attenuate or eliminate all signal components above a specified frequency. By deleting these high frequency components, one can get rid of the idiosyncrasies of the signal and end up with the more general parts. Unlike smoothing, low pass filtering is expected to provide a finer-grained control over utility loss, and the mean squared error is precisely quantifiable since the transformation is orthonormal and therefore preserves the L_2-norm of the signal.

I apply low-pass filtering as follows. First, the signal is transformed to its frequency domain using orthonormal Discrete Cosine Transform (DCT). After DCT transformation, the number of removed high frequency components is determined. In general, the more components is dropped from the signal the lower the utility becomes. The resulting utility is measured by calculating the normalized euclidean distance between the original and the low passed signal, i.e. I delete as many of the highest frequency components as many needed to reach a predefined *error distance* (aka., reconstruction error) from the original signal. For example, in order to have a reconstruction error of 10% at most, the maximum number of the highest frequencies of the transformed signal are removed such that the L_2-norm of the removed components is not greater than the 10% of the total L_2-norm of the whole signal. Once the desired error rate is reached, the filtered signal is transformed back to the time domain and published.

Figure 7 shows the result of trajectory reconstruction without map (i.e., only model-based prediction) after low-pass filtering the C3 test case. In comparison with smoothing, low-pass filtering with the chosen parameters distort the original traces more significantly. The counter-intuitive changes of the turn angles can also be observed here.

Figure 8 shows that my macrotracking algorithm is capable of reconstructing the original C3 trajectory if the prescribed error rate of low pass filtering is below 40%. The accuracy of the reconstruction for all cases with different amount of low-pass filtering are depicted in Table 7. In summary, the reconstruction is prevented at a low pass filtering error of 40%, however, at this point the utility of the data is significantly impacted.

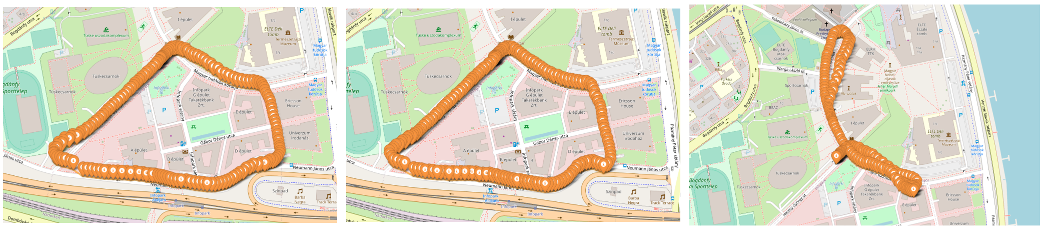


(a) Filtering: 10%

(b) Filtering: 20%

(c) Filtering: 40%

Figure 7: C3 test case macrotracking results on low pass filtered data without map



(a) Filtering: 10%

(b) Filtering: 20%

(c) Filtering: 40%

Figure 8: C3 test case macrotracking results on low pass filtered data with map

Table 7: Effects of low pass filtering on the location tracking algorithm.

Test case	Allowed reconstruction error	Average trajectory reconstruction error (meter)	Std. deviation of error (meter)	Endpoint reconstruction error (meter)
C1	10%	8.63	9.07	8.45
	20%	8.25	7.33	12.9
	40%	47.71	74.23	602.64
C2	10%	8.71	8.94	11.43
	20%	10.98	11.9	13.37
	40%	175.08	159.73	589.17
C3	10%	7.01	5.92	9.36
	20%	7.58	5.93	8.16
	40%	207.08	151.12	124.05

Acknowledgement

The research presented here received the financial support from the following projects, programmes, and funding agencies:

- National Research, Development and Innovation Office (NKFIH) of Hungary [4,5,6,7,8,9]
- ECSEL[10]
- Innovative Mobility Program of KTI[11]

[4] The research presented here have been supported by the NRDI Office, Ministry of Innovation and Technology, Hungary, within the framework of the Artificial Intelligence National Laboratory Programme, and the NRDI Fund based on the charter of bolster issued by the NRDI Office.

[5] The research presented here have 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.

[6] The work presented here was partially supported from the grant GINOP-2.1.1-15. The project has been supported by the European Union, co-financed by the European Social Fund. EFOP-3.6.2-16-2017-00002.

[7] Project no. 138903 has been implemented with the support provided by the Ministry of Innovation and Technology from the NRDI Fund, financed under the FK_21 funding scheme.

[8] Project no. 2019-1.3.1-KK-2019-00004 has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the 2019-1.3.1-KK funding scheme.

[9] Project no. 2018-1.2.1-NKP-2018-00004 has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the 2018-1.2.1-NKP funding scheme.

[10] This work has been funded by the European Commission via the H2020-ECSEL-2017 project SECREDAS (Grant Agreement no. 783119).

[11] 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.

List of own publications

Conference and Workshop Papers

[C1] András Gazdag, Levente Buttyán, and Zsolt Szalay
Towards Efficient Compression of CAN Traffic Logs
34th Int. Coll. on Adv. Manufacturing and Repairing Vehicle Industry, 2017.

[C2] András Gazdag, Levente Buttyán, and Zsolt Szalay
Efficient lossless compression of CAN traffic logs
25th Int. Conference on Software, Teleco. and Comp. Networks (SoftCOM), 2017.

[C3] András Gazdag, Tamás Holczer, Levente Buttyán, and Zsolt Szalay
Vehicular can traffic based microtracking for accident reconstruction
Vehicle and Automotive Engineering, Springer, 2018.

[C4] András Gazdag, Dóra Neubrandt, Levente Buttyán, and Zsolt Szalay
Detection of Injection Attacks in Compressed CAN Traffic Logs
Security and Safety Interplay of Intelligent Software Systems, Springer, 2019.

[C5] András Gazdag, Csongor Ferenczi, and Levente Buttyán
Development of a Man-in-the-Middle Attack Device for the CAN Bus
1st Conference on Information Technology and Data Science, 2020.

[C6] András Gazdag, György Lupták, and Levente Buttyán
Correlation-based Anomaly Detection for the CAN Bus
Euro-CYBERSEC, 2021.

[C7] Irina Chiscop, András Gazdag, Joos Bosman, and Gergely Biczók
Detecting Message Modification Attacks on the CAN Bus with Temporal Convolutional Networks
Proceedings of the 7th International Conference on Vehicle Technology and Intelligent Transport Systems, 2021.

Journal Papers

[J1] András Gazdag, Levente Buttyán, and Zsolt Szalay
Forensics aware lossless compression of CAN traffic logs
Scientific Letters of the University of Zilina, 2017

[J2] András Gazdag, Szilvia Lestyán, Mina Remeli, Gergely Ács, Tamás Holczer, and Gergely Biczók
Privacy pitfalls of releasing in-vehicle network data
Vehicular Communications, 2023.

[J3] András Gazdag, Rudolf Ferenc, Levente Buttyán
CrySyS dataset of CAN traffic logs containing fabrication and masquerade attacks
Nature: Scientific Data, 2023.

References

[1] Cohen, A., and Nissim, K. Towards formalizing the gdpr's notion of singling out. *Proceedings of the National Academy of Sciences 117*, 15 (2020), 8344–8352.

[2] Cohen-Hadria, A., Cartwright, M., McFee, B., and Bello, J. P. Voice anonymization in urban sound recordings. In *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)* (2019), IEEE, pp. 1–6.

[3] Ellis, G. Filters in control systems. *Control System Design Guide 9* (2012), 165.

[4] Hanselmann, M., Strauss, T., Dormann, K., and Ulmer, H. CANet: An unsupervised intrusion detection system for high dimensional can bus data. *IEEE Access 8* (2020), 58194–58205.

[5] Koscher, K., Czeskis, A., Roesner, F., Patel, S., Kohno, T., Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., and Savage, S. Experimental security analysis of a modern automobile. In *2010 IEEE Symposium on Security and Privacy* (2010), pp. 447–462.

[6] Kukkala, V. K., Thiruloga, S. V., and Pasricha, S. INDRA: Intrusion detection using recurrent autoencoders in automotive embedded systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 39* (2020), 3698–3710.

[7] Lestyan, S., Ács, G., Biczók, G., and Szalay, Z. Extracting vehicle sensor signals from CAN logs for driver re-identification. In *Proceedings of the 5th International Conference on Information Systems Security and Privacy, ICISSP 2019, Prague, Czech Republic, February 23-25, 2019* (2019), P. Mori, S. Furnell, and O. Camp, Eds., SciTePress, pp. 136–145.

[8] Miller, C., and Valasek, C. Remote exploitation of an unaltered passenger vehicle. Tech. rep., IOActive Labs Research, 2015.

[9] Taylor, A., Japkowicz, N., and Leblanc, S. Frequency-based anomaly detection for the automotive can bus. In *2015 World Congress on Industrial Control Systems Security (WCICSS)* (2015), pp. 45–49.

[10] Weber, M., Wolf, G., Sax, E., and Zimmer, B. Online detection of anomalies in vehicle signals using replicator neural networks. In *ESCAR 2018* (2018).