# On Frame Fingerprinting and Controller Area Networks Security in Connected Vehicles

Alessio Buscemi*, Ion Turcanu*†, German Castignani* and Thomas Engel*
*Faculty of Science, Technology and Medicine (FSTM), University of Luxembourg
†Luxembourg Institute of Science and Technology (LIST), Luxembourg
{alessio.buscemi,thomas.engel}@uni.lu
ion.turcanu@list.lu german.castignani@ext.uni.lu

***Abstract*— Modern connected vehicles are equipped with a large number of sensors, which enable a wide range of services that can improve overall traffic safety and efficiency. However, remote access to connected vehicles also introduces new security issues affecting both inter and intra-vehicle communications. In fact, existing intra-vehicle communication systems, such as Controller Area Network (CAN), lack security features, such as encryption and secure authentication for Electronic Control Units (ECUs). Instead, Original Equipment Manufacturers (OEMs) seek security through obscurity by keeping secret the proprietary format with which they encode the information. Recently, it has been shown that the reuse of CAN frame IDs can be exploited to perform CAN bus reverse engineering without physical access to the vehicle, thus raising further security concerns in a connected environment. This work investigates whether anonymizing the frames of each newly released vehicle is sufficient to prevent CAN bus reverse engineering based on frame ID matching. The results show that, by adopting Machine Learning techniques, anonymized CAN frames can still be fingerprinted and identified in an unknown vehicle with an accuracy of up to 80 %.**

***Index Terms*—Connected Vehicles Security, CAN Bus Reverse Engineering, Machine Learning, Frame Identification**

## I. Introduction

In recent years, the digitalization of the automotive sector and release of new technologies in the market have lead to a dramatic growth in the number of Electronic Control Units (ECUs) present inside vehicles. The data sent from these components is a valuable source of information for researchers and companies proposing solutions for connected vehicles, such as fleet management and cloud services [1], [2]).

As of today, most of the data sent from these ECUs transit on the Controller Area Network (CAN) bus, a message-based protocol considered the de-facto world standard for in-vehicle communications. CAN allows ECUs to send messages to each other without a master node orchestrating the communication. However, attention was brought to the lack of security of the CAN bus, due to the fact that no encryption is put in place [3]–[5]. In this regard, a variety of attacks have been presented in literature [6], [7]. The magnitude of such threats will likely be amplified by the increase of vehicle connectivity and the subsequent augment of access points for attacks.

Despite the absence of security features, the data transiting on the CAN bus is not easily interpretable. As a matter of fact, each Original Equipment Manufacturer (OEM) encodes the data with its own proprietary format, which is kept secret from the general public. The only way to disclose information regarding these formats is through reverse engineering, which is traditionally performed by following a list of tedious manual operations [8]. Recent works have focused on automating this process to reduce the time and manual effort required [9]–[14]. The proposed solutions are mostly based on correlating the CAN data with external sensors and involve precise actions to be performed by a trained operator during data collection.

In one of our recent studies we unveiled that the IDs of CAN frames sent by ECUs can be exploited to carry out reverse engineering in a highly automated way [15]. The main reason is that a limited number of Tier-1 suppliers provide the majority of ECUs to car manufacturers worldwide. As a consequence, the same ECU can be found in the electronic system of multiple vehicle models, which send the same frames using the same frame IDs. This characteristic can be exploited by matching the frame of a vehicle to reverse engineer with frames of known vehicles sharing the same frame ID. Differently from related work, this approach is event-agnostic, i.e. the data can be decoded without knowing the events occurred at data collection time. By using this method, an attacker with remote access to the CAN bus of a connected vehicle can perform reverse engineer and inject an attack without physical access nor prior knowledge about the target vehicle.

OEMs seem unwilling to bring core modifications to the CAN protocol (i.e., by adding encryption) as it would inevitably bring massive disruptions in the supply chains. One potential solution to protect against frame matching based reverse engineering approaches and improve the security of the CAN bus, while meeting the necessities of the OEMs, is to avoid the reuse of frame IDs for newly released vehicle models.

In this work, we investigate whether the abandonment of the frame ID reuse practice is sufficient to completely anonymize the frames, thus nullifying the benefit of a reverse engineering approach based on frame matching. In particular, we study the possibility of performing matching on anonymized frames by exploiting other properties of the frames. If confirmed, the implication would be that frame-matching based algorithms can still be used to perform CAN bus reverse engineering despite the ID anonymization with minimal extra effort. This would further corroborate the view that OEMs must improve the CAN bus security by applying substantial modifications to the protocol. To validate this thesis, we propose a frame



IEEE Consumer Communications & Networking Conference (CCNC 2022)

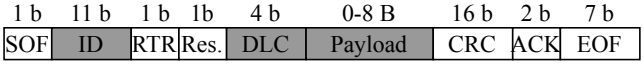


Figure 1. Structure of a CAN frame

deanonymization algorithm, which makes use of Machine Learning (ML) models to fingerprint frames based on their characteristics and dynamic behaviour. The ML models can then recognize the frames in a new vehicle to reverse engineer based on previously acquired knowledge.

The contribution of this work can be summarized as follows:

- We investigate the benefits of allocating a complete new set of IDs to each newly released vehicle model.
- We present and evaluate a new ML-based algorithm, whose goal is to deanynonymize frames. This algorithm works under Open Set Recognition (OSR) assumptions.
- We discuss the implications of such a method in the scope of CAN bus security.

## II. Background and Related Work

The communication on the CAN bus relies on messages, or *frames*. The frames do not contain any information regarding the sender nor the receiver ECU. A CAN frame is composed of a number of fields, as shown in Figure 1. This work focuses on three of them: (i) Identifier (ID) – uniquely represents the frame and assigns its priority, (ii) Data Length Code (DLC) – reports the payload length expressed as an integer between 0–8 Byte, and (iii) Payload – contains the actual information carried by the frame.

ECUs send one or more CAN frames periodically. In absence of collisions, these frames are received by all the ECUs connected to the bus. While an ECU can receive or send frames with different IDs, all frames associated with the same ID are sent by the same ECU. Due to the lack of a master node supervising the communication on the bus, multiple messages can be sent simultaneously by different ECUs, thus generating a collision. When a collision occurs, higher priority frames (defined by their IDs) override lower priority frames, which then have to be re-transmitted.

CAN bus reverse engineering is the process of finding the boundaries of the signals within a frame payload, known as *tokenization*, and decode their format (e.g. scale factor and offset) and their semantic meaning (i.e. what vehicle function they encapsulate), known as *translation*. In the manual approach, the reverse engineering is performed by a human operator who physically connects and disconnects ECUs and detects changes in the CAN traffic [8]. Some signals can also be retrieved by injecting diagnostic messages through the OBD-II port to generate a response from the CAN bus.

In automated reverse engineering, tokenization is mostly achieved by analyzing the flipping of the bits composing the frames over time [11], [16]. After tokenization, typically, the tokens are compared with GPS/IMU data, which offer a ground truth about the current status of the vehicle at driving time, to find correlations [12]. The injection of specific diagnostic messages and the analysis of subsequent responses from the CAN bus is often automated too. The translation, or part of it, can be achieved through ML-based classification [10], [13], [14]. In this approach, an ML classifier is trained to recognize the characteristics (or features) of each signal among CAN traces from known vehicles, so that it can later exploit this knowledge on an unknown vehicle.

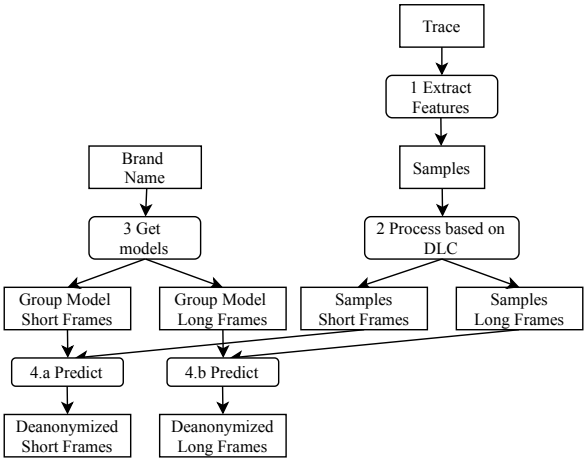


Figure 2. Pipeline of the proposed solution.

By analyzing a large dataset containing CAN traces obtained from 427 distinct vehicle models, we noticed that OEMs do not change the frame IDs when mounting the same ECU on multiple vehicle models. Assuming a decoded set of frames is available (i.e., via manual reverse engineering), this can be exploited to decode unknown CAN traces by simply matching the unknown frame IDs with the ones from the decoded dataset [15]. One way to prevent this is to anonymize the frame IDs for each newly released vehicle model. In this work, we show that it is still possible to fingerprint and identify anonymized frames with high accuracy, hence facilitating the automated reverse engineering process based on frame matching.

## III. Fingerprinting Anonymized CAN Frames

In this work, we assume that frame IDs are anonymized, i.e., same ECU model manufactured by the same OEM and installed in any two different vehicle models use different frame IDs for frames carrying the same vehicle functions. In particular, we make two fundamental assumptions:

1) Following the standard CAN protocol, the new ID associated to a frame should uniquely identify that frame.
2) The attribution of new IDs operated by the OEM should not be random, but rather preserve the frames priority. This assumption is important because lower-priority frames have more chances to be overwritten by higher-priority frames, when sent on the CAN bus simultaneously.

Hereafter, we present a tool to deanonymize frames which are anonymized following the aforementioned assumptions. The pipeline of this method is shown in Figure 2. The tool firstly divides the CAN trace collected from the vehicle to reverse engineer into $n$ sub-traces, where $n$ corresponds to the amount of unique IDs. Every sub-trace contains the payload and

DLC of the frames with the same ID, following the temporal order in which they appear in the trace. Then, from each sub-trace we extract six features described as follows:

- **Payload Length** – it corresponds to the number of data payload bytes, as reported in the DLC field.
- **Mean Sending Frequency** – it is the mean frames sending frequency, as recorded by the CAN dongle.
- **Standard Deviation of Sending Frequency** – it is the standard deviation of the frames sending frequency, as recorded by the CAN dongle. The sending frequency is influenced by the precision of the internal clock of the sending ECU. As demonstrated by Cho and Shin [17], this property can be used to fingerprint the sending ECUs.
- **Percentage of Active Bits** – it corresponds to the percentage of bits that flip at least once among the total number of bits in the frame payload throughout the trace. A bit is said to flip, when its value changes from 0 to 1 or vice versa.
- **Mean Bit Flip Rate** – it is the mean flip rate of the bits in the frame. Assuming a sub-trace composed by $f$ consecutive frames, the Bit Flip Count (BFC) of a bit $b$ in the payload corresponds to the total number of times that the bit $b$ flips. The bit flip rate is then calculated as $BFC/(f-1)$ [11].
- **Frame Priority** – the frames within a trace are divided into quartiles based on their priority, which is given by their frame ID.

The trace splitting and features extraction process are referred in Figure 2 (step 1).

Once a sample is generated for every sub-trace in the described manner, the algorithm splits the set of samples into two distinct subsets according to their length (extracted from the DLC) (see step 2, Figure 2). The first subset contains all frames whose payload is 8 Byte while the other includes all frames whose payload is shorter than 8 Byte. We refer to frames in the former set as *long frames* and to the frames in the latter as *short frames*.

From a preliminary analysis of a dataset of 427 traces, each collected from a different vehicle model, we discovered that as much as 70.5 % of frames are long, while 29.5 % are short. The reason behind this division is that the feature Payload Length, while constituting a helpful discriminant for fingerprinting the short frames, is ineffective for the long frames. Therefore, this feature is removed from the set of long frames.

Each of the two sets of samples is given in input to an ML model, specifically trained on samples (see step 3, Figure 2) of the same kind (i.e., from long or short frames). This model is trained exclusively on samples from the same industrial group/alliance of the current vehicle to reverse engineer. The underlying rationale is that vehicles are more likely to have more frames in common with models from the same industrial group. Preliminary tests highlight that training specifically a classifier for each industrial group rather than on all samples leads to lower amount of misclassified samples, due to the reduced variance. Finally, all the samples are deanonymized (see step 4, Figure 2).

Table I
EVALUATION SET STATISTICS

| OEM group | n. vehicle traces | n. unique frames | mean n. frames per vehicle | long / short frames (%) |
|---|---|---|---|---|
| A | 4 | 65 | 32.75 | 61.1 / 39.9 |
| B | 5 | 87 | 45.4 | 58.6 / 41.4 |
| C | 6 | 119 | 57.5 | 47.0 / 53.0 |
| D | 6 | 193 | 52.5 | 88.9 / 11.1 |
| E | 10 | 164 | 53.6 | 95.0 / 5.0 |
| F | 14 | 289 | 47.1 | 99.7 / 0.3 |
| G | 24 | 190 | 33.5 | 82.3 / 17.7 |
| H | 25 | 335 | 73.28 | 86.1 / 13.9 |
| I | 25 | 323 | 71.2 | 77.1 / 22.9 |
| J | 25 | 293 | 68.7 | 49.9 / 50.1 |
| K | 32 | 299 | 43.9 | 96.2 / 3.8 |
| L | 33 | 491 | 34.5 | 71.4 / 28.6 |
| M | 60 | 548 | 40.4 | 45.8 / 54.2 |
| N | 76 | 502 | 40.2 | 56.0 / 44 |
| O | 82 | 630 | 55.2 | 83.3 / 16.7 |

## IV. PERFORMANCE EVALUATION

Our algorithm is validated on a set of 10 s CAN traces obtained from 427 distinct vehicle models from 28 different automotive brands belonging to 15 different industrial groups from EU, USA, Japan, India and South Korea. Each of these traces was collected by a partner company with a `PCAN-USB FD` from a parked vehicle, with no action performed by a human operator. The vehicles in our evaluation set contain a total of 33 034 CAN frame IDs, from which we extract an equal number of samples. Table I summarizes other relevant information for all (anonymized) industrial groups in the evaluation set. For each industrial group, the table reports:

- The number of vehicle traces (n. vehicle traces).
- The number of unique frames that are found across all vehicle models (n. unique frames), as identified by their ID. The table highlights that a bigger and more variegated set of vehicle models corresponds to an increase in the total number of unique frames.
- The mean number of unique frames per vehicle (mean n. frames per vehicle). It depends on the level of digitalization of the vehicles produced by the manufacturer, which is usually correlated to the market segment (i.e. high-end vehicle models have more electronic components).
- The percentage of long and short frames on the total number of unique frames (long / short frames).

### A. Performance Metrics

In our evaluation we adopt a *leave-one-out-cross-validation* approach. Namely, each vehicle in the dataset is iteratively considered as the vehicle to reverse engineer and its ground truth is discarded. The classifiers are then trained on the rest of the dataset.

Our classification task is an Open Set Recognition (OSR) problem. In an OSR problem the knowledge of the world is incomplete at training time [18]. As opposed to the standard closed-world classification scenario, the ML models do not

have only to classify samples from known classes, but also to adequately reject samples belonging to unknown classes. As a matter of fact, apart from the ECUs mounted in previous vehicle models, we expect newly released vehicle models to be also equipped with last generation components, which will send frames unseen until then. For this reason, the trace of a new vehicle to reverse engineer may contain frames that the ML model has never been trained on.

In the OSR scenario the classes are typically divided into four sets: Known Known Class (KKC), Known Unknown Class (KUC), Unknown Known Class (UKC), Unknown Unknown Class (UUC). KKC is the set of classes for whom a distinct label is available. In our case, it is the set of labels (the frame IDs) associated with the frames on which the model was trained on. On the contrary, UUC is the set of the classes on which the classifier has never been trained on and for whom no side semantic information is available at training time. KUC and UKC are out of the scope of this work.

Being an OSR, we cannot evaluate our tool with metrics commonly accepted for standard ML close-world problems, such as the accuracy and F1-Score. New accuracy metrics for OSR tasks are firstly presented by Júnior et al. [19] and respectively reported in Equation (1) and Equation (2), namely Accuracy for KKC (AKS) and Accuracy for UUC (AUS):

$$AKS = \frac{\sum_{i=1}^{C}(TP_i + TN_i)}{\sum_{i=1}^{C}(TP_i + TN_i + FP_i + FN_i)} \quad (1)$$

$$AUS = \frac{TU}{TU + FU} \quad (2)$$

In Equation (1), $TP_i$, $TN_i$, $FP_i$, $FN_i$ correspond respectively to the number of true positives, true negatives, false positives, and false negatives for the i-th KKC $i \in \{1, ..., C\}$, where $C$ corresponds to the cardinality of KKC. Note that $FN_i$ includes the samples of KKC wrongly classified as unknowns. In Equation (2), true unknowns $TU$ correspond to the samples of UUC correctly classified as unknown, while false unknowns $FU$ correspond to the samples of UUC wrongly classified with one of the KKC labels.

### B. Classifiers

OSR problems are mainly addressed in two ways: (i) by enabling a common ML classifier to reject unknown samples, and (ii) by employing a classifier inherently designed to deal with UUC.

Regarding (i), we adapt a Random Forest (RF) [20] classifier and a Fully Connected Neural Network (FCNN)[21] to reject samples whose confidence score for the predicted label is below a certain *rejection threshold*, in the way described in [18]. We choose RF due to its robustness to outliers and the consistent handling of unbalanced datasets, which is particularly relevant in our case. After a tuning on the RF, we discovered that the optimal performance is obtained with around 200 tree estimators. Regarding FCNN, we choose it for its robustness to outliers and the overall superior performance of neural networks compared to traditional ML classifiers documented in literature [21]. After an extensive tuning on the number of layers and neurons for each layer, we found out that having more than three hidden layers with more than 1024 neurons each (and ReLu activation function) does not improve the overall performance of the model.

For what concerns (ii), we have reviewed the main inherently-Open Set (OS) classifiers in literature. The vast majority of related works specifically addresses computer vision tasks – as OSR is especially relevant for this field – and, therefore, the new classifiers are designed accordingly. In this regard, a particular effort is put into the adaptation or design of new Convolutional Neural Network (CNN) architectures [18], [22]. However, since our data is characterized by a maximum of six features, as opposed to the high-dimensional data typically processed in computer vision tasks, the majority of the methods presented in literature are unsuitable for our task.

For this reason, we selected two algorithms, PI-SVM [23] and Extreme Value Machine (EVM) [24], which can process input with different spatial properties without the need of data dimensionality transformations. PI-SVM integrates the notions of data distribution from Extreme Value Theory (EVT) [25] into the widely known Support Vector Machine (SVM) classifier. EVM is a brand new algorithm, which also exploits the EVT theory about data distribution to group samples in consistent areas of the space.

### C. Comparison between classifiers

We first analyze the results obtained by RF, FCNN, EVM, and PI-SVM to assess which classifier provides the best performance overall. After testing PI-SVM and EVM based on the source code released by the authors and the settings suggested in the respective papers [23], [24], we performed an extensive tuning of the hyperparameters to optimize their performance. The tuning revealed that EVM and PI-SVM score AKS and AUS lower than 20 %. Given their poor performance, we further show only the results obtained with RF and FCNN.

Figure 3 shows the average AKS and AUS, along with 95 % confidence intervals, obtained testing RF and FCNN on the evaluation dataset. The figure highlights that AKS and AUS can vary greatly according to the choice of the rejection threshold. As expected, an increase of the rejection threshold corresponds to a decrease of the AKS and an increase of the AUS. As a matter of fact, the higher the threshold, the more samples are rejected. This causes an increase of the samples in KKC wrongly rejected (hence, the worsening of the AKS), and a decrease of the samples in UUC wrongly labeled as known (hence, the improvement of the AUS).

RF performs equal or better than all the other classifiers on both samples from the short and long frames subsets on all considered metrics and datasets. RF and FCNN achieve a similar AUS for what concerns samples from short frames. It is interesting to notice that, for high values of the rejection threshold, FCNN achieves a higher AKS compared to RF when considering samples from both the short and the long frames.

It has been shown in [26] that the probabilistic output provided by most classifiers is skewed towards the extreme

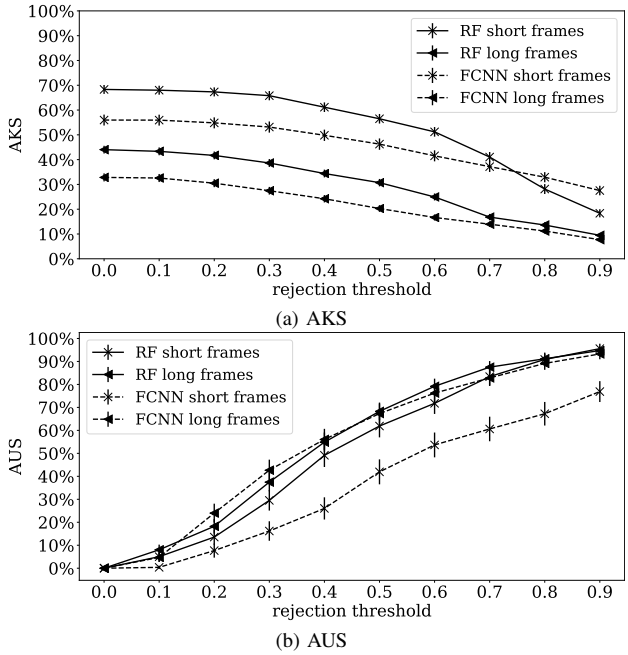


Figure 3. Comparison of the performance achieved by RF and FCNN for different rejection thresholds.

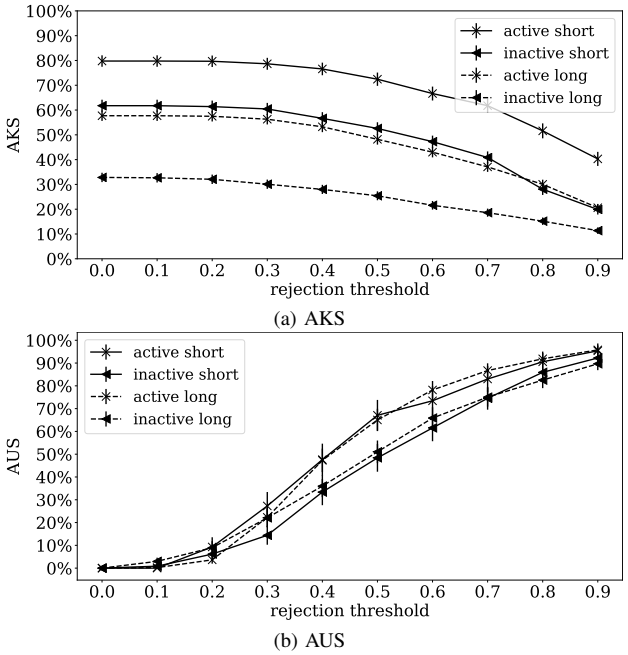


Figure 5. Comparison of the mean AKS and AUS obtained by RF on active and inactive samples.

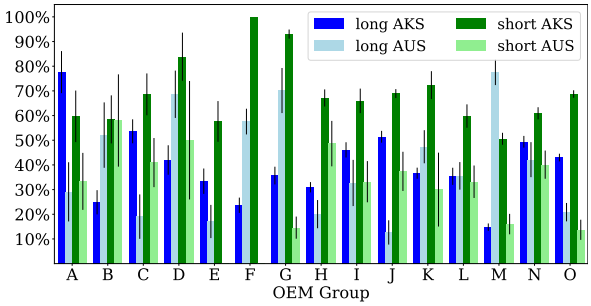


Figure 4. Performance of RF with a rejection threshold of 0.2 achieved on the vehicles of each OEM group.

values (0 and 1) and, thus, does not enable an understanding of the true probability of the prediction correctness. It follows that the predictions probabilities outputted by the selected classifiers do not correspond to the actual probability of a sample belonging to a determined class. Hence, since different classifiers calculate the probabilistic scores according to different logic, the setting of a certain rejection threshold can have consistently different impact on AKS and AUS.

### D. Result Analysis

Having designated RF as ultimate classifier for the task, based on the superior classification accuracy compared to the other algorithms, we further investigate its performance. In particular, we analyze the results obtained on the different industrial groups/alliances. Figure 4 illustrates the average AKS and AUS, with 95 % confidence intervals, obtained by RF with a rejection threshold of 0.2 on all vehicles for each industrial group/alliance. The results show a high variance in the average performance scored by the models trained on vehicles from different industrial groups for all considered metrics. We have not found any clear correlation between the classification accuracy and any other characteristic related to the composition of the sample sets for each industrial group reported in Table I.

Finally, we investigate the importance of the features Percentage of Active Bits and Mean Bit Rate. As mentioned in Section IV, the traces in the dataset were collected on parked vehicles. For this reason, numerous signals are never triggered and their bits never flip. Such signals are, for instance, all those related to the steering wheel or the vehicle speed. Following an analysis of the dataset, we discovered that, due to this phenomenon, the traces have 53.8 % of frames whose bits in the payload never flip. We refer to these frames as *inactive*. In contrast, the frames in which at least one bit flips during data collection are called *active*. The consequence of having inactive frames is that both the Percentage of Active Bits and Mean Bit Rate of the samples extracted from them are 0, thus making these two features irrelevant for their fingerprinting. By extension, we refer to the samples generated from the active and inactive frames as, respectively, *active* and *inactive samples*.

In this scope, we investigate the impact that inactive samples have on the overall classification performance. Figure 5 compares the mean AKS and AUS obtained by the classifier on active and inactive samples according to different rejection threshold. The results show that RF achieves higher AKS and AUS on active samples compared to inactive samples. For instance, with a no-rejection threshold (i.e. equal to 0), the classifier achieves an AKS of 79.8 % on active samples extracted from the short frames set, compared to 61.7 % scored on inactive samples from the same set, thus marking

a difference of 18.1 % in the performance. Still assuming no threshold, an even larger gap of almost 25 % remarks the difference in the performance achieved by the classifier on active samples compared to inactive samples extracted from the long frames set.

The presented results show that the features Percentage of Active Bits and Mean Bit Rate impact greatly on the classification performance. This fact suggests that the classification can be sensitively more accurate if the classifier is trained on samples from traces collected on vehicles that are driven.

## V. Conclusion

In this work, we study whether anonymizing the CAN frame IDs makes frame matching based CAN bus reverse engineering methods ineffective, thus preventing vehicle-agnostic remote attacks on CAN bus. We test the efficacy of this approach by fingerprinting frames whose IDs are anonymized. In this scope, we train ML classifiers to recognize frames based on six features extracted exploiting DLC, bit flipping, sending frequency and frame priority. We evaluate our fingerprinting method on 33 034 samples extracted from traces collected on 427 different vehicle models.

The results show that an RF classifier can recognize frames with a 8 Byte-long payload and frames with a shorter payload with a mean AKS up to 44.1 % and 68.7 % respectively. When considering samples extracted from frames with at least one flipping bit, a superior AKS up to 57.7 % and 79.8 % is achieved on samples extracted from long and short payload frames respectively. The presented results highlight that anonymizing the CAN frame IDs does not prevent reverse engineering based on frame matching. This suggests that connected vehicles are vulnerable to the remote decoding of CAN data by potential adversaries who can then inject attacks with minimal effort. In conclusion, to preserve better the anonymity of the CAN data transiting in-vehicle networks, substantial changes in the CAN encoding practices is necessary.

Future work includes research for new features able to increase the fingerprinting performance. We also plan to look for an effective defense against the fingerprinting of frames. Such a solution will have to respect the assumptions made in the paper, while keeping the CAN protocol unchanged to meet the needs of the OEMs.

## Acknowledgement

We acknowledge support from the National Research Fund (FNR) under grant number PRIDE15/10621687. We thank Xee for the provided datasets we used to validate our solution.

## References

[1] M. Bertoncello, G. Camplone, P. Gao, et al., "Monetizing car data—new service business opportunities to create new customer benefits," *McKinsey & Company*, 2016.

[2] L. Nkenyereye and J.-W. Jang, "Integration of big data for querying CAN bus data from connected car," in *9th International Conference on Ubiquitous and Future Networks (ICUFN)*, 2017, pp. 946–950.

[3] V. H. Le, J. den Hartog, and N. Zannone, "Security and privacy for innovative automotive applications: A survey," *Computer Communications*, vol. 132, pp. 17–41, 2018.

[4] J. Cui, L. S. Liew, G. Sabaliauskaite, and F. Zhou, "A review on safety failures, security attacks, and available countermeasures for autonomous vehicles," *Ad Hoc Networks*, vol. 90, p. 101 823, 2019.

[5] W. Wu, R. Li, G. Xie, et al., "A survey of intrusion detection for in-vehicle networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 919–933, 2019.

[6] G. Brindescu. "DARPA Hacked a Chevy Impala Through Its OnStar System." (2015), [Online]. Available: https://www.autoevolution.com/news/darpa-hacked-a-chevy-impala-through-its-onstar-system-video-92194.html (visited on 04/02/2021).

[7] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, vol. 2015, no. S 91, 2015.

[8] C. Quigley, D. Charles, and R. McLaughlin, "CAN Bus Message Electrical Signatures for Automotive Reverse Engineering, Bench Marking and Rogue ECU Detection," in *SAE Technical Paper*, SAE International, Apr. 2019.

[9] A. Buscemi, I. Turcanu, G. Castignani, R. Crunelle, and T. Engel, "Poster: A Methodology for Semi-Automated CAN Bus Reverse Engineering," in *13th IEEE Vehicular Networking Conference (VNC)*, IEEE, Nov. 2021.

[10] M. Jaynes, R. Dantu, R. Varriale, and N. Evans, "Automating ECU identification for vehicle security," in *15th International Conference on Machine Learning and Applications (ICMLA)*, IEEE, 2016, pp. 632–635.

[11] M. Marchetti and D. Stabili, "READ: Reverse engineering of automotive data frames," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 4, pp. 1083–1097, 2018.

[12] M. D. Pesé, T. Stacer, C. A. Campos, E. Newberry, D. Chen, and K. G. Shin, "LibreCAN: Automated CAN Message Translator," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, ACM, 2019, pp. 2283–2300.

[13] M. R. Moore, R. A. Bridges, F. L. Combs, and A. L. Anderson, "Data-Driven Extraction of Vehicle States From CAN Bus Traffic for Cyberprotection and Safety," *IEEE Consumer Electronics Magazine*, vol. 8, no. 6, pp. 104–110, 2019.

[14] A. Buscemi, G. Castignani, T. Engel, and I. Turcanu, "A Data-Driven Minimal Approach for CAN Bus Reverse Engineering," in *3rd IEEE Connected and Automated Vehicles Symposium (CAVS)*, Victoria, Canada: IEEE, Oct. 2020.

[15] A. Buscemi, I. Turcanu, G. Castignani, R. Crunelle, and T. Engel, "CANMatch: A Fully Automated Tool for CAN Bus Reverse Engineering based on Frame Matching," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 12, Nov. 2021.

[16] B. C. Nolan, S. Graham, B. Mullins, and C. S. Kabban, "Unsupervised time series extraction from controller area network payloads," in *IEEE 88th Vehicular Technology Conference (VTC-Fall)*, IEEE, 2018, pp. 1–5.

[17] K.-T. Cho and K. G. Shin, "Fingerprinting Electronic Control Units for Vehicle Intrusion Detection," in *25th USENIX Security Symposium (USENIX Security 16)*, Aug. 2016, pp. 911–927.

[18] C. Geng, S. Huang, and S. Chen, "Recent Advances in Open Set Recognition: A Survey," *CoRR*, vol. abs/1811.08581, 2018.

[19] P. R. M. Júnior, R. Souza, R. de Oliveira Werneck, et al., "Nearest neighbors distance ratio open-set classifier," *Machine Learning*, vol. 106, pp. 359–386, 2016.

[20] T. K. Ho, "Random decision forests," in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1, 1995, 278–282 vol.1.

[21] M. Z. Alom, T. M. Taha, C. Yakopcic, et al., "A State-of-the-Art Survey on Deep Learning Theory and Architectures," *Electronics*, vol. 8, no. 3, 2019.

[22] A. Bendale and T. E. Boult, "Towards Open Set Deep Networks," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1563–1572.

[23] L. P. Jain, W. J. Scheirer, and T. E. Boult, "Multi-class Open Set Recognition Using Probability of Inclusion," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Springer International Publishing, 2014, pp. 393–409.

[24] E. M. Rudd, L. P. Jain, W. J. Scheirer, and T. E. Boult, "The Extreme Value Machine," *CoRR*, vol. abs/1506.06112, 2015. [Online]. Available: http://arxiv.org/abs/1506.06112.

[25] S. Kotz and S. Nadarajah, *Extreme value distributions: theory and applications*, W. Scientific, Ed. 2000.

[26] A. Niculescu-Mizil and R. Caruana, "Predicting Good Probabilities with Supervised Learning," in *Proceedings of the 22nd International Conference on Machine Learning*, ser. ICML '05, Bonn, Germany: Association for Computing Machinery, 2005, pp. 625–632.