# Privacy-Preserving IDS for In-Vehicle Networks with Local Differential Privacy

Michael Kreutzer, Hervais Simo, Peter Franke

# Privacy-preserving IDS for In-Vehicle Networks with Local Differential Privacy

Peter Franke[1], Michael Kreutzer[2] and Hervais Simo[2]

[1] Technische Universität Darmstadt, Germany
[2] Fraunhofer Institute for Secure Information Technology, Darmstadt, Germany

**Keywords: In-Vehicle Networks, Privacy, Data Protection, Cybersecurity**

**Abstract.** Intrusion Detection Systems (IDS) for In-Vehicle Networks routinely collect and transfer data about attacks to remote servers. However, the analysis of such data enables the inference of sensitive details about the driver's identity and daily routine, violating privacy expectations. In this work, we explore the possibilities of applying Local Differential Privacy to In-Vehicle Network data and propose a new privacy-preserving IDS for In-Vehicle Networks. We have designed and conducted various experiments, with promising results, showing that useful information about detected attacks can be inferred from anonymized CAN Bus logs, while preserving privacy.

## 1 Introduction and Motivation

Modern vehicles contain a large and increasing amount of electronic components. These components are commonly integrated into so-called *In-Vehicle Networks* (IVN) and are referred to as ECUs (Electronic Control Units). ECUs usually consist of a controller and a transceiver for sending and receiving messages on the network and a small processing unit to fulfill specific functions. Protocols used in modern In-Vehicle Networks include the CAN [25], FlexRay (ISO 17458), LIN (ISO 17987), MOST (ISO 21806-1), and automotive ethernet. The most used and important protocol is CAN, which is why will focus on it further on. With the development of connected cars, an even wider range of features, e.g. entertainment and connectivity found their way into vehicles through the addition of more communication interfaces. The latter include conventional interfaces like Bluetooth and USB that have been enhanced by applications like Android Auto [13] or Apple CarPlay [1], but also WiFi as another mean of short-range communication. Additionally, many present-day vehicles offer long-range communication via cellular networks like GSM, UMTS or LTE by using embedded SIM cards. These new interfaces enable a new range of applications for entertainment, like streaming of music and videos or live map data. Connected Cars also offer great maintenance possibilities, e.g. by transferring diagnostic data to vehicle manufacturers or receiving over-the-air (OTA) updates for the vehicle. Convenience

functions like control of the vehicle via a mobile phone [30] are also enabled, just like safety features like automatic emergency calling [10]. Data gathered from connected cars (we use the terms car and vehicle interchangeably) also enable a growing number of business models, e.g. for usage-based individual insurance, taxation or car sharing [6]. However, the newly added interfaces come with new attack surfaces, which might be exploited from a distance. This is problematic, because IVN protocols, especially CAN, often provide little or no security features. Thus, an insecure remote access to the IVN might lead to an attacker taking control over the vehicle [20]. An increasingly popular approach to thwart such attacks is to rely on *Intrusion Detection Systems* (IDS) to detect ongoing attacks while also being adaptive to newly occurring attacks. Current uses of IDS in vehicles involve extensive collection of data that flows on the CAN Bus. Collected data about attacks is typically sent to remote servers for an in-depth analysis. This guarantees high detection rates, but also allows many sensitive conclusions about the car and the passengers [8,18,12]. In particular, it might allow to create profiles with details about the identification of a driver, driving behaviour, habits or where and when someone drove. Such in-depth data collection and analysis collides with the German Basic Law (Grundgesetz) Art. (1)(1), Art. (2)(1) and Art. 8 (1) of the Charter of Fundamental Rights of the European Union (CFR).

**Contributions.** This motivates our research into i) highlighting the tension between cybersecurity for IVN and privacy; and ii) developing privacy-preserving IDS for IVN, i.e., new IVN-tailored IT-solutions aiming to lessen the tension between cybersecurity and privacy. In this work, we explore the possibilities of applying Local Differential Privacy (LDP) [17,32,34] to In-Vehicle data collected by IDS and propose a privacy-preserving IDS. LDP offers strong, formal privacy guarantees when used. The key advantage of LDP from the users' perspective is that no trust in the data collector is needed. We especially highlight the challenge of applying LDP to raw CAN data, which has, to our knowledge, not been considered before. We propose an extension of a generic In-Vehicle IDS leveraging LDP techniques. We consider different forms of logs about detected intrusions ("anomaly logs"). For each form, we evaluate the perturbation induced by applying LDP methods to the logs. We evaluate if valuable information about the detected attacks can still be recovered from the logs, while protecting sensitive details about affected individuals, with promising results.

**Outline.** The remainder of this paper is organized as follows: In Sect. 2, we introduce the CAN protocol, related security flaws, attacks and countermeasures as well as Local Differential Privacy in a more detailed fashion. In Sect. 3, we present a generic model of an IDS for IVN and its privacy limitations. In Sect. 4, we propose an extended IDS model leveraging LDP and consider three concrete scenarios of anomaly log transmission with LDP. In Sect. 5, we describe our evaluation and its results.

## 2 Background

**The CAN protocol (ISO 11898).** The CAN (Controller Area Network) protocol was first specified by Robert Bosch GmbH in 1983. The latest version (CAN 2.0) was published in 1991 [25]. CAN allows multiple ECUs connected to a bus to communicate reliably with low latency while avoiding collisions. On a CAN network, messages are sent in the form of CAN frames. Any CAN frame sent on the bus is received by all connected ECUs. The transmission rate of a CAN Bus network is preconfigured and can vary depending on use cases. CAN networks can be in two states: The *recessive* state (logic 1) and the *dominant* state (logic 0). By default, the bus floats towards the recessive state. If two bits are transmitted at the same time, a dominant bit will always succeed. There are four different formats of CAN frames. The most important *data frame* (see Fig. 1), which is used for actual message transmission, contains the Arbitration ID (or CAN ID) field, which determines the content and priority of a message (not to be confused with the Vehicle Identification Number that is unique per vehicle and used when communicating with the manufacturer). Also, it contains a DATA field of up to 8 bytes and some further fields.

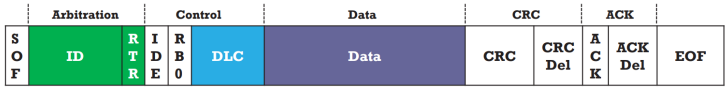


**Fig. 1.** Structure of a CAN 2.0 data frame. Image taken from Cho and Shin [5].

**Security Flaws in In-Vehicle Networks (IVN) and CAN.** IVN are used to fulfill critical vehicle functions, and failures and misbehaviour may lead to dramatic consequences for the passengers, possibly injuries or even loss of life. Security flaws may allow attacks that lead to such failures to take place. Thus, the topic of security is of high importance for IVN. CAN and similar protocols for IVN were designed in a time when the internet was still in an early phase. It was never considered or intended that such an IVN would be connected to the outside world, opening attack surfaces. Thus, there were no security considerations in the design of early IVN and especially CAN. Instead, they were primarily designed to offer safety and reliability, which obviously are sought after properties in vehicle development [35]. A concrete flaw in the CAN protocol is the lack of any authentication of the sender of messages or encrypted communication. Because messages are received by all participants on the network, any malicious party can read and use all information that is being sent. Furthermore, any participant is able to send messages with any CAN ID. This gives attackers the possibility to greatly disturb the vehicle's operation or even control its behaviour once access to the CAN Bus has been gained [35,15]. This can be established by

physical access to the vehicle, e.g. by adding an additional, malicious ECU. In the case of connected vehicles however, wireless communication interfaces and entertainment systems may contain additional vulnerabilities. When exploited, these vulnerabilities may give an attacker access to critical parts of the IVN from a distance [20].

**Attacks against In-Vehicle Networks.** The mentioned problems have already led to concrete attacks being performed on vehicles: Checkoway et al. [4] showed several attacks on vehicles. Using a CD with a specially prepared WMA file, they managed to achieve full control of the CAN Bus. In 2015, Miller and Valasek [20] managed to gain access to the CAN Bus of a Jeep Cherokee car by exploiting weaknesses in the infotainment system, and remotely controlled it. Particularly, they manipulated the functions of steering and braking, being able to stop the car or force it off the road. Researchers of the Tencent Keen Security Lab managed to remotely exploit weaknesses in Tesla [21] and BMW [3] vehicles. In both cases, they gained access to the CAN Bus and were thus able to control the vehicle.

**Existing Countermeasures.** Due to the usually long life span of vehicles, during which many new vulnerabilities may be found, offering an appropriate protection is a challenging task [27,2]. To fix vulnerabilities, a long maintenance of vehicle types and the used software is needed. In order to secure In-Vehicle Networks and the CAN Bus in particular, several measures have been tried. To increase the difficulty of attacks, a thoughtful design of the network topology is important. The strict separation of critical and non-critical components into several network segments can stop attacks from compromising the entire IVN. For CAN in particular, there have been attempts to introduce MAC (Message Authentication Code) schemes to ensure integrity [23,24]. However, these attempts suffer from the low amount of space (8 bytes) available in the data frame in which the MAC could be transmitted. The usage of the MAC might also result in a higher latency due to increased computation effort. The CAN FD protocol that was presented in 2012 [26] and is now slowly being introduced into vehicles might provide better possibilities for MAC, as it offers 64 bytes of payload in each frame. To increase security of the CAN Bus and the entire IVN, *Intrusion Detection Systems* (IDS) [5,31,22] are gaining increasing interest among researchers and practitioners. These systems attempt to detect ongoing attacks, enabling the development of countermeasures or immediate reactions. This detection can either be provided by simple signatures of known attacks, or by complex algorithms based on machine learning to detect anomalies. In addition, with the usage of Intrusion Prevention Systems (IPS), which allow active reactions on attacks detected by an IDS, attacks may also be prevented directly.

**Local Differential Privacy.** *Local Differential Privacy* (LDP) is a so-called *noise-based* privacy model. Here, privacy protection is achieved by the addition of

random noise to user data. The noise is added by each user individually before sending data to the collector. The collector can then aggregate multiple user responses to receive analysis results. LDP uses a *privacy parameter* or *privacy budget* $\epsilon$ that determines the level of privacy protection, where a lower $\epsilon$ means higher privacy protection, but also more required noise. LDP can be seen as an adaptation of the central model of Differential Privacy, where the noise is added by the collector to the analysis results of the collected data [7]. The advantage of LDP is that users no longer need to trust the data collector: Because the collector only receives noisy data, true data of individuals can not be abused or lost. Formally, LDP is defined as follows:

**Definition 1 ($\epsilon$-Local Differential Privacy [11]).** *A randomized algorithm $\mathcal{A}$ satisfies $\epsilon$-differential Privacy if for all pairs of client's values $v_1$ and $v_2$ and for all $R \subseteq Range(\mathcal{A})$, where $Range(\mathcal{A})$ denotes the image of $\mathcal{A}$,*

$$P[\mathcal{A}(v_1) \in R] \leq e^{\epsilon} \cdot P[\mathcal{A}(v_2) \in R].$$

Intuitively, this means that nobody can conclude with high confidence that a user has a specific value and not any other value - from the data this user reported. A convenient property of LDP is *composability*. It means that multiple releases of the same or correlated data result in limited and quantifiable loss of privacy protection. The *Sequential Composition* theorem states that the combination of a release with $\epsilon_1$-LDP and one with $\epsilon_2$-LDP still satisfies $(\epsilon_1 + \epsilon_2)$-LDP.

## 3 Privacy Limitations in Generic In-Vehicle IDS

Based mainly on the IDS ecosystems of ESCRYPT [9] and Argus Cybersecurity [36], we derived a model of the architecture of a generic In-Vehicle IDS, see Fig. 2. It entails the following components: 1) the Vehicle with a CAN Bus and a security system; 2) a Signature- and Specification-based IDPS component in the security system; 3) an Anomaly-based IDS component in the security system; and 4) a Backend Server performing the analysis. Processes and features provided by such a generic In-Vehicle IDS can be summarized in these steps:

1. Signature- and specification-based detection on the CAN data and blocking of known malicious traffic and attacks.
2. Anomaly detection for unknown attacks using the Anomaly-based IDS component on the data that have not been blocked in the previous step.
3. Transfer of anomaly data (logs and circumstances of the anomaly) about the anomalies detected by the Anomaly-based IDS to the Backend component.
4. Analysis of the collected anomaly data in the Backend.
5. Reaction: Deployment of Over-The-Air-Updates to the vehicle, containing updates of the detection rules for the ID(P)S.

In our generic in-vehicle IDS model, the detection of attacks takes place mainly inside the vehicle using the signature-based and anomaly-based ID(P)S. However, there is one step in which data transfer to a remote party occurs. This is

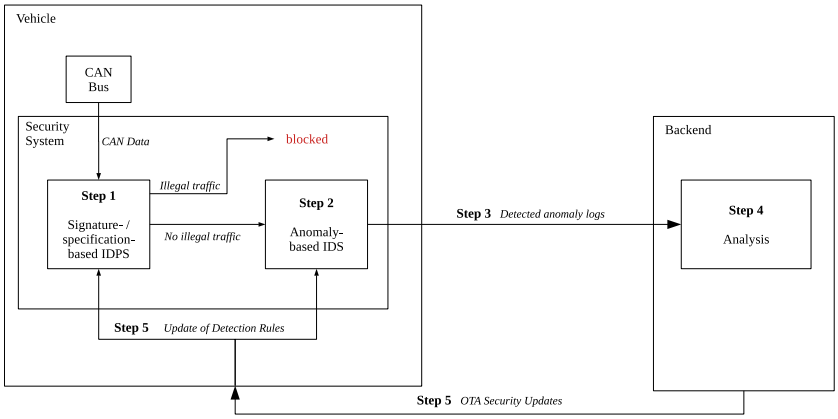


**Fig. 2.** Scheme of a generic IDS system.

Step 3, in which logs of detected anomalies are transmitted to a backend server belonging to the vehicle manufacturer or the provider of the IDS solution. The user's control over the transmitted anomaly logs is lost as soon as they leave the vehicle, resulting in potential privacy issues.

## 4 Our Approach

### 4.1 Generic IVN IDS with LDP

Our aim is to ensure privacy protection even if sensitive anomaly logs are processed outside of the vehicle (in Step 3 and 4 of our model). Because sensitive data is no longer protected as soon as it leaves the vehicle, the proposed privacy protection measures need to be applied before any data is sent to the backend. For this purpose, we propose the use of a LDP technique. This means that the anomaly logs that are sent in Step 3 need to be transformed to ensure privacy protection while they are still in the vehicle. Therefore, we modify our generic In-Vehicle IDS model, by adding a new component in the vehicle that perturbs anomaly logs in order to achieve LDP. This component is used in a new step between the anomaly detection and the transfer of the anomaly data to the backend. In this step, LDP techniques with the personal privacy budget $\epsilon_p$ are applied to the anomaly log. The selection of $\epsilon_p$ can be done by each vehicle, resp. the owner, individually, so that each vehicle $v$ has an own budget $\epsilon_{pv}$. In this way, each individual can choose their own minimum privacy level. To analyze the data, another new component is needed in the backend that aggregates the anomaly logs perturbed with LDP. So, we propose a new model of a generic in-vehicle IDS system that uses LDP for privacy protection, incorporating the previously mentioned new components. It is illustrated in Fig. 3 from the viewpoint of a single vehicle $v$. These are the steps taken in the new model:

1. Signature- and specification-based detection on the CAN data and blocking of known malicious traffic and attacks.
2. Anomaly detection for unknown attacks using the Anomaly-based IDS component on the data that have not been blocked in the previous step.
3. Application of LDP techniques to data (logs and circumstances) about the anomalies detected by the Anomaly-based IDS. This step takes place in the component for Application of LDP. The anomaly data are perturbed using the LDP techniques with the user-selected privacy parameter $\epsilon_{pv}$.
4. Transfer of the perturbed anomaly data to the Backend component.
5. Aggregation of collected anomaly logs from multiple vehicles that are perturbed with LDP in the Backend.
6. Analysis of the collected and aggregated anomaly data in the Backend.
7. Reaction: Deployment of Over-The-Air-Updates to the vehicle, containing updates of the detection rules for the ID(P)S.

Because LDP is applied before data is transferred from the vehicle, privacy is protected during transfer and analysis of the data (Step 4 and 5). In this way, the driver can be confident that his privacy is protected without relying on the manufacturer of the vehicle or the provider of the IDS solution.

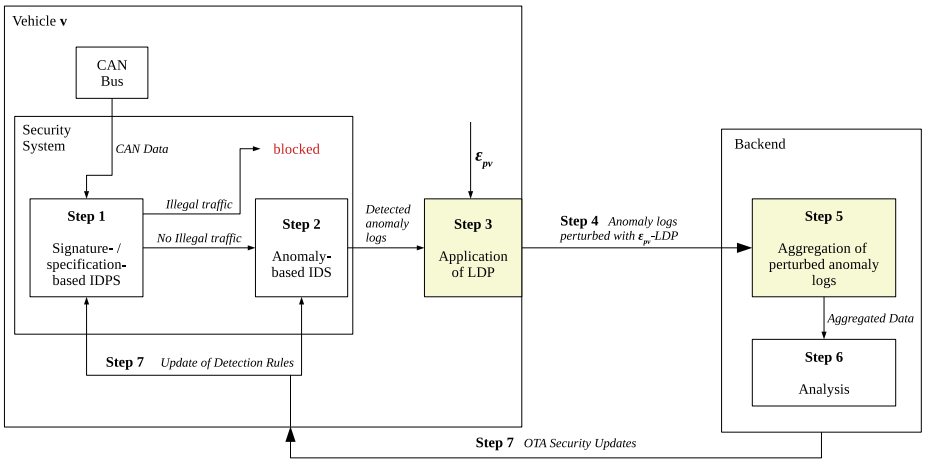


**Fig. 3.** Illustration of privacy protection using LDP in our scheme of a generic IDS system from the viewpoint of a vehicle.

### 4.2 Modelling Anomaly Logs

After introducing the model of the generic IVN IDS with LDP, we now elaborate on the form of anomaly logs, their content and how to apply LDP to them. We provide a formal model of an anomaly log and LDP for anomaly logs, as well as possible data types and their privacy sensitivity. These are used in Sect. 4.3 to give concrete scenarios of anomaly log transmission.

**LDP for Anomaly Logs.** We will now provide a formal definition of Local Differential Privacy for anomaly logs. We assume that an anomaly log consists of multiple, insensitive or sensitive parts, among which the privacy budget $\epsilon_p$ is distributed. We define an anomaly log and LDP for Vehicle Anomaly Data:

**Definition 2 (Anomaly Log).** *An anomaly log is a tuple $L = (v, w)$. The tuple $v = (v_1, v_2, ..., v_j)$ consists of $j$ attributes that are privacy sensitive. The tuple $w = (w_1, w_2, ..., w_k)$ consists of the $k$ attributes that are privacy insensitive.*

**Definition 3 (LDP for Vehicle Anomaly Data).** *Let $L = (v, w)$ be an anomaly log, $v = (v_1, v_2, ..., v_j)$ be the privacy sensitive attributes of the log and $w = (w_1, w_2, ..., w_k)$ be the privacy insensitive attributes. Let $V_i$ be the domain from which the attribute $v_i$ comes. Let $\epsilon_p \in \mathbb{R}$ be the personal privacy parameter that is selected by the user. Let $\epsilon = (\epsilon_1, \epsilon_2, ..., \epsilon_j) \in \mathbb{R}^j$ such that $\epsilon_i \geq 0$ for all $i$. Let $A = (A_1, A_2, ..., A_j)$ be a tuple of algorithms. LDP for Anomaly Data is satisfied if:*

1. $\sum_{i=1}^{j} \epsilon_i \leq \epsilon_p$
2. *For every $i \in [1, j]$, every $v_{i1}, v_{i2} \in V_i$ and every $R \subseteq Range(A_i)$:*
   $Pr[A_i(v_{i1}) \in R] \leq exp(\epsilon_i) \cdot Pr[A_i(v_{i2}) \in R]$

This means that we select an algorithm for each individual part. Each algorithm must satisfy $\epsilon_i$-LDP for its own part. Due to the sequential composition of LDP, the sum of all used partial budgets $\epsilon_i$ must not surpass the privacy budget $\epsilon_p$ to guarantee $\epsilon_p$-LDP in total. This definition is general enough to be applicable to various scenarios of anomaly log transmission regarding the actual content, data types and algorithms used. In order to apply this definition in practice, we further specify possible contents of an anomaly log and their sensitivity.

**Content and Data Types in an Anomaly Log.** First, we consider data types that may be useful for an anomaly log of an IDS on the CAN Bus. We call these data types "abstract" data types and their concrete, mathematical representation the "basic" data type. The relevant abstract data types are these:

- *CAN Frame*: Under the assumption that the anomaly-based IDS that detected the anomaly operates on CAN data, CAN frames are relevant to give information about the anomaly. The relevant parts of a CAN frame are the CAN ID and the DATA field. The CAN DATA field has the basic data type of "general binary data", as it is a bit vector with a length of up to 64, and we can not assume anything about the content or structure. The CAN ID field can also be seen as general binary data, but it is known that it contains a numeric identifier value. Thus, it should be viewed as discrete numeric or categorical data. For the purpose of logging, we assume that each CAN frame is paired with a timestamp, which can be viewed as continuous numeric data.
- *High level data*: This type of data respects the semantics of CAN Bus messages and can be directly derived from CAN frames. Usually, information proprietary to vehicle manufacturers is needed for this task. This abstract data type can consist of various different basic data types. These types are single bits, continuous numeric data, discrete numerical or categorical data.

- *Metadata about the anomaly*: It is reasonable to assume that metainformation about the anomalies benefits the analysis. For example, the IDS could differentiate different types of anomalies/attacks and send this information. This metainformation can consist of any basic type.
- *Metadata about driving situation*: There are several more types of metainformation that don't directly concern the anomaly, but the situation in which it occurred and are collected additionally. Examples are the time of detection, the location at which the anomaly was detected, information about the driver and more. This abstract type can also contain any of the basic types.

**Sensitive and Insensitive Data Types.** Based on the information that they contain, we classify the abstract data types into sensitive and insensitive data types. The LDP techniques are applied only to the sensitive data types for privacy protection. In order to not jeopardize the privacy protection efforts, only a very limited amount of information should be classified as insensitive and transmitted without the application of LDP. Thus, we assume that only timestamps (relative to the start of the anomaly log) and metainformation about the anomaly should be considered insensitive, since other data can be used to deduce locations, driving behaviour, usage of vehicle features and more.

### 4.3 Scenarios of Anomaly Log Transmission

Based on our knowledge of the possible data types, we now illustrate possible scenarios of anomaly log transmission such that our definition of LDP for anomaly logs is satisfied. We give three concrete instantiations of such a scenario.

*Scenario 1*: Transmission of a whole log of CAN frames: In this scenario, a log of all CAN Bus traffic surrounding the anomaly is constructed. This means that a certain number of messages $n$ around the detection of the anomaly are included. Additionally, metainformation about the anomaly type is transmitted. An anomaly log $(v, w)$ is created in the following way: $v$ contains all the CAN frames divided into ID and DATA fields. $w$ contains timestamps and the type of the anomaly determined by the IDS. Using one algorithm suitable for reporting CAN ID values with LDP and one for CAN DATA fields, the log is perturbed. The privacy budget is divided equally among the CAN frames, and a fixed ratio is used to divide the budget per frame between the ID and DATA field.

*Scenario 2*: Transmission of a single random CAN frame: Since the transmission of large amounts of data requires adding a high amount of noise, in this scenario, only a single CAN frame is randomly selected from the CAN traffic log and transmitted to the backend. This allows using less privacy budget or sending less, but more accurate information with the same budget. Additionally, metainformation about the anomaly type is transmitted. The anomaly log $(v, w)$ consists of the following components: $v$ contains the single CAN ID and DATA field, $w$ contains the anomaly type. The algorithms are selected as in Scenario 1. The privacy budget is divided between ID and DATA field with a fixed ratio.

*Scenario 3*: Transmission of only CAN frames that have been classified as anomalous: Here, it is assumed that the IDS is able to determine which frame

caused the detected anomaly. Then, only one CAN frame that was classified as anomalous is transmitted to the backend, again with metainformation about the anomaly and timestamps. Else, this scenario is equivalent to Scenario 2.

Further scenarios are possible, e.g. including high level data, which were not in the focus of this work and the evaluation.

## 5 Evaluation

We performed an evaluation of our approach in the described scenarios. In this section, we describe the used dataset, our methodology and the results.

### 5.1 Dataset

For the evaluation, we obtained the "*Car-Hacking Dataset*" [14] created by the Hacking and Countermeasures Research Lab (HCRL) at Korea University. The labelled dataset contains 5 parts of CAN traffic, each lasting about 30-40 minutes: "DoS", "Fuzzy", "Spoofing gear", "Spoofing RPM" and attack-free data. The message counts for each part can be seen in Fig. 4. Each attack was per-

| Attack Type | # of messages | # of normal messages | # of injected messages |
|---|---|---|---|
| DoS Attack | 3,665,771 | 3,078,250 | 587,521 |
| Fuzzy Attack | 3,838,860 | 3,347,013 | 491,847 |
| Spoofing the drive gear | 4,443,142 | 3,845,890 | 597,252 |
| Spoofing the RPM gauze | 4,621,702 | 3,966,805 | 654,897 |
| GIDS: Attack-free (normal) | 988,987 | 988,872 | - |

**Fig. 4.** Message counts for each part of the HCRL Car-Hacking Dataset [14].

formed for 3-5 seconds and every part contains 300 attacks. For the DoS attack, a message with the ID 0x000 was injected at high frequency. For the Fuzzy attack, a message with a random ID and payload was injected. In the two spoofing attacks, a message was injected with the CAN ID for gear (0x316) respectively RPM (0x43f) information. The attributes collected are the timestamp, CAN ID, DLC (Data Length Code), the DATA field, and the label (injected or normal message). The dataset was created for and first used by Seo et al. in [28]. It is also used in [29].It is available at https://sites.google.com/a/hksecurity.net/ocslab/Datasets/CAN-intrusion-dataset.

### 5.2 Methodology

Here, we will describe the methodology and setup of our evaluation. We evaluate *Scenario 1* (Transmission of a whole log of CAN frames), *Scenario 2* (Transmission of a random frame from the log) and *Scenario 3* (Transmission of a frame that is marked as anomalous). In each scenario, we evaluate various population sizes, where the population size is the number of users reporting at a time, LDP algorithms and values of the privacy parameter $\epsilon_p$. In every configuration, we run the evaluation 5 times. We first describe the values we use for the mentioned parameters, then the evaluation procedure and finally the analysis of the results.

#### Parameters

**LDP Algorithms.** For the transmission of CAN IDs, we assume the existence of two cases that influence the algorithm choices: In the first case, only CAN IDs occurring in normal traffic are allowed on the bus, and all other IDs are filtered out by the Signature-/specification-based IDPS component (see Fig. 3). In our dataset, there are 27 normal IDs if the ID of $0x000$ is considered a normal ID, whereas the maximum number of IDs is 2048 (11-bit CAN IDs are used in the dataset). Such a reduced domain size is of advantage for processing times and the utility of the produced data. However, the Fuzzy attack can not be evaluated in this case, as all possible IDs occur in this part. In the second case, the anomaly logs can contain any of the 2048 possible IDs. The LDP algorithms we use are:

***CAN ID field (only normal IDs allowed)***: Exponential Mechanism (EM) [19] (originally used for central DP), Optimized Unary Encoding (OUE)[34]. These algorithms can estimate the frequency of all possible values. Due to the small domain of 27 elements, Unary Encoding techniques (bit vector with 1 digit for every possible value is reported) can be efficiently used here. For the EM, which needs a utility function for user values, we encode each ID into an integer number corresponding to the order of IDs, i.e. the lowest ID is encoded to 0 and the highest to 26. The distance between numbers is the utility function.

***CAN ID field (all IDs allowed)***: Optimized Local Hashing (OLH)[34]. Because of the larger domain (2048 elements), we use OLH, which uses hash functions to reduce communication cost while requiring more computation during aggregation. We did not evaluate the EM in this case because of poor results even with the normal IDs. We did not include OUE here because of its large communication cost (a vector of 2048 bits for each report).

***CAN DATA field / Payload***: Our choices were guided by the large domain size of $2^{64}$ for 64-bit long DATA fields, making it impossible to estimate frequencies for every value. We use Parallel Randomized Response (PRR), which estimates the mean of each individual bit. Also, we use the Prefix Extending Method (PEM) [32]. It can identify the most common values and their frequencies (heavy hitters). It divides users into groups that report prefixes of increasing length using OLH. With PEM, we identify the top 4 heavy hitters. For a DATA field of 64 bits, we use 6 groups with prefix lengths of 12, 22, 32, 42, 53 and 64. These choices support fast computation times, as the needed time is exponential

in the prefix length extension per group. By using less groups, one may improve the accuracy while increasing the computation time.

**Population sizes.** Population size in this context refers to the amount of reports on an anomaly that can be aggregated at once. The maximum population size that can be used is limited by the number of attacks in the dataset. We assume that one anomaly log in Scenario 1 contains 10 CAN frames in our evaluation. We construct the logs by choosing one attack frame and the 9 frames following it, ensuring at least one attack frame (but usually more) in each log. Using this approach, the maximum amount of anomaly logs that can be constructed from the DoS dataset is 103180. Thus, we limit the maximum population size to 100000 and consider the following population sizes: $[1000, 10000, 25000, 50000, 100000]$.

**Privacy parameter settings.** While we created our model to be as general as possible by allowing each user to choose their own privacy parameter $\epsilon_p$, we will assume that all users choose the same privacy parameter in this evaluation. The privacy parameter indicates the available privacy budget *per anomaly log* in this context. We evaluate the integer values in the range [1, 10] as privacy parameters per anomaly log. Further, we use 70% of the privacy budget available for one frame for the DATA field and 30% for the ID field.

**Anomaly labels by the IDS.** For the sake of simplicity, we assume that the IDS is able to perfectly tell apart the different attack types occurring in our dataset. In our scenarios (Sect. 4.3), we assume that the anomaly type is sent unperturbed in the anomaly log by each user. Thus, the aggregator is also able to distinguish anomaly logs originating from different anomalies. This allows us to perform the evaluation on each part of the dataset individually.

**Used Hardware and Software.** The evaluation was conducted on a machine with an AMD Ryzen 7 3700X and 16GB of RAM and a machine with an AMD FX-6300 and 8GB of RAM. Both machines used Windows 10. Python 3 was used for the implementation. The used third-party packages were diffprivlib, numpy, xxhash and matplotlib (pyplot). For the Exponential Mechanism, we used the implementation in the "DiffPrivLib" package by IBM [16]. For OLH, we took inspiration from the implementation by Wang available at [33].

**Evaluation Procedure** Here, we describe the steps taken in our evaluation. These are Data Preprocessing, Perturbation, Aggregation and Analysis.

**Data Preprocessing.** For each of the attack datasets, we create the anomaly logs for each population size in Scenario 1 by finding a frame labelled as malicious and extracting this and the following 9 frames. From each of these anomaly logs, an anomaly log for Scenario 2 is created by choosing a random frame from the log and one for Scenario 3 by choosing the first frame from the log. Further, we use the attack-free part in the dataset to create a histogram of the normally occurring CAN IDs and DATA field values, which is used in the analysis step.

**Perturbation.** For each combination of population size, privacy parameter $\epsilon_p$, dataset part and scenario, the corresponding input dataset $D$ is loaded. Let $\epsilon_f$ be the privacy budget per CAN frame ($0.1 \cdot \epsilon_p$ in Scenario 1). We run each

algorithm $A_I \in \{EM, OUE, OLH\}$ on $D$ with the privacy budget $\epsilon_i = \epsilon_f \cdot 0.3$ to create perturbed CAN ID data. On the Fuzzy dataset, perturbed data are only created for the case of 2048 allowed IDs (using OLH). We run each algorithm $A_D \in \{PRR, PEM\}$ on $D$ with the privacy budget $\epsilon_d = \epsilon_f * 0.7$ to create perturbed CAN DATA field data. In Scenario 1, the algorithms are run on all 10 frames in each anomaly log individually.

**Aggregation and Analysis.** For each scenario, population size, privacy parameter and dataset part, we aggregate the perturbed data in the following way: For each algorithm $A_I \in \{EM, OUE, OLH\}$, we aggregate the perturbed ID data into a histogram of ID frequencies. For the PEM algorithm, we aggregate the perturbed DATA field data to identify the $k = 4$ top heavy hitters and their frequencies. For the PRR algorithm, we use the perturbed DATA field data to estimate the frequency (which is also the mean) of every bit individually. The aggregated data are analyzed dependent on the dataset part used and thus the anomaly label: For the DoS Attack, we use the ID data to infer the CAN ID used in the attack. For the Gear spoofing and RPM spoofing attacks, we try to both identify CAN ID and DATA values used in the attack. Because the Fuzzy attack in our dataset uses completely random IDs and payloads, it is impossible to identify a single or few IDs and payloads responsible for the attack. In order to find out malicious CAN IDs and DATA field values, we use the histogram of ID values that was created from the aggregated anomaly logs respectively the heavy hitter DATA fields estimated by PEM. Frequencies of IDs and DATA fields that are smaller than 5% are considered insignificant and set to 0. The frequencies of IDs and DATA fields are then compared to a histogram of normal CAN traffic. Any value that is more than 3 times as frequent as in the normal data is pointed out as possible cause of the anomaly. If the PRR algorithm was used, only one DATA field value is always identified using the frequencies estimated for each bit: If the estimated frequency of a bit is higher than 0.5, it is set to 1, else to 0. The bit vector constructed in this way is pointed out as a malicious payload.

**Evaluation of the Analysis results.** We evaluate the analysis results in two ways: First, we compare the empirical error (Mean Absolute Error (MAE) averaged over the 5 runs) induced by the perturbation when compared with the same aggregation on unperturbed data. For the CAN ID algorithms, we create a histogram of the used IDs in the unperturbed input dataset and compare it with the estimated histogram (before post-processing). We compute the MAE between the two histograms. For the PRR algorithm, we compute the mean of each bit in the unperturbed data and use the MAE between the estimated mean values and the true values. For the PEM algorithm, we compare the up to 4 estimated heavy hitters and their frequencies with the true heavy hitters computed from the unperturbed data. We again use MAE between the estimated and the true frequencies. If a true heavy hitter is not identified, or an identified heavy hitter is not a true heavy hitter, the frequency of this heavy hitter is added to the error sum before the mean is taken. If CAN ID or DATA field values that caused the anomaly are inferred, we count one inference as successful detection if the inferred value actually caused the anomaly. If a value that was not the

cause of the anomaly was inferred, we count this as a false positive. We compute the detection rate and precision over the 5 runs of each evaluation setup.

### 5.3 Results

We now describe the findings of our evaluation. We illustrate the findings using plots, where one plot shows the performance (w.r.t. quality of results, not computing performance) of one algorithm in one scenario for all $\epsilon$ values and population sizes. Each plot contains 3 subplots containing empirical error, detection rate or precision for all $\epsilon$ values with one graph for each population size. We found that the obtained results for the DoS, RPM and Gear datasets were almost identical across all scenarios, which is why we only show plots for the RPM dataset for brevity. Throughout the experiments, it is a common feature that the incurred error drops when the privacy parameter $\epsilon$ is increased, as well as when the population size is increased. In the same way, the detection rate and precision rise. This is the expected behaviour, as larger samples decrease variance and a larger $\epsilon$ allows for more accurate information in one report. For the population size of 1000, the performance was inferior to the other population sizes. The MAE was a lot higher than the error for other population sizes, which is also manifested in a slower rise of the detection rate and precision with $\epsilon$. This leads us to the conclusion that a population size of 1000 is not sufficient for our use case in all of the tested scenarios. With larger population sizes, especially 25000 and upwards, meaningful results can be obtained while using smaller $\epsilon$. While the population size of 100000 led to the best performance, the improvement over 50000 was very small in most cases. In a real world scenario, it is obviously best to use the largest available population size that can still be processed in an appropriate time.

An interesting observation is that the performance in Scenario 1, in which a log of 10 CAN frames is transmitted in each report, was inferior to the performance in the Scenarios 2 and 3, where one CAN frame is transmitted. Even though the largest amount of data is collected in Scenario 1, the MAE was typically about 3 to 5 times as high as the Error of the same algorithm run on the same dataset part, but under Scenario 2 or 3. The achieved detection rates and precision in Scenario 1 also rose slower than in Scenario 2 and 3. We show this using OUE as an example, illustrated using Fig. 5, which shows the performance of OUE on the RPM dataset in Scenario 1 and 2. When using OUE in Scenario 1, the error was consistently big: For any population size, the MAE was always larger than 0.3, which is a large amount when remembering that the true value is a histogram of ID Frequencies where the sum of all frequencies is 1, which means that most individual frequencies are a lot smaller than 0.3. While the detection rate was 1 for all population sizes and $\epsilon$ values, meaning that the malicious ID could always be identified, the precision increased noticeably smaller than in Scenario 2 and 3, where a precision of 1 was achieved for every population size greater than 1000 even with $\epsilon = 1$. Because $\epsilon$ designates the privacy budget available per log, the budget that is available per CAN frame in Scenario 1 is only $\frac{1}{10}$ of $\epsilon$. Thus, every single frame needs to be perturbed a lot

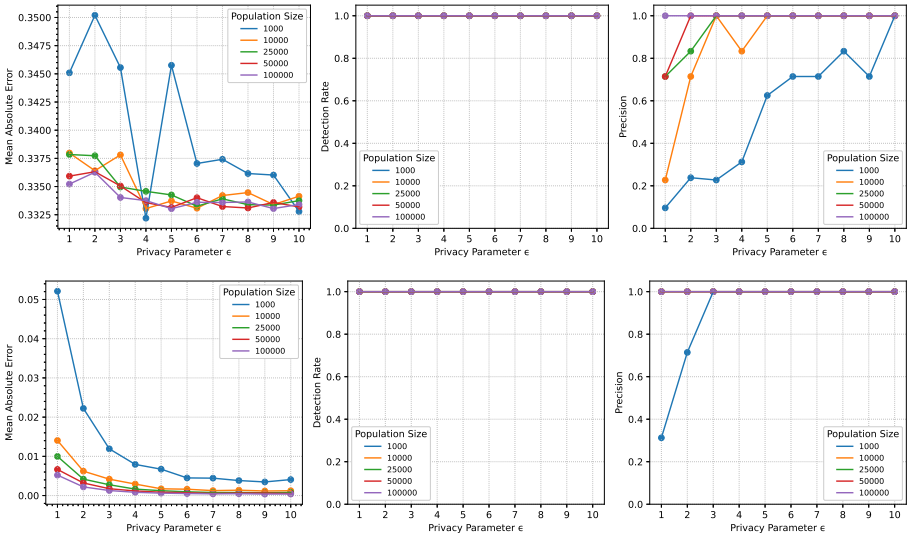


**Fig. 5.** Performance of the Optimized Unary Encoding on the RPM dataset in Sc. 1 (top) and Sc.2 (bottom)

more to achieve $\epsilon$-LDP for the whole log. In Scenario 2 and 3, only one frame is contained in the anomaly logs, and the whole $\epsilon$ can be used for this frame. Thus, each individual frame is perturbed much less than each frame in Scenario 1. The errors show that sending a small amount of data with low perturbation per data unit is better for the utility of the analysis results than sending a large amount of data among which the privacy budget needs to be divided, leading to high perturbation per data unit. Thus, the transmission of anomaly logs with LDP should take place in Scenario 2 or 3 and not Scenario 1.

When viewing the effectiveness of detecting malicious IDs and payloads with respect to the used $\epsilon$, very good detection rate and precision were usually obtained for most population sizes with $\epsilon$ of 3 and more per anomaly log, even in Scenario 1. This indicates that a higher privacy budget per log is not necessary in our case. Choosing a too high budget may even be counterproductive, as it may lead to a rapid decrease in the protection level if multiple reports are sent by one user. Then, that a stricter limit on the number of reports per user might need to be used. If the same information can be learned using less privacy budget, this decreases the total amount of useful information learnable by the aggregator.

An exception from the good results seen with most algorithms is the EM. It showed consistently bad performance in all scenarios, with unacceptably high error and low precision, as demonstrated in Fig. 6 for Scenario 3 with the RPM dataset. Even here, an improvement in detection rate and precision can be seen in the Scenarios 2 and 3 over Scenario 1. The bad results with normal IDs were the main reason why we did not evaluate the EM with the full ID range. Because

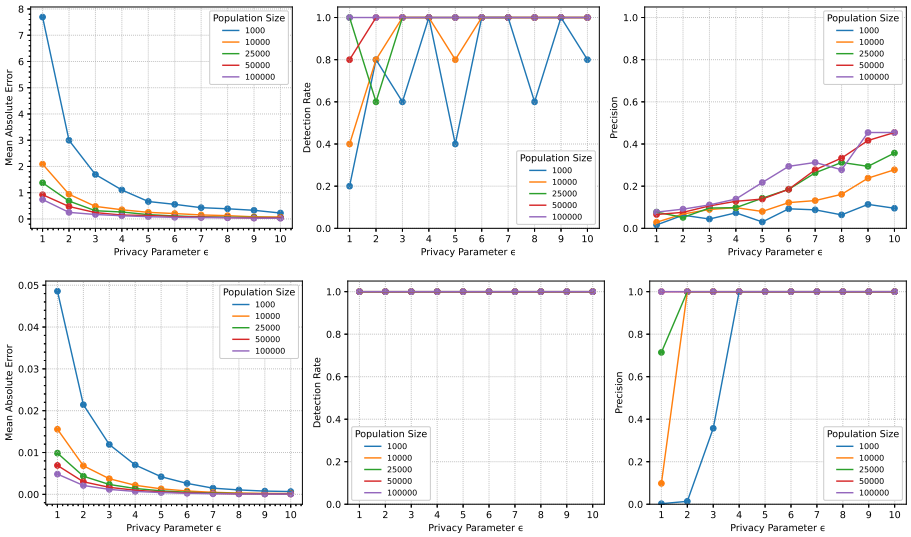


**Fig. 6.** Performance of the Exponential Mechanism in Sc.3 (top) and Optimized Local Hashing in Sc.2 (bottom) on the RPM dataset

of its bad performance, not the EM should be used to infer malicious CAN IDs, but OUE and OLH. These two algorithms showed a good performance in all scenarios even with low $\epsilon$ values and small population sizes as seen in Fig 5 and 6 in Scenario 2.

An appealing property of using PRR for the DATA field is its simplicity. However, the identification of a suspicious payload value using the mean of each individual bit can only succeed if the malicious payload dominates all other payloads: Even when using unperturbed data, the frequency of the malicious payload needs to be above 0.5 in order to be able to identify it in any case. Even one estimation that is on the wrong side of 0.5 makes the identified heavy hitter a false positive. Thus, PRR in the way it is used here is not a robust way to identify a malicious payload. This is also reflected by our experimental results, as PRR was not able to identify any malicious payload in Scenario 1 and 2. Only in Scenario 3, where *all* frames contain the malicious payload, this payload could be identified. Still, the identification with our proposed upper bound of $\epsilon = 3$ only had a good success rate for a population size of 25000 and more. Even in Scenario 3, the PRR algorithm would have almost no chance in identifying a malicious payload if multiple payloads are part of the attack. Thus, we do not recommend using PRR for the DATA field, but PEM, which showed a consistently good performance in all Scenarios. PEM is more robust, can detect multiple values and determine frequencies while outperforming PRR in all scenarios. The performance in Scenario 2 can be compared in Fig. 7.

On the Fuzzy dataset, the errors achieved in Scenario 2 and 3 show that it

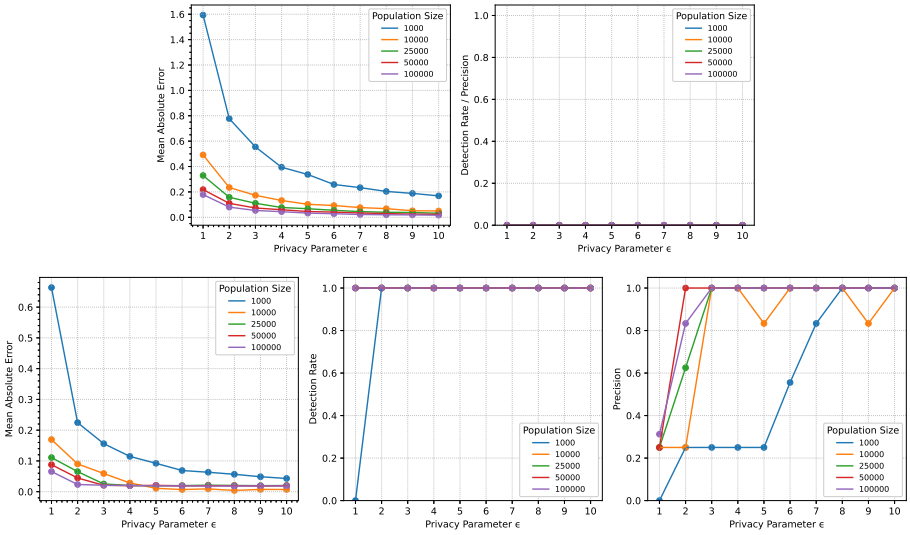


**Fig. 7.** Performance of the Parallel Randomized Response (top) and Prefix Extending Method (bottom) on the RPM dataset in Sc. 2

is still possible to closely estimate the frequencies of IDs and payload values even if random data are injected into the anomaly logs. However, these results do not carry much more information than the fact that a fuzzy attack is going on.

**Limitations.** The main limitation of our results is that the attacks and thus the identification tasks that are associated with the dataset parts "DoS", "Gear" and "RPM" are relatively simple and similar: In each of these attacks, messages with one single CAN ID are injected, and each injected message carries the same payload. This likely is the reason why the performance of the algorithms was extremely similar on these three dataset parts. Thus, the performance of the used algorithms in more complicated situations, for example if the input data are mixed from multiple datasets is of interest. In a trial on a mix of 25% data from the RPM part, 25% of the Gear part and 50% of the Fuzzy part in Scenario 3 with a population size of 10000 and $\epsilon = 3$, PEM correctly identified both malicious payloads associated with the RPM and Gear parts. Two other payload values were identified with frequencies below 5% and would thus be discarded in our analysis. This result indicates the effective applicability of our approach even if data from different attacks are mixed up at the aggregator. However, this needs to be evaluated more thoroughly to be confirmed. Overall, the results show that it is indeed possible to transmit anomaly logs to a backend while using LDP algorithms to provide privacy guarantees. It is possible to recover relevant information about the attacks that caused the creation of the transmitted anomaly logs from the aggregated data. In particular, promising results were achieved with OUE, OLH and PEM. We showed possibility of the effective

recovery of basic pieces of information about the anomalies from the perturbed anomaly data, namely the CAN IDs and payloads used in the attacks.

## 6 Conclusion

In this work, we highlight the tension between cybersecurity for In-Vehicle Networks (IVN) and privacy implications of transferring CAN logs to remote server for further analysis. We propose applying Local Differential Privacy (LDP) techniques to CAN anomaly logs as a means to lessen such a tension. LDP techniques perturbe IVN data before transfer to a remote collector and provide strong, formal privacy guarantees. The underlying architecture here is an adaptation of a generic deployment of an In-Vehicle IDS that ressembles approaches from the industry. In contrast to the widespread In-Vehicle IDS deployment model, our architecture and model aim to protect the privacy of individuals even if IVN data is transferred to the backend. The proposed model includes the perturbation of the anomaly logs using LDP techniques on the vehicle side, and an aggregation step of multiple perturbed logs on the backend side. The proposed adaptations for the transfer of anomaly logs to the backend server using LDP algorithms are generic enough to cover various scenarios for In-Vehicle IDS. We performed an experimental evaluation of our proposal in three different attack scenarios. Our evaluation aimed at assessing whether it is possible to ensure privacy via LDP while preserving the ability to learn useful information about ongoing attacks from the collected IVN data. In our underlying scenarios, anomaly logs consist of varying amounts of frames from the CAN Bus. Using various LDP algorithms and a widely used CAN intrusion dataset, we showed that it is possible to recover useful information about attacks from the perturbed anomaly logs while preventing re-identification. Our results have promising implications for the possibility of privacy-preserving IDS for In-Vehicle Networks, even though they are currently limited to relatively simple pieces of information about attacks that are recovered.

### 6.1 Further Research

As part of future work, it may be interesting to conduct a complete evaluation of more sophisticated scenarios including mixed attacks. Another interesting topic could be a thorough investigation of the impact of continuous reporting of anomaly logs on privacy degradation and lowering this impact. Investigating how users may choose an own $\epsilon$ is another item for future research.

## Acknowledgments

This research work has been funded in part by the German Federal Ministry of Education and Research (BMBF) and the Hessen State Ministry for Higher Education, Research and the Arts within their joint support of the National

Research Center for Applied Cybersecurity ATHENE, and co-funded by the BMBF within the project "Forum Privatheit und selbstbestimmtes Leben in der Digitalen Welt".

## References

1. Apple: Apple carplay, https://www.apple.com/ios/carplay
2. Boehner, M.: Security for connected vehicles throughout the entire life cycle. ATZ-electronics worldwide **14**(1-2), 16–21 (2019)
3. Cai, Z., Wang, A., Zhang, W., Gruffke, M., Schweppe, H.: 0-days & mitigations: Roadways to exploit and secure connected bmw cars. Black Hat USA (2019)
4. Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., Koscher, K., Czeskis, A., Roesner, F., Kohno, T., et al.: Comprehensive experimental analyses of automotive attack surfaces. In: USENIX Security Symposium. vol. 4, pp. 447–462. San Francisco (2011)
5. Cho, K.T., Shin, K.G.: Fingerprinting electronic control units for vehicle intrusion detection. In: Proceedings of the 25th USENIX Conference on Security Symposium. p. 911–927. SEC'16, USENIX Association, USA (2016)
6. Coroama, V.: The smart tachograph – individual accounting of traffic costs and its implications. pp. 135–152 (05 2006). https://doi.org/10.1007/11748625_9
7. Dwork, C.: Differential privacy: A survey of results. In: International conference on theory and applications of models of computation. pp. 1–19. Springer (2008)
8. Enev, M., Takakuwa, A., Koscher, K., Kohno, T.: Automobile driver fingerprinting. Proceedings on Privacy Enhancing Technologies **2016**(1), 34–50 (2016)
9. ESCRYPT: Flyer: Intrusion detection and prevention (5 2018), https://www.escrypt.com/sites/default/files/2018-08/ESCRYPT_Flyer_IDPS_Web.pdf
10. European Commision: The interoperable eu-wide ecall, https://ec.europa.eu/transport/themes/its/road/action_plan/ecall_en
11. Fanti, G., Pihur, V., Úlfar Erlingsson: Building a rappor with the unknown: Privacy-preserving learning of associations and data dictionaries. Proceedings on Privacy Enhancing Technologies (PoPETS) **issue 3, 2016** (2016)
12. Gao, X., Firner, B., Sugrim, S., Kaiser-Pendergrast, V., Yang, Y., Lindqvist, J.: Elastic pathing: Your speed is enough to track you. In: Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing. p. 975–986. UbiComp '14, Association for Computing Machinery, New York, NY, USA (2014). https://doi.org/10.1145/2632048.2632077
13. Google: Android auto., https://www.android.com/auto
14. Hacking and Countermeasures Research Lab: Car-hacking dataset, https://ocslab.hksecurity.net/Datasets/CAN-intrusion-dataset
15. Hoppe, T., Kiltz, S., Dittmann, J.: Security threats to automotive can networks–practical examples and selected short-term countermeasures. In: International Conference on Computer Safety, Reliability, and Security. pp. 235–248. Springer (2008)
16. IBM: Diffprivlib: The ibm differential privacy library (version 0.2.1) (2020), https://github.com/IBM/differential-privacy-library
17. Kairouz, P., Oh, S., Viswanath, P.: Extremal mechanisms for local differential privacy. In: Advances in Neural Information Processing Systems 27, pp. 2879–2887 (2014)
18. Lawson, P., McPhail, B., Lawton, E.: The connected car: Who is in the driver's seat? a study on privacy and onboard vehicle telematics technology (2015)

19. McSherry, F., Talwar, K.: Mechanism design via differential privacy. In: 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07). pp. 94–103 (Oct 2007). https://doi.org/10.1109/FOCS.2007.66
20. Miller, C., Valasek, C.: Remote exploitation of an unaltered passenger vehicle. Black Hat USA **2015**, 91 (2015)
21. Nie, S., Liu, L., Du, Y.: Free-fall: Hacking tesla from wireless to can bus. Briefing, Black Hat USA pp. 1–16 (2017)
22. Nowdehi, N., Aoudi, W., Almgren, M., Olovsson, T.: Casad: Can-aware stealthy-attack detection for in-vehicle networks. arXiv preprint arXiv:1909.08407 (2019)
23. Nürnberger, S., Rossow, C.: vatican – vetted, authenticated can bus. In: Gierlichs, B., Poschmann, A.Y. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2016. pp. 106–124. Springer Berlin Heidelberg, Berlin, Heidelberg (2016)
24. Radu, A.I., Garcia, F.D.: Leia: A lightweight authentication protocol for can. In: Askoxylakis, I., Ioannidis, S., Katsikas, S., Meadows, C. (eds.) Computer Security – ESORICS 2016. pp. 283–300. Springer International Publishing, Cham (2016)
25. Robert Bosch GmbH: CAN Specification, Version 2.0
26. Robert Bosch GmbH: Can with flexible data-rate, specification version 1.0
27. Schoitsch, E., Schmittner, C., Ma, Z., Gruber, T.: The need for safety and cybersecurity co-engineering and standardization for highly automated automotive vehicles. In: Advanced Microsystems for Automotive Applications 2015, pp. 251–261. Springer (2016)
28. Seo, E., Song, H.M., Kim, H.K.: Gids: Gan based intrusion detection system for in-vehicle network. In: 2018 16th Annual Conference on Privacy, Security and Trust (PST). pp. 1–6 (Aug 2018). https://doi.org/10.1109/PST.2018.8514157
29. Song, H.M., Woo, J., Kim, H.K.: In-vehicle network intrusion detection using deep convolutional neural network. Vehicular Communications **21**, 100198 (2020)
30. Tesla: Introducing software version 10.0, https://www.tesla.com/blog/introducing-software-version-10-0
31. 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: 2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W). pp. 231–238 (June 2018). https://doi.org/10.1109/DSN-W.2018.00069
32. Wang, T., Li, N., Jha, S.: Locally differentially private heavy hitter identification. IEEE Transactions on Dependable and Secure Computing pp. 1–1 (2019)
33. Wang, T.: Implementation of optimized local hashing (olh) (2019), https://github.com/vvv214/LDP_Protocols/blob/master/olh.py
34. Wang, T., Blocki, J., Li, N., Jha, S.: Locally differentially private protocols for frequency estimation. In: 26th USENIX Security Symposium (USENIX Security 17). pp. 729–745 (Aug 2017), https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/wang-tianhao
35. Wolf, M., Weimerskirch, A., Paar, C.: Security in automotive bus systems. In: Proceedings Of The Workshop On Embedded Security In Cars (ESCAR) (2004)
36. Yaron Galula, M.B.: Combining the strengths of Elektrobit's SecOC with argus IDPS. Tech. rep., Elektrobit Automotive GmbH (2017), https://www.elektrobit.com/tech-corner/combining-strengths-ebs-secoc-argus-idps/