Investigating the Impact of Evasion Attacks Against Automotive Intrusion Detection Systems

Paolo Cerracchio, Stefano Longari, Michele Carminati, and Stefano Zanero
Politecnico di Milano
paolo.cerracchio@mail.polimi.it stefano.longari@polimi.it, michele.carminati@polimi.it, stefano.zanero@polimi.it

***Abstract*—The evolution of vehicles has led to the integration of numerous devices that communicate via the controller area network (CAN) protocol. This protocol lacks security measures, leaving interconnected critical components vulnerable. The expansion of local and remote connectivity has increased the attack surface, heightening the risk of unauthorized intrusions. Since recent studies have proven external attacks to constitute a real-world threat to vehicle availability, driving data confidentiality, and passenger safety, researchers and car manufacturers focused on implementing effective defenses. intrusion detection systems (IDSs), frequently employing machine learning models, are a prominent solution. However, IDS are not foolproof, and attackers with knowledge of these systems can orchestrate adversarial attacks to evade detection. In this paper, we evaluate the effectiveness of popular adversarial techniques in the automotive domain to ascertain the resilience, characteristics, and vulnerabilities of several ML-based IDSs. We propose three gradient-based evasion algorithms and evaluate them against six detection systems. We find that the algorithms' performance heavily depends on the model's complexity and the intended attack's quality. Also, we study the transferability between different detection systems and different time instants in the communication.**

I. Introduction

Modern vehicles heavily depend on Electronic Control Units (ECUs) for various safety-critical functions, ranging from cruise control to infotainment. These ECUs communicate using the CAN bus, the standard for in-vehicle communication. A significant security concern is the lack of encryption and authentication in the CAN protocol, making it vulnerable to attacks. This issue was prominently highlighted in the 2015 by Miller and Valasek, who remotely controlled a Jeep Cherokee [21], [22]. Additionally, vehicles now connect with external networks via technologies such as Bluetooth and 5G, increasing exposure to external threats. A common solution to contrast these threats are Intrusion Detection Systems (IDS). IDSs monitor bus traffic to detect potential intrusions, mainly through: frequency-based detectors, analyzing patterns like packet timing or sequence, and payload-based detectors, examining packet content for irregularities. Machine Learning (ML) models are increasingly employed for this purpose due to their ability to handle large, diverse data sets, as discussed in Rajapaksha et al. and Longari et al. [28], [19]. However, these ML systems are vulnerable to adversarial examples [31], deliberately crafted inputs that cause misclassifications. Current research on adversarial threats, however, mainly focuses on computer vision, with relatively few specialized studies in the automotive sector.

This paper aims to ascertain the feasibility of existing adversarial evasion attacks against automotive IDSs; specifically, we aim to test the resilience of relevant deep learning (DL) models against gradient-based attacks adapted to the automotive domain. To this end, we hypothesize a knowledgeable malicious actor, capable of impersonating an electronic control unit (ECU) on the network, attempting to exploit these advantages by morphing his predefined attack sequence to simultaneously inject it and remain undetected. The attacker iteratively queries a copy of the target IDS, if it does not detect the current intrusion, they can proceed with the injection, otherwise, they apply a perturbation according to one of three novel algorithm variants (two derived from basic iterative method (BIM) [13] and one from [23]) and repeat this process.

We conduct our experiments on a dataset of real CAN traffic augmented with synthetic attacks to explore the effectiveness of the existing evasion attacks against payload-based IDSs, considering an attacker with different capabilities and knowledge of the target IDS under attack. In particular, we test a white-box scenario where the attacker has full knowledge of the target IDS, a grey-box scenario where the attacker has no access to the target IDS but can exploit a surrogate model trained on the same dataset, and finally we evaluate the feasibility of precomputing the adversarial samples to inject them when given requirements on bus traffic are met. In addition, we assess the effectiveness of the adversarially perturbed attacks on the vehicle by comparing the shape of the original and adversarial attacks signals. We find evasion possible but challenging in all scenarios, given the domain constraints and the need to keep the temporal coherence of the original packets through the perturbations. Most notably, we also discover that when attacking a complex model with slow algorithm convergence, a white-box attack could perform worse than a transfer attack originally targeting a simpler substitute model.

Our contributions are the following:

- We design and implement three variants of the popular BIM [13] and DeepFool [23] evasive perturbation algorithms, customized to handle CAN signals;
- We implement and benchmark six different IDSs based on state-of-the-art designs against a publicly available dataset;

Symposium on Vehicles Security and Privacy (VehicleSec) 2024
26 February 2024, San Diego, CA, USA
ISBN 979-8-9894372-7-6
https://dx.doi.org/10.14722/vehiclesec.2024.23056
www.ndss-symposium.org

- We evaluate the transferability and re-usability of the newly introduced adversarial techniques to explore the feasibility of such attacks under different conditions.

II. CAN security primer

The CAN protocol [11] is the industry standard for intra-vehicle communication. The significant features that led to its widespread use are low cost, high interference resilience, robust error detection, and handling of many short messages with a multi-master system suited for real-time applications. CAN packets are up to 108 bits long and have fixed structure. Only the *ID field* (11 or 29 bits), identifying the sending interface, and the *Data field* (up to 64 bits), containing the actual payload of the message, are relevant for the purpose of this paper. In standard CAN traffic, packets with a given ID are always sent by the same ECU and carry information regarding the same signals. They are commonly sent periodically by the ECU with updated signal values. We refer the interested reader to the CAN standard for further details [4].

Security challenges. CAN specification is over 30 years old and does not implement any intrinsic security mechanism: it does not natively support neither encryption nor authentication, it lacks defenses against remote access attacks and, nowadays, it handles an unforeseen variety of data related to the movement, position and general status of the car [1]. An attack can take place after a malicious actor has physical access to the on-board diagnostics (OBD) port or remote access through any vulnerability in the multitude of connected applications that run on modern cars. Common attack categories are injection, drop, and masquerade attacks.

Injection attacks consist in sending CAN data frames with crafted ID and payload. Through injection attacks, it is possible to implement a denial of service of the network by flooding it with high priority messages or to implement data manipulation attacks by spoofing ID and payloads of other ECUs.

Drop attacks require the attacker to have injection capabilities, but also to either exploit bootloops or diagnostic features, or to have control of the bus at bit level [8], [17], and consist in forcing an ECU to stop sending its own CAN packets.

Masquerade attacks require the attacker to be capable of implementing both drop and injection attacks, and consist in replacing an existing packet with one with the same ID but with a modified payload, thus appearing as the same packet with different signal values. This means that detection based on frequency cannot recognize such attacks. The necessity to mask injection attacks stems from the presence of both the tampered and the legitimate packet on the bus, which - especially for safety-related tasks - leads to the ECU discarding both packets and ignoring the command [22].

III. Related Work

Intrusion detection is a wide research topic deeply studied in the past literature [9]. We refer the reader to [14] for an overview of automotive IDSs. In this section, we focus on relevant works in the adversarial machine learning field.

Adversarial machine learning is the branch of machine learning (ML) that studies attacks against ML algorithms. Goodfellow et al. [31] proved that it is possible to induce a classification error by applying small perturbations to the input of a deep learning model in many realistic settings; inputs crafted in this way are called *adversarial examples* [10]. These techniques could be exploited by a malicious attacker in several ways and with different goals [3]: **(a)** Exploratory attacks are an attempt to investigate a model that appears as a black-box, probing its response to different inputs to extract as much knowledge as possible, **(b)** Evasion attacks are the most common and were the first to be investigated. The adversary tries to adjust the inputs they want to inject to cause a misclassification. **(c)** Poisoning attacks are any kind of contamination of the training data. In many real-world cases, this implies evading some check, generally by human experts or by a previous version of the system under attack, to cause a mislabeling when included in a later training dataset. This research field, originated from the field of computer vision, has also been applied to security-critical applications dealing with network or transaction monitoring and malware recognition [30], [25]. Previous research has shown interesting properties of adversarial examples: Papernot et al. [26] showcase the transferability of adversarial attacks by training an *oracle* – a substitute network – to attack instead of the actual target. Longari et al. [16] design such an oracle-based approach for our domain of interest, developing a greedy algorithm for a black-box adversarial attack on automotive IDSs; however, this work resorts to Hamming distance to evaluate the distortion introduced in the perturbed packets, not fully capturing the actual semantic distance and do not consider attackers with different degrees of knowledge of the target system. An immediate way to improve the resilience of DL models in supervised or semi-supervised settings is adversarial training, i.e. the inclusion of labeled adversarial examples in the training set [33]. Another solution is to select resilient input features, Papernot et al. [27] suggest training a first model and then approximating it with a second one, made more resilient thanks to the knowledge deriving from the confidence scores and class similarity insights produced by the first. In network intrusion detection, Li et al. [15] attack the in-vehicle Ethernet monitored by an long short-term memory (LSTM) IDS classifier [12] with fast gradient sign method (FGSM) and BIM, resulting in a recall score of at most 2%; then the authors retrain the LSTM including adversarial examples, thus approaching the baseline attack-free score ($\sim$ 98%). Similarly, Sauka et al. [29] performs several tests against FGSM, projected gradient descent (PGD), and , successfully mitigating the attack through adversarial training.

IV. Motivation and Threat Model

In the automotive domain, the implications of machine learning models can directly influence the safety, efficiency, and operational reliability of vehicles. With the rapid integration of AI technologies for intrusion detection, autonomous driving, predictive maintenance, and personalized in-car experiences, the security of these models has become a pressing concern. Adversarial machine learning attacks pose a unique threat to these systems, potentially disrupting their functioning and leading to dire consequences. Studying adversarial machine learning attacks in this context is, therefore, a critical necessity, enhancing our understanding of potential vulnerabilities and, consequently, contributing to building safer, more reliable automotive systems. More precisely, our interest is to analyze the response of modern IDSs in the worst-case scenario of a capable and knowledgeable attacker in order to understand the robustness of existing solutions. From an offensive standpoint, the possibility to algorithmically produce evasive examples has an undeniable appeal and would represent a critical vulnerability in DL models. As progressively more properties of evasive examples emerge in computer vision applications, our goal is to understand their transferability to the automotive domain.

This paper advances prior techniques for creating adversarial examples on CAN and addresses unique challenges in its specific field. Unlike previous studies where adversarial examples are produced offline without considering the ongoing data stream, this study focuses on the online creation of adversarial samples. These samples must align with the requirements of the vehicle data stream, meaning they should adapt to the vehicle's dynamics. Additionally, they should retain characteristics of typical automotive cyber-attacks, which usually involve several harmful packets to be effective. Essentially, this study focuses on developing stealthy attacks that take into account the history of transmitted data up to the point of attack and involve the use of multiple disguised attack packets.

Attacker Model. We consider a single CAN channel monitored by a network IDS, where the intent of the attacker is to inject a sequence of malicious packets to achieve a range of effects while avoiding detection. The attacker has either white- or grey-box knowledge of the target IDS[1]. With full white-box access, the attacker has full knowledge of the IDS: model architecture, training data, and parameters. The attacker can also compromise any ECUs on the bus. Thus, given the lack of inherent security mechanisms, they can inject arbitrary messages at any given instant and can intercept inbound and outbound communications, effectively eavesdropping the whole traffic on the channel. In the grey-box scenario, the attacker has no access to the target IDS but has knowledge of the training data and exploit surrogate machine learning models trained on the same dataset. In brief, the adversarial strategy consists in perturbing the predetermined set of objective malicious sequences with our proposed algorithms, trying to morph existing samples into evasive examples. For the purpose of this evaluation, we make the following assumptions:

1. We assume an attacker with complete knowledge of the system and the data transmitted on CAN, that - given the expected behavior of the nodes - can predict the packet sequences being generated by the victim node, and query multiple times the oracle IDSs in advance to obtain the adversarial sample sequence. We provide an evaluation of the realism of such scenario in Experiment VI-F.

2. The attacker has complete control over the compromised ECUs, impersonating it. This is not unreasonable to assume given the wide attack surface and the established methods to perform complex attacks on CAN;

3. Since car manufacturers are usually very secretive about the semantics of CAN messages, we will exploit reverse engineering methods to extract signals in the traffic and treat them as the intended values.

[1]Black box approaches are already discussed in [16].

V. Approach

In our approach, the attacker starts with an initial CAN log containing some unperturbed injected messages as a baseline, then her actual strategy is to morph these intended sequences to be evasive by applying repeated modifications. An evasion attack is the act of finding a subtle perturbation of the input to cause its misclassification; we can formalize it more rigorously as the optimization problem of finding the adversarial sample $\tilde{x}$, undetected under the discrimination function $F(x)$ while minimizing the perturbation $\delta(\tilde{x}, x)$:

$$\tilde{x} = argmin_{x^*}[\delta(x^*, x)] \quad \text{s.t. } F(x^*) = 0 \wedge x^* \in D_x \tag{1}$$

In almost every application, the produced $\tilde{x}$ also needs to satisfy some domain constraint ($x^* \in D_x$). For example, in the computer vision domain, it is customary for $\tilde{x}$ to contain pixel intensities in the integer range $[0, 255]$. On the other hand, when dealing with tabular data, features become interconnected and heterogeneous, with an intricate valid problem space, and as a consequence finding a suitable adversarial perturbation becomes harder [2], [7].

The algorithms we propose are derived from well-known gradient-based techniques originally designed for computer vision; the common rationale behind these kinds of techniques, namely fast gradient method (FGM), BIM [13], and [23], is to leverage the backpropagation algorithm against the model under attack. The intuition is to push the original input towards areas in the problem space with lower confidence, thus approaching and eventually crossing the decision boundary by exploiting the gradient ascent of an arbitrary loss function. As it is uncommon for a malicious actor to have access to the model for gradient computation, researchers developed black-box techniques that attempt to fool a victim via iterative queries aimed at approximating the discriminator in some measure, like in the zeroth order optimization (ZOO) [6] and HopSkipJump [5]. Figure 1 illustrates a single iteration of the process. At each iteration, the chosen algorithms try to find an optimal additive term and produce x^{t+1}.

A. BIM-based algorithms

The original BIM iteratively applies the FGSM perturbation described by (2).

$$x^{t+1} = clip_X(x^t - \epsilon \frac{\nabla_x L(w, x^t)}{||\nabla_x L(w, x^t)||}) \tag{2}$$

From a geometric point of view, this method produces a perturbation directed towards a maximum of the loss function L via approximation of the gradient $\nabla_x L$ and constrained by $||x^{t+1} - x^t||_\infty \leq \epsilon$, depending on the hyperparameter ϵ.

We implement two slightly different variants of the BIM attack, namely the *step decay* BIM and the $l2$ BIM; both versions of the algorithm include an adjustment procedure to restrict the resulting value x^{t+1} to stay in the problem space. We achieve this by clipping and rounding to the nearest integer so that the features can be effectively represented in the actual underlying bit vector. Besides this modification, the *step decay* method differs from the simple BIM in the parameter ϵ: in our implementation, it has not a fixed magnitude but rather a geometrically decreasing value according to the update rule

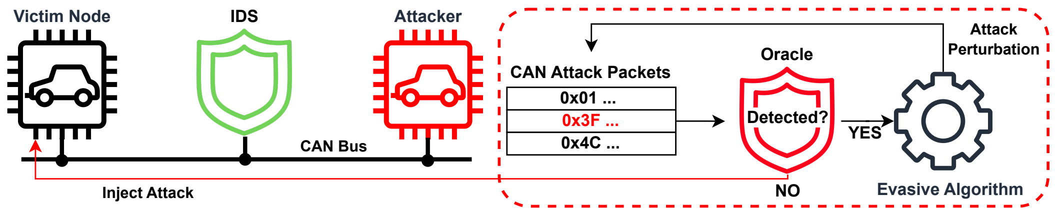


Fig. 1: Scheme of the attacker's approach.

Algorithm 1: Proposed BIM step decay variant

```
begin
    $t \leftarrow 0$;
    $step \leftarrow \epsilon$;
    while $t < max\_iter$ do
        $score \leftarrow \texttt{get\_score}(ids\_model, sample)$;
        $grad \leftarrow \texttt{gradient}_{sample}(score)$;
        $pert \leftarrow step \cdot sign(grad)$ ;                    // (2)
        $pert \leftarrow -pert \cdot tamper\_mask$ ;       // minimizing
        $sample \leftarrow \texttt{clip}_{min,max}(sample + pert)$;
        $sample \leftarrow \texttt{round}(sample)$;
        $step \leftarrow step \cdot decay$ ;                 // step decay
        if $get\_score(ids\_model, sample) < threshold$ then
            return $sample$ ; // sample is now evasive
        end
        $t \leftarrow t + 1$;
    end
    return $None$ ;                     // Abort computation
end
```

Algorithm 2: Proposed $l2$ BIM variant

```
begin
    $t \leftarrow 0$;
    $step \leftarrow \epsilon$;
    while $t < max\_iter$ do
        $score \leftarrow \texttt{get\_score}(ids\_model, sample)$;
        $grad \leftarrow \texttt{gradient}_{sample}(score)$;
        $pert \leftarrow step \cdot \frac{grad}{||grad||_2}$ ;                    // (3)
        $pert \leftarrow -pert \cdot tamper\_mask$ ;       // minimizing
        $sample \leftarrow \texttt{clip}_{min,max}(sample + pert)$;
        $sample \leftarrow \texttt{round}(sample)$;
        if $get\_score(ids\_model, sample) < threshold$ then
            return $sample$ ; // sample is now evasive
        end
        $t \leftarrow t + 1$;
    end
    return $None$ ;                     // Abort computation
end
```

Algorithm 3: Pseudocode for the DeepFool variant

```
begin
    $t \leftarrow 0$;
    $step \leftarrow \epsilon$;
    while $t < max\_iter$ do
        $score \leftarrow$
          $\texttt{get\_score}(ids\_model, sample) - threshold$;
        $grad \leftarrow \texttt{gradient}_{sample}(score)$;
        $pert \leftarrow \frac{score \cdot grad}{||grad||_2^2}$ ;                    // (4)
        $pert \leftarrow (1 + \epsilon) pert \cdot tamper\_mask$ ;
          // projection
        $sample \leftarrow \texttt{clip}_{min,max}(sample + pert)$;
        $sample \leftarrow \texttt{round}(sample)$;
        if $get\_score(ids\_model, sample) < 0$ then
            return $sample$ ; // sample is now evasive
        end
        $t \leftarrow t + 1$;
    end
    return $None$ ;                     // Abort computation
end
```

$\epsilon^{t+1} = \epsilon^t * \omega$, introducing the hyperparameters ϵ_0 and ω. The reason for the introduction of *step decay* is that, similarly to what happens with the learning rate decay during the training of DL models [34], it may allow to reduce the number of iterations by avoiding oscillation around an evasive point in case of a highly non-linear decision boundary and big ϵ values, possibly also resulting in a smaller perturbation.

$$x^{t+1} = clip_X(x^t - \epsilon \frac{\nabla_x L(w, x^t)}{||\nabla_x L(w, x^t)||_2}) \tag{3}$$

Conversely, the $l2$ BIM simply uses the Euclidean norm instead of the absolute value for the FGSM equation, resulting in (3). In this case, ϵ quantifies the module of the perturbation vector, which now completely orients itself according to the gradient and constitutes the tighter bound $||x^{t+1} - x^t||_2 \leq \epsilon$. We deem this approach more suitable in our case - given the higher dimensionality and feature inter-correlation in our domain of interest - but also in general when dealing with diverse features, as they apply small perturbations along all dimensions. In both implementations, we choose as loss function the opposite of the anomaly score.

B. DeepFool-based algorithm

[23] is another iterative method that approximates the IDS as an affine classifier $w \cdot x + b = F(w, x)$, then the perturbation δ tries to push the sample beyond the affine decision boundary, which we assume to be the 0 plane for F.

$$x^{t+1} = x^t + (1 + \epsilon) \cdot (-F(w, x) \frac{\nabla_x F(w, x^t)}{||\nabla_x F(w, x^t)||_2^2})) \tag{4}$$

(4) illustrates the original computation, notice the overshooting factor $1 + \epsilon$: since the algorithm can't converge to a point precisely on the decision boundary, we attempt to cross it by a small value ϵ instead - according to the approximation - and cause the objective misclassification. In our case, we do not deal with a sign-dependent decision value like $F(w, x)$, however, we can consider the reconstruction error $L(w, x)$ as a classification confidence score and apply a shift by the thresholding value θ to obtain an analogous zero-centered boundary $F(w, x) = L(w, x) - \theta$. We apply the same adjustment procedure to obtain valid samples that we use for the BIM algorithms. This may hinder the algorithm convergence; however, we mitigate this phenomenon by testing different overshooting magnitudes. According to the evaluation of the original paper, albeit more computationally complex, should terminate in fewer iterations than simple BIM. The

input windows contain malicious and legitimate packets for all the models under test except the feed-forward neural network (FFNN). Therefore, we apply the computed perturbation only on the injected packages at each step via a simple projection of the computed perturbation matrix. Also, note how the algorithm interacts with the predictive detection approach: the attacker morphs the target output, i.e., the most recent packet, to be closer to the prediction. In case of successful evasion, the input window will then slide to include the generated packet, influencing the classification.

VI. Experimental Evaluation

We aim to answer the following research question: *What is the effectiveness of existing adversarial evasion attacks against payload-based automotive IDSs?*. In the first experiment, we evaluate a scenario in which an ideal attacker has full knowledge of the target IDS, of the nodes behaviour, and the data transmitted on CAN (i.e., white box scenario). Consequently, they can predict the network traffic flow being generated by the victim node and coherently inject attack sequences. This experiment allows us to explore the effectiveness of such attacks in the worst-case scenario. In the second experiment, we assess the effectivness of the adversarially perturbed attacks on the vehicle by comparing the shape of the original and adversarial attacks signals. In the third experiment, we restrict the assumptions on the attacker's knowledge; they have no access to the target IDS, but they can exploit a surrogate model trained on the same dataset to generate evasive samples. In other words, we evaluate the transferability of our attacks in a grey-box scenario. Finally, we restrict the assumption of the attacker's computational capabilities. To emulate the physical real-time constraints, we precompute the evasive payload sequence allowing an attacker to inject it entirely when given requirements on bus traffic are met.

A. Experimental settings

Signal extraction and preprocessing. As suggested in previous works [35], [19], the packet payloads undergo a heuristic analysis, derived with slight improvements from the reverse engineering of automotive data frames (READ) method [20]. This procedure allows us to identify bit ranges with distinct semantic meanings. The analysis exploits the rate at which the bits sent by a ECU change to infer information on the fields, such as discriminating adjacent bits that act as a counter or binary flags that are not tightly correlated with neighbor bits. We process the attack-free data and identify five range categories, summarized in Table I. We confirm previous research results [18] related to using only physical and binary ranges as features: this has shown to bring benefits in terms of both reconstruction performance and model complexity in all our baseline models when compared with directly feeding the vector of non-constant payload bits. Specifically, we parse the physical signals as integers and normalize them into the $[0, 1]$ interval by dividing each sample by the largest representable number with the corresponding bit vector.

Dataset. For our evaluation, we use the *C-1 ReCAN* dataset, containing real attack-free traffic from a Giulia Veloce car [35]. It comprises about two hours of city and highway driving logs. However, it lacks instances of message injection. We generate

TABLE I: Bit ranges with semantic meaning identified

Type	Description
Constant ranges	Bits that stay constant across all considered traffic and are excluded from the final feature vector representing the frames.
Physval ranges	Sequences of bits that contain some kind of values, usually corresponding to a physical signal like speed or wheel position.
Binary ranges	Ranges containing an isolated, non-constant bit, interpreted as a logical flag.
CRC ranges	Application-level checksums for message integrity, detected through their Gaussian random behavior.
Counter ranges	Application-level counters, increasing by one with each subsequent frame.

synthetic attacks on the ReCAN logs through the *CANtack* tool [24], performing different types of message injection to obtain the following separate augmented sets:

Injection-replay attack. It adds packets to the flow of messages with an injection rate of 0.4 – i.e., the malicious payloads are injected two and a half times slower than the average interarrival time of normal frames. The payloads are sniffed from previous legitimate sequences sent by the compromised ECU.

Full replay dataset. It is a masquerade replay attack: as before, we take the injected payload from previously sniffed traffic. This time, however, there are no packets added to the communication. Instead, the tool tampers with the data that would be normally sent to substitute the desired malicious content. All the following attacks in this list replicate such an impersonation strategy.

Continuous change dataset. A variant of the full replay attack that gradually modifies the bits of one signal in the packets to reach a (randomly) predetermined value. For each sequence of 25 packets, this strategy tries to seamlessly bring one of the identified 9-bit fields to represent an arbitrary value.

Change to minimum dataset. It is a variant of the previous one: instead of a random goal value, the target final signal is a string of zero bits.

Fuzzy dataset. Fuzzing is a random attack whose purpose is usually to probe and reverse engineer the victim system. All the cyclic redundancy checks (CRCs), counters, and constant bits are ignored in this implementation. The others, belonging to physical and binary signals, are changed at each packet with a pseudo-random value.

All the attacks are replay attacks at their core, but some additionally tamper the sniffed signals according to a specific strategy; moreover, all attacks consist of 10 sequences of 25 packets each, starting between 20 and 25 seconds from the first timestamp in the dataset. In every dataset, separate sequences are always over one minute apart, and the attacks are generated independently for each considered CAN ID; we choose 12 IDs following the reasoning and experiment allocation of CANova [24]. We set the threshold value at a percentile of the scores obtained from the thresholding set; we chose the 98th percentile for all models with a false positive rate (FPR) of at most 4.7%.

Evaluation metrics. In our experiments, we mainly make use of the two most common detection metrics, true positive rate

(TPR) and area under the curve (AUC)[2], and, to better capture the magnitude of the adversarial perturbations, we introduce the *aggregate perturbation (AP) metric* that measures the *mean maximum perturbation of a single field* in the packet. We define the *AP metric* as shown in (5) , where N is the total number of malicious packets in the test set, x_i is the array of features in the original i-th malicious packet, and $\tilde{x}_i$ is the array of features in the corresponding adversarial packet. Note that the term $||x_i - \tilde{x}_i||_\infty$ is the infinity norm representing the maximum variation introduced in a single feature of the i-th packet to make it evasive. We compute this metric from normalized values; hence it is always in the range $[0, 1]$.

$$\mathtt{AP} = \frac{\sum_{i=1}^{N} ||x_i - \tilde{x}_i||_\infty}{N} \tag{5}$$

B. Selected intrusion detection systems

Given that attacks that alter the frequency are easy to detect, an adversarial attacker will implement masquerade attacks that do not alter frequency of the network stream. Therefore, to assess the effectiveness of evasion attacks against IDSs in the automotive field, we select six commonly used [16] architectures of payload-based anomaly detector.

FFNN. A one-to-one autoencoder with two fully connected layers with 16 units each; note that this architecture is blind to replay attacks as it considers a single packet at a time, however, it is a simple solution to detect more obvious payload tampering and is included to provide a baseline reference.

CANdito [18]. A window-to-window symmetrical autoencoder with two fully connected (128 units each) and two LSTM layers (64 cells each), it is the most complex among the implemented networks.

LSTM predictors [32]. Two window-to-one predictors, we implement a short variant with just two LSTM layers having 32 cells each, and a long variant with four LSTM layers, having symmetrically 64 and 16 cells.

GRU-based predictors. Two window-to-one predictors analogous to the short and long LSTM variants, employing the more lightweight gated recurrent units (GRUs) instead.

A final sigmoid-activated dense layer with one unit per input feature follows each individual architecture to provide an output with the correct dimensionality. While the predictor models produce an anomaly score for one packet at a time, with a rolling input window, the CANdito autoencoder reconstructs the whole window, operating with non-overlapping input sequences. We choose a window size of 40 (or 39 plus one predicted frame for the predictor models).

C. Experiment 1: Ideal attacker

In this experiment, we compare the performances of all six IDSs over all the available attacks, first establishing a baseline and then morphing all the malicious frames with each of the three proposed algorithms. Tables II, III and IV show the aggregate results for the generated full replay and continuous attack on the ReCAN dataset, meaning that the shown AUC and AP values are an average over the 12 IDs. The full replay attack (see Table II) is the "hardest" among the proposed scenario for the detectors: since the packets come from legitimate traffic, the system can only leverage the semantic discontinuity in the flow of messages for its classification. In this scenario, the long predictive model performs slightly worse than the respective short version, while CANdito widely outperforms the other architectures in both baseline recall and resilience to adversarial evasion attempts. This smaller number of evasion points corresponds to smaller perturbations: we shall discuss how the algorithms succeed for many input samples that were already near the decision boundary. As expected, the continuous change (see Table III) and change to minimum attacks [3] bear almost identical evasion rates, with the former being slightly harder to detect as a baseline (with a difference of about 5% recall for the FFNN, 1% recall for CANdito and 3% recall for the predictive models). In these attacks, the intruder replays previous packets while tampering with just one signal field – with a minimum length of 9 bits –, therefore, the payload becomes progressively easier to detect the longer the attack continues. Unfortunately, this behavior also causes the proposed algorithms to strongly perturb that target signal, often resulting in an example that is very similar to the corresponding packet in the attack-free scenario, meaning that, while the evasion is successful, the intended effect is lost. The $l2$ BIM requires double the expected maximum perturbation than in the previous scenario, making the *step decay* variant preferable in many attack settings even thought it has the worst performance in terms of TPR and AUC reduction. Despite achieving good evasion rates in all the tests, the algorithm also produced unfeasibly high perturbations for the predictor models, being more suitable for the complex CANdito design. In the injection replay scenario (see Table IV), the attacker interleaves additional replayed packets to the normal traffic so that the content is analogous to the full replay sequences, but discontinuities are present at each injection. This is the only attack against which the predictors outperform CANdito in the baseline test; the experiment also stands out since the long LSTM architecture results significantly more vulnerable to adversarial perturbations than the other predictive models. Here the *step decay* algorithm is preferable since it almost matches the evasion rate of the other algorithms on the many-to-one models while keeping the AP one order of magnitude lower. The fuzzy injection is predictably the easiest to detect due to the random generation of the malicious packets, with the minimum AUC being 0.995 for the FFNN; this also makes finding a suitable perturbation harder since the average packet contains signals far from their regular distribution. The intent behind the fuzzing attack is often to investigate the interaction with the IDSs rather than to produce a specific effect.

D. Experiment 2: Perturbation effectiveness

Figure 2 shows some exemplary adversarial behaviors to provide a qualitative assessment of the effectivness of the adversarially perturbed attacks on the vehicle by comparing the shape of the original with the adversarial attacks signals In the plot, the dotted lines represent the intended content of an injected sequence while the blue lines are the results of the

[2]Note that we do not evaluate false positives since they are not affected by evasion attacks.

[3]Due to page limits, we do not report these results given the similarity to the continous change ones.

TABLE II: Full replay attack evasion results.

			FFNN	**CANdito**	**Short LSTM**	**Long LSTM**	**Short GRU**	**Long GRU**
Algorithm	Base	TPR	0.0290	0.6615	0.3950	0.3857	0.3333	0.3030
		AUC	0.5085	0.8556	0.7646	0.7429	0.7390	0.7165
	$l2$ BIM	TPR	0.0113	0.6250	0.1883	**0.1347**	0.1680	**0.1577**
		AUC	0.5063	0.8540	**0.6680**	**0.6394**	**0.6340**	**0.6631**
		AP	0.0085	0.0023	0.1165	0.1142	0.1082	0.1098
	decay BIM	TPR	**0.0100**	0.6563	0.1837	0.1750	0.1780	0.1637
		AUC	**0.5063**	0.8550	0.7489	0.7402	0.7412	0.7372
		AP	0.0159	0.0159	0.0378	0.0402	0.0398	0.0368
	DF	TPR	0.0257	**0.5729**	**0.1720**	0.1610	**0.1677**	0.1707
		AUC	0.5084	**0.8532**	0.6898	0.6516	0.6843	0.6946
		AP	0.0049	0.0076	0.4736	0.4835	0.4828	0.4990

TABLE III: Continuous change attack evasion results.

			FFNN	**CANdito**	**Short LSTM**	**Long LSTM**	**Short GRU**	**Long GRU**
Algorithm	Base	TPR	0.6617	0.9196	0.8103	0.8047	0.7970	0.8023
		AUC	0.8385	0.9774	0.9488	0.9537	0.9484	0.9509
	$l2$ BIM	TPR	**0.3077**	0.9146	0.3743	0.3367	0.3573	0.3743
		AUC	**0.7851**	0.9773	**0.8560**	**0.8312**	**0.8444**	**0.8560**
		AP	0.1177	0.0029	0.1921	0.2033	0.1821	0.1948
	decay BIM	TPR	0.3903	0.9146	0.5730	0.5880	0.5143	0.5670
		AUC	0.8035	0.9772	0.9312	0.9349	0.9290	0.9352
		AP	0.0401	0.0098	0.0526	0.0524	0.0526	0.0528
	DF	TPR	0.4203	**0.8844**	**0.3093**	**0.3113**	**0.3200**	**0.3350**
		AUC	0.8239	**0.9764**	0.8622	0.8634	0.8485	0.8745
		AP	0.1404	0.1477	0.4772	0.4974	0.4852	0.5112

TABLE IV: Injection-replay attack evasion results.

			FFNN	**CANdito**	**Short LSTM**	**Long LSTM**	**Short GRU**	**Long GRU**
Algorithm	Base	TPR	0.0303	0.7273	0.7860	0.7777	0.7783	0.7857
		AUC	0.5114	0.9001	0.9273	0.9180	0.9246	0.9210
	$l2$ BIM	TPR	0.0083	0.7193	0.4467	0.3783	0.4453	0.4330
		AUC	0.5083	0.8988	**0.8465**	**0.8221**	**0.8512**	**0.8433**
		AP	0.0136	0.0085	0.0979	0.1222	0.1046	0.1232
	decay BIM	TPR	**0.0000**	0.7246	0.4580	0.4417	0.4457	0.4553
		AUC	**0.5056**	0.8998	0.8883	0.8784	0.8849	0.8852
		AP	0.0160	0.0100	0.0364	0.0367	0.0351	0.0351
	DF	TPR	0.0243	**0.6578**	**0.4383**	**0.3437**	**0.4210**	**0.4247**
		AUC	0.5110	**0.8952**	0.9002	0.8534	0.8960	0.8946
		AP	0.0157	0.3339	0.3565	0.3588	0.3616	0.3739

adversarial perturbation and the green lines provide a baseline reference depicting the normal signal in the attack-free state; a red background highlights packets that have successfully evaded the IDS and a grey background indicates the packets that were already undetected.

Plot 2a. It captures a output sequence of the continuous change attack against the short LSTM model for CAN ID "0DE". In some cases, especially where the AP metric exceeds the 15% threshold, the evasive points are very close to the reference normal traffic that the ECU would have transmitted if it was not silenced. This is particularly true for the two continuous experiments where the attack heavily manipulates only one specific signal and replays the others: the adversarial gradient ascent correctly captures the intra-packet dependency and pushes the rogue signal to values that are consistent with the context. This behavior is of course undesirable for a malicious actor since, despite the success of the evasion attempt, the meaning of the target payload is completely lost and the result is almost identical to not carrying out any attack.

Plot 2b. It captures the $l2$ BIM output sequence of the fuzzy attack against the long LSTM model for CAN ID "100". The algorithm fails to find a suitable perturbation for several packets in the sequence; the purpose of this example is to show how introducing a single point of discontinuity has a severe impact on the classification of the predictive models. Following the correctly flagged packets that the algorithm failed to morph (with the white background) we can perturb the signal much more easily towards values similar to the last received packet. We did not observe the same behavior in CANdito, as the initial fully connected layer and the target sequence reversal force the model to not put too much weight only on the latest packets during the reconstruction, besides differing in the non-overlapping sliding window input.

Plot 2c. It captures the $l2$ BIM output sequence of the full replay attack against the long LSTM model for CAN ID "0FB". Here the evasion is successful except for two packets in the middle of the sequence. The evasion algorithm could not find a suitable configuration that allowed to continue the descent of the plotted value since the general behavior of the ECU for those bits is to have a slowly varying signal, after the first markedly monotonous messages, the IDS correctly recognized an intrusion. Among the analyzed CAN IDs some exhibited specific signals with a very regular behavior in terms of periodicity or speed of variation, making the whole ECU more vulnerable to some attack strategies than others, the

TABLE V: Comparison of the baseline TPR of CANdito with the TPRs against the best performing white-box algorithm and the LSTM transfer attack.

		Full Replay	**Continuous change**	**Continuous to Minimum**	**Fuzzy**	**Injection Re-play**
Baseline	TPR	0.6615	0.9196	0.9267	1.0000	0.7273
	AP	–	–	–	–	–
LSTM oracle $l2$ BIM	TPR	**0.5677**	**0.8543**	**0.8848**	0.9895	**0.5588**
	AP	0.1142	0.2033	0.2158	0.2936	**0.1222**
FFNN oracle $l2$ BIM	TPR	0.6615	0.9196	0.9110	1.0000	0.7272
	AP	0.0085	0.1177	0.1228	0.2111	0.0136
white-box	TPR	0.5729	0.8844	0.9162	**0.9842**	0.6578
	AP	0.0076	0.1477	0.1449	0.5508	0.2547

most notable example is ReCAN ID ’0FE’, that had very high intercorrelation and very low variability rate, making it not vulnerable to the proposed evasion approach.

E. Experiment 3: Attack Transferability

In the transferability grey-box experiment the attacker has no access to the specific model architecture, rather they use an oracle [26], a surrogate model trained on a dataset representative of the actual training dataset (in this case they are identical) to generate evasive input against in place of an unknown detection model. Therefore, the attacker tries to transfer the adversarial examples crafted for a specific IDS implementation to whatever real world system is actually operating on the target bus. We test the evasive packets generated by for the CANdito model and by the $l2$ BIM variant for the long LSTM and FFNN models against all the other available architectures. For this test, we define the transfer rate as the ratio between the evasive points that evade the target model and all the evasive points produced with the current oracle. The most notable result is that the transfer of evasive examples from the LSTM to CANdito is more successful than the direct white-box attack in all scenarios except the fuzzy one. Table V showcases this behavior, we highlight in bold the best performing evasion strategy. It is also notable that in the injection replay and fuzzy scenarios the LSTM-based adversarial examples are much closer to the original packets than the ones produced with and white-box access. We find reasonable to explain this phenomenon with the autoencoder structure of CANdito: the window-to-window behavior causes all the packets to be perturbed at once, with any modification of the input reflected into the target output; this hinders a steady convergence, resulting in very high number of iterations for all the algorithms. Albeit the complexity of the model also lowers the transfer rate from other oracles, its resilience to this type of evasion favors the transfer attack over the white-box approach.

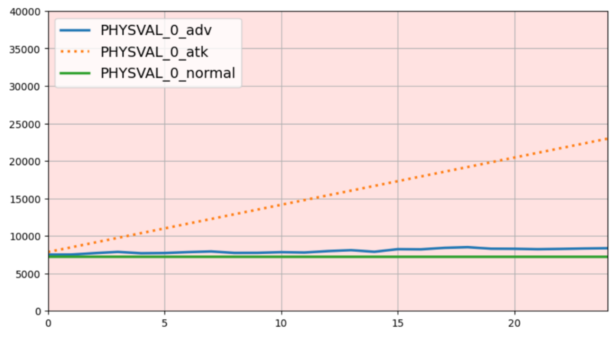
On the other hand, due to its low evasion rate and specific examples, the CANdito oracle emerges as not suitable to carry the proposed grey-box attack, with either a negligible transfer rate or even a detrimental outcome from the attacking point of view, with the evasive points resulting in higher TPRs for different IDSs. The long LSTM transfers to other predictive models on all the architectures, which behave similarly: the transfer rate of roughly 40% of the adversarial examples for the continuous change, change to minimum, fuzzy, and injection replay attacks; the evasion for the full replay attack seems more specific as we recorded a 18.3% mean transfer rate. Even the FFNN oracle succeeds against the predictive IDSs for the two continuous-type attacks, with an average transfer rate of 7.5% for the continuous change scenario and of 32.9% for the change to minimum scenario. Moreover, this oracle performed slightly better on short and GRU-based models.

F. Experiment 4: Attack Precomputation

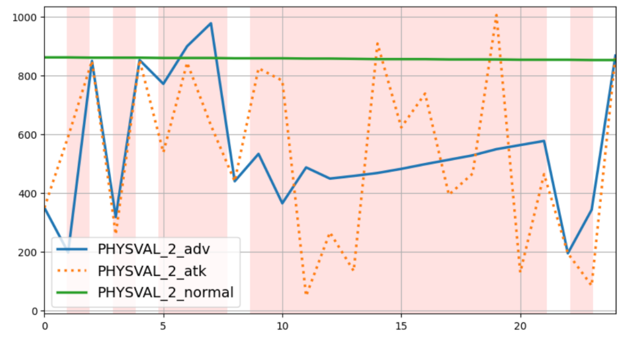
While the grey-box experiment tightens the assumption of the knowledge of the attacker, the precomputation test restricts the time constraints for the malicious injection. Since the current strategy requires repeated querying of a target autoencoder or predictor, we test whether an attacker could compute a sequence of adversarial packets in advance and successfully inject it at a later time while avoiding detection. In practice, this experiment takes all those sequences that are *fully evasive*, i.e., exclusively composed of packets classified as normal traffic, and tries to find similar points in the flow of messages where they could equally evade the IDS. We also consider as candidate injection points every point in the traffic preceded by at least ten packets identical to the preamble found at the original attack location. We exclude from this test the FFNN model, as it performs classification independently of the order or position of messages, and CANdito as, given its superior resilience, there were not enough completely evasive sequences to carry out the test. The results are widely dependent on the specific CAN ID, with two clear clusters:

Cluster 1. IDs “1FB” and “104” have very slowly varying physval signals, that for the duration of the traffic gravitate around some common values rather than assuming any possible bit configuration (albeit all configurations are valid and no bit always remains constant). This behavior causes a relatively high number of possible reinjection points, with a peak cardinality of nearly 1800 points found for a single sequence, and a high success rate. In general, sequences similar to the one in the view of Plot 2a transfer easily into spots in the traffic with a longer matching preamble, as we observe a 95% success rate with an average number of 38 identical preceding packets, obtaining several hundreds of potentially *fully evasive* sequences from 4 to 10 precomputed attacks, depending on the ID, attack, and oracle;

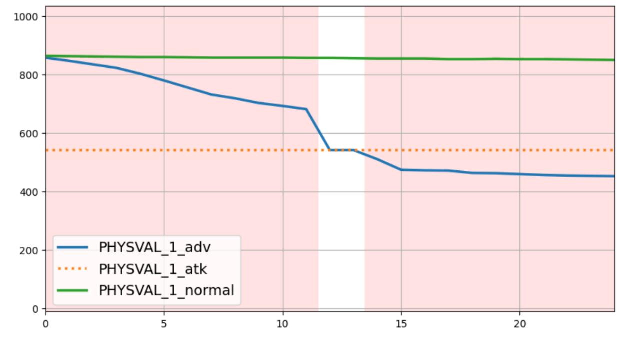
Cluster 2. The remaining 10 IDs do not bring the same degree of success as they provide way fewer injection points, with many preambles without a match in the whole traffic flow; once again, the few successful precomputed attacks require a preamble almost identical to the original, with over 37



(a)



(b)



(c)

Fig. 2: Exemplary plots of adversarially perturbed signals. On the X-axis we represent the number of frames, while on the Y-axis the value of the signal.

matching messages on average. In general, this precomputed attack is feasible and reliable only within the trivial case of a preamble that is close to the original IDS input window, with more of the 90% of the successful reinjections differing only for a couple of packets. This makes just a few specific devices among the considered ECUs vulnerable to the approach under testing, however, it is not possible to ascertain the impact of the resulting risk with the current information about the function and semantic associated with each affected CAN ID.

G. Discussion

The results of our evaluation provide insight on the resilience of the tested IDSs against adversarial attacks: we observed that the introduced adversarial techniques were partially effective in evading detection, raising concerns about the robustness and reliability of IDSs in the presence of adversarial examples in the automotive domain. In particular, we found the proposed attacks to achieve a high evasion rate against the predictive models while CANdito [18] proves to be more resilient; however, both the success and the quality of the adversarial samples highly depended on the starting payload. The difficulty in evading the IDSs is influenced by the previously mentioned constraints and specific challenges of the automotive sector. In particular, adversarial attacks require the injection of adversarial samples. In the examined context, this implies injecting streams of altered data during a transition period, where the IDS is analyzing both normal and perturbed traffic. This significantly complicates evading the entire sequence of packets scrutinized by the IDS, whereas theoretically, a single packet would be easier to evade. We also evaluated the transferability of adversarial techniques and the feasibility of such attacks with varying levels of knowledge about the target IDS and computational constraints. Albeit transfer attacks proved to be harder than in the image domain, the LSTM-based predictive model achieved a high evasion rate on CANdito [18], exceeding the white-box adversarial performance while exploiting a simpler oracle. Finally, we studied whether an attacker can craft adversarial samples offline and inject them later, satisfying the strict timing constraints of CAN: for two of the twelve CAN IDs under analysis, this technique was able to generate a large number of adversarial samples, highlighting a potential vulnerability.

Limitations. A limitation of our study, derived from the lack of availability of a real testing vehicle, is that it relies on previously collected traffic data. This may lead to a potential divergence from real-world situations. In practice, attacks in the traffic can influence the control system of the vehicle, thereby altering the actual traffic patterns, due to the feedback loop in cyber-physical systems. This aspect may lead to results that vary from those we have reported. Secondly, we lacked proprietary information about the semantics and functioning of the ECUs, making it difficult to evaluate the real-world consequences of perturbations and train baseline models effectively.

VII. Conclusions

In this paper, we addressed the potential impact of adversarial attacks on state-of-the-art automotive IDSs. We conducted a thorough evaluation of known adversarial evasion attacks, adapted to the automotive domain, against payload-based IDSs using real CAN traffic augmented with synthetic but realistic attacks. Our experiments involved designing and implementing customized variants of the popular BIM and perturbation algorithms to handle CAN packet signals. Evasion is achievable yet difficult across all tested scenarios, considering the specific constraints of the domain and the necessity to maintain the temporal integrity of the original packets even when they are altered. Interestingly, our findings also reveal that in the case of attacking a sophisticated model characterized by slow algorithmic convergence, an attack directly informed by full knowledge of the target (a white-box attack) might be less effective than a transfer attack, initially aimed at a less complex substitute model. In future work, we aim to explore more complex attacks from the image and malware domains. We also highlight the need for a standardized benchmarking procedure to support future research and ensure consistent and realistic evaluations of automotive intrusion detection systems.

Ackwnowledgment

This study was carried out within the MICS (Made in Italy – Circular and Sustainable) Extended Partnership and received funding from Next-Generation EU (Italian PNRR – M4 C2, Invest 1.3 – D.D. 1551.11-10-2022, PE00000004). CUP MICS D43C22003120001

References

[1] M. Bozdal, M. Samie, S. Aslam, and I. Jennions, "Evaluation of can bus security challenges," *Sensors*, vol. 20, no. 8, p. 2364, 2020.

[2] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 ieee symposium on security and privacy (sp)*. Ieee, 2017, pp. 39–57.

[3] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, "Adversarial attacks and defences: A survey," 2018.

[4] H. Chen and J. Tian, "Research on the controller area network," in *2009 International Conference on Networking and Digital Society*, vol. 2, 2009, pp. 251–254.

[5] J. Chen, M. I. Jordan, and M. J. Wainwright, "Hopskipjumpattack: A query-efficient decision-based attack," in *2020 ieee symposium on security and privacy (sp)*. IEEE, 2020, pp. 1277–1294.

[6] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, "Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *Proceedings of the 10th ACM workshop on artificial intelligence and security*, 2017, pp. 15–26.

[7] A. Chernikova and A. Oprea, "Fence: Feasible evasion attacks on neural networks in constrained environments," *ACM Transactions on Privacy and Security*, vol. 25, no. 4, pp. 1–34, 2022.

[8] A. de Faveri Tron, S. Longari, M. Carminati, M. Polino, and S. Zanero, "Canflict: Exploiting peripheral conflicts for data-link layer attacks on automotive networks," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, H. Yin, A. Stavrou, C. Cremers, and E. Shi, Eds. ACM, 2022, pp. 711–723. [Online]. Available: https://doi.org/10.1145/3548606.3560618

[9] D. E. Denning, "An intrusion-detection model," *IEEE Trans. Software Eng.*, vol. 13, no. 2, pp. 222–232, 1987. [Online]. Available: https://doi.org/10.1109/TSE.1987.232894

[10] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[11] S. C. HPL, "Introduction to the controller area network (can)," *Application Report SLOA101*, pp. 1–17, 2002.

[12] Z. Khan, M. Chowdhury, M. Islam, C.-Y. Huang, and M. Rahman, "Long short-term memory neural networks for false information attack detection in software-defined in-vehicle network," *arXiv preprint arXiv:1906.10203*, 2019.

[13] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Artificial intelligence safety and security*. Chapman and Hall/CRC, 2018, pp. 99–112, from 2016 preprint.

[14] B. Lampe and W. Meng, "A survey of deep learning-based intrusion detection in automotive applications," *Expert Systems with Applications*, p. 119771, 2023.

[15] Y. Li, J. Lin, and K. Xiong, "An adversarial attack defending system for securing in-vehicle networks," in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2021, pp. 1–6.

[16] S. Longari, F. Noseda, M. Carminati, and S. Zanero, "Evaluating the robustness of automotive intrusion detection systems against evasion attacks," in *Cyber Security, Cryptology, and Machine Learning - 7th International Symposium, CSCML 2023, Be'er Sheva, Israel, June 29-30, 2023, Proceedings*, ser. Lecture Notes in Computer Science, S. Dolev, E. Gudes, and P. Paillier, Eds., vol. 13914. Springer, 2023, pp. 337–352. [Online]. Available: https://doi.org/10.1007/978-3-031-34671-2_24

[17] S. Longari, M. Penco, M. Carminati, and S. Zanero, "Copycan: An error-handling protocol based intrusion detection system for controller area network," in *Proceedings of the ACM Workshop on Cyber-Physical Systems Security & Privacy, CPS-SPC@CCS 2019, London, UK, November 11, 2019*, L. Cavallaro, J. Kinder, and T. Holz, Eds. ACM, 2019, pp. 39–50. [Online]. Available: https://doi.org/10.1145/3338499.3357362

[18] S. Longari, C. A. Pozzoli, A. Nichelini, M. Carminati, and S. Zanero, "Candito: Improving payload-based detection of attacks on controller area networks," in *Cyber Security, Cryptology, and Machine Learning - 7th International Symposium, CSCML 2023, Be'er Sheva, Israel, June 29-30, 2023, Proceedings*, ser. Lecture Notes in Computer Science, S. Dolev, E. Gudes, and P. Paillier, Eds., vol. 13914. Springer, 2023, pp. 135–150. [Online]. Available: https://doi.org/10.1007/978-3-031-34671-2_10

[19] S. Longari, D. H. N. Valcarcel, M. Zago, M. Carminati, and S. Zanero, "Cannolo: An anomaly detection system based on LSTM autoencoders for controller area network," *IEEE Trans. Netw. Serv. Manag.*, vol. 18, no. 2, pp. 1913–1924, 2021. [Online]. Available: https://doi.org/10.1109/TNSM.2020.3038991

[20] M. Marchetti and D. Stabili, "Read: Reverse engineering of automotive data frames," *IEEE Transactions on Information Forensics and Security*, 09 2018.

[21] C. Miller and C. Valasek, "Adventures in automotive networks and control units," *DefCon 2013*, 2013.

[22] ——, "Can message injection," 2016, [Online, accessed 1-Oct-2022]. [Online]. Available: https://illmatics.com/can%20message%20injection.pdf

[23] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2574–2582.

[24] A. Nichelini, C. A. Pozzoli, S. Longari, M. Carminati, and S. Zanero, "Canova: A hybrid intrusion detection framework based on automatic signal classification for CAN," *Comput. Secur.*, vol. 128, p. 103166, 2023. [Online]. Available: https://doi.org/10.1016/j.cose.2023.103166

[25] T. Paladini, F. Monti, M. Polino, M. Carminati, and S. Zanero, "Fraud detection under siege: Practical poisoning attacks and defense strategies," *ACM Trans. Priv. Secur.*, vol. 26, no. 4, oct 2023. [Online]. Available: https://doi.org/10.1145/3613244

[26] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, 2017, pp. 506–519.

[27] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE symposium on security and privacy (SP)*. IEEE, 2016, pp. 582–597.

[28] S. Rajapaksha, H. Kalutarage, M. O. Al-Kadri, A. Petrovski, G. Madzudzo, and M. Cheah, "Ai-based intrusion detection systems for in-vehicle networks: A survey," *ACM Computing Surveys*, vol. 55, no. 11, pp. 1–40, 2023.

[29] K. Sauka, G.-Y. Shin, D.-W. Kim, and M.-M. Han, "Adversarial robust and explainable network intrusion detection systems based on deep learning," *Applied Sciences*, vol. 12, no. 13, p. 6451, 2022.

[30] R. S. Siva Kumar, M. Nyström, J. Lambert, A. Marshall, M. Goertzel, A. Comissoneru, M. Swann, and S. Xia, "Adversarial machine learning-industry perspectives," in *2020 IEEE Security and Privacy Workshops (SPW)*, 2020, pp. 69–75.

[31] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," 2013. [Online]. Available: https://arxiv.org/abs/1312.6199

[32] A. Taylor, S. Leblanc, and N. Japkowicz, "Anomaly detection in automobile control network data with long short-term memory networks," in *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 2016, pp. 130–139.

[33] E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," 2020.

[34] K. You, M. Long, J. Wang, and M. I. Jordan, "How does learning rate decay help modern neural networks?" *arXiv preprint arXiv:1908.01878*, 2019.

[35] M. Zago, S. Longari, A. Tricarico, M. Carminati, M. G. Pérez, G. M. Pérez, and S. Zanero, "Recan–dataset for reverse engineering of controller area networks," *Data in brief*, vol. 29, p. 105149, 2020.