Short: CARdea: Two-Phase Plausibility-Based Vehicle-to-Vehicle Anomaly Detection System

Mert D. Pesé[1], Arman Tabaddor[2], Teja Guruvelli[1], Roland Varriale[3], Marco De Vincenzi[4], Kang G. Shin[2]

[1]*Clemson University* [2]*The University of Michigan* [3]*Argonne National Laboratory* [4]*IIT-CNR*

[1]*{mpese, gguruve}@clemson.edu* [2]*{armantab, kgshin}@umich.edu*

[3]*rvarriale@anl.gov* [4]*marco.devincenzi@iit.cnr.it*

Abstract

Secure Vehicle-to-Vehicle (V2V) communication is threatened by internal attackers broadcasting anomalous Basic Safety Messages (BSMs). Existing anomaly detection methods often neglect deployment feasibility in resource-constrained vehicles, relying heavily on computationally expensive machine learning (ML) or multi-modal sensor fusion. To address this, we propose `CARdea`, a practical, real-time two-phase anomaly detection framework relying solely on standard BSM kinematics. Phase 1 performs a lightweight, in-vehicle statistical plausibility check, while Phase 2 offloads complex edge cases to a robust, opportunistically deployed ML classifier. Evaluated on 108 hours of simulated traffic, `CARdea` achieves an ultra-low 0.10 ms detection latency on edge hardware alongside an average 95.35% TPR and 0.55% FPR across seven attack types.

1 Introduction

Connected and automated vehicles (CAVs) rely on Vehicle-to-Vehicle (V2V) communication, exchanging Basic Safety Messages (BSMs) [18] to share kinematic state data such as position, speed, and acceleration. While authentication protocols effectively mitigate threats from external actors lacking valid credentials [14], a critical vulnerability remains: internal attackers. Compromised vehicles can broadcast legitimate but anomalous BSMs to launch false data injection, Sybil, or replay attacks [19], posing severe safety risks to the control systems of surrounding vehicles.

Despite recent advances in BSM anomaly detection, existing solutions largely neglect the deployment feasibility within resource-constrained Electronic Control Units (ECUs) [17]. Current methods propose either computationally heavy Machine Learning (ML) approaches or rely on expensive multi-modal sensor fusion (e.g., radar, Angle-of-Arrival) that is not yet economically viable for standard vehicles [10]. Crucially, prior work frequently ignores the ultra-low detection latency required to mitigate safety risks in real time.

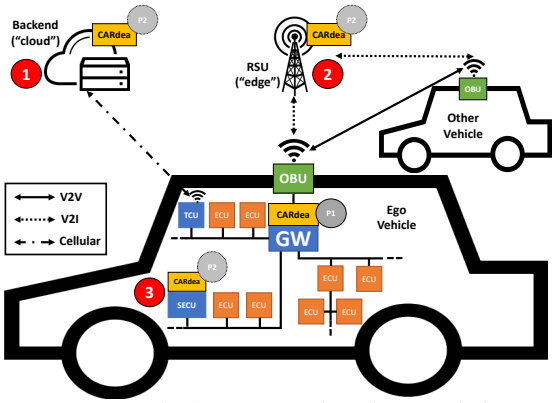


Figure 1: `CARdea` deployment options in V2X infrastructure

To address this disconnect, we propose `CARdea`[1], a practical, real-time anomaly detection scheme for V2V communication. `CARdea` bridges the gap between performance and practicality through a novel two-phase architecture. Phase 1 performs a lightweight, data-centric statistical plausibility check locally inside the vehicle, evaluating inherent correlations among BSM kinematic variables. Phase 2 acts as a robust ML classifier deployed opportunistically (e.g., edge or cloud) to resolve complex edge cases that Phase 1 cannot definitively classify. This paper makes the following main contributions:

- **Novel Two-Phase Architecture**: A hybrid system combining in-vehicle statistical checks with a remote ML classifier to optimize V2V safety and computational usability.
- **Ultra-Low Latency**: Achieves 0.10 ms detection latency on resource-constrained hardware (Raspberry Pi 3), meeting real-time in-vehicle requirements.
- **High Detection Efficacy**: Validated via 108 hours of simulation [25], achieving 95.35% TPR and 0.55% FPR across seven attack types using only standard BSM data. Code and datasets are publicly available [2].

[1]Cardea was the ancient Roman goddess of the hinge, Roman doors being hung on pivot hinges.

[2]https://github.com/tigerseclab/VehicleSec26_CARdea

Table 1: Comparison with related work. Notation: Position p, speed v, acceleration a, heading h, yaw rate ω.

Work	So19 [23]	Sha24 [20]	Yao17 [32]	Yav17 [33]	Bis12 [3]	Sun17 [28]	Van19 [29]	**CARdea**
BSM Data	p	p, v, a, h, ω	N/A	p, v, a	p, h, v	p, a	p, v, a	p, v, a
Other Data	RSSI	None	RSSI	Map	Radar	AoA, DS	None	None
Anomalies	Constant (+Offset) Random (+Offset) Eventual Stop	Constant (+Offset) Random (+Offset) High/Low Value Opposite Rotating Perpendicular	Sybil	Random Error	Sybil	Constant Offset	Instant Constant Gradual Drift Bias	Constant (+Offset) Random (+Offset) Sybil Replay
Approach	Physical Layer Consistency Check	Wasserstein GAN (WGAN)	Dynamic Time Warping (DTW)	Vehicle Physics (VP)	Particle Filter (PF)	Extended Kalman Filter (EKF)	CNN+KF	VP + (RF,SVM,DNN)
Performance (%)	TPR: 83.7	AUROC: 89 FPR: < 5	TPR: > 90 FPR: < 10	TPR: 90 FPR: 2.7	N/A	TPR: 100 FPR: 5	TPR: 51.5-99.2	Phase 1 TPR: 98 FPR: 0.02 Phase 2 TPR: 99 FPR: 0.16
Detection Time (ms)	N/A	40	630	N/A	5-25	3.2	KF: 5.1	Phase 1: 0.09-0.11 Phase 2: 5.1-15.5

2 Threat Model

Based on adversary incentives [26], V2X attacks fall into two taxonomic dimensions [10]: (i) *Active* vs. *Passive*: active attackers interact with the system; passive attackers eavesdrop. (ii) *Internal* vs. *External*: internal attackers hold valid credentials; external ones do not. Most V2X attacks assume an active, internal adversary, including DoS, Sybil, replay, and false data injection [19]. DoS exhausts resources, Sybil creates ghost vehicles via fake identities, replay resends stale BSMs, and false data injection broadcasts spoofed sensor data. We focus on falsification of kinematic state data (Sybil, replay, false data injection), as these are validatable against physical laws. High-level semantic attacks (e.g., bogus traffic warnings) are excluded, as they require context-aware verification beyond CARdea's kinematic plausibility checks.

False data may stem from faulty sensors or deliberate spoofing via a (i) compromised OBU or (ii) compromised IVN. Miller *et al.* [13] demonstrated that the CAN bus is vulnerable to both physical and remote tampering. Anomalies should be caught before BSM broadcast or before data reaches actuator ECUs. While CAN-based IDS detects IVN compromise [31], it misses OBU-level tampering; thus, false data is best detected upon BSM receipt before IVN forwarding. We assume a compromised vehicle broadcasting contiguous spoofed BSMs to our ego vehicle, with the attacker controlling all BSM field values. In contrast, faulty sensors typically cause issues in only a few frames rather than sustained segments. Note that CARdea is agnostic to the European equivalent of BSM, the *Cooperative Awareness Message* (CAM) [9].

We propose CARdea as a data-centric, plausibility-based V2V anomaly detection module running *inside* the central gateway. Detected BSM anomalies trigger data discarding; anomalies persisting beyond a defined threshold result in full suspension of communication with the offending vehicle.

Generalization to Multi-Sensor Spoofing. One drawback of CARdea's current system design is that it only comprises *single*-sensor spoofing in AT1–4, where one consistent sensor is anomalous per BSM. Another case is *multi*-sensor spoofing, where more than one sensor may be anomalous. If two sensors are compromised, Phase 1 detects inconsistency via inter-sensor correlation. If all three are spoofed consistently, Phase 1 may fail, an inherent limitation of physics-based detection also noted by So *et al.* [23]. Adding RSSI as a parallel plausibility check mitigates this: even when a smart attacker maintains kinematic consistency across all three sensors, RSSI can still flag position spoofing. A BSM is marked anomalous if *either* module triggers. Multi-sensor spoofing experiments are left for future work.

3 Related Work

Existing Approaches. As summarized in Table 1, previous V2X anomaly detection largely relies on either statistical plausibility checks or Machine Learning (ML). Statistical methods leverage vehicle dynamics [33], particle or Kalman filters [3, 28], and RSSI [23, 32]. While computationally lighter, pure data-centric statistical checks frequently fail against sophisticated GPS spoofing unless augmented by physical-layer signals [23] or expensive multi-modal sensor fusion, such as radar and Angle-of-Arrival [28]. Conversely, ML-based solutions employing Convolutional Neural Networks [29] or Generative Adversarial Networks (VEHIGAN) [20] achieve highly robust detection against complex attacks. However, these models introduce computational and memory overhead in the Electronic Control Units (ECUs).

Comparison to Previous Work. While recent advancements in BSM anomaly detection have demonstrated promising detection rates, they frequently overlook the practical and economic constraints of actual in-vehicle deployment. CARdea distinguishes itself from existing literature through three novel contributions tailored for real-world automotive feasibility:

Ultra-Low Latency on Resource-Constrained Hardware. Existing ML-based approaches incur high latency. CARdea

achieves 0.10 ms detection latency on ECU-like hardware, at least 46× faster than Kalman Filter (KF)-based detectors. Moreover, most prior work does not report latency [7, 30, 33].

Cost-Effective Data Requirements. Prior work relies on costly multi-modal sensing (e.g., AoA, Doppler Shift) [1, 22]. CARdea instead uses only existing kinematic data (position, speed, acceleration) from standard BSMs, eliminating the need for additional sensors and reducing deployment cost.

Strategic Two-Phase Hybrid Architecture. Prior work trades off latency (statistical models) and accuracy (ML). CARdea bridges this via a two-phase design: a lightweight in-vehicle statistical check (Phase 1) and an opportunistic remote ML classifier (Phase 2), preserving real-time safety while maintaining high detection performance.

4 System Design

4.1 Overview

Considering the limitations of related work, we propose CARdea to provide real-time, accurate, and light-weight detection of BSM anomalies. It uses a hybrid of statistical (Phase 1, Sec. 4.3) and ML-based (Phase 2, Sec. 4.4) anomaly detection. Phase 1 runs locally *inside* the vehicle to provide very low detection latency that only statistical approaches can provide. Another benefit of CARdea is the low memory consumption which is important due to automotive resource constraints [17]. In contrast, Phase 2 can run *outside* the vehicle, e.g., on the car-maker's backend or the roadside edge or even cloud, and can rely on more computing resources.

Phase 1 performs real-time detection of both *highly-likely anomalous frames* and *highly-likely non-anomalous frames*. We define a *frame* f as a sequence of contiguous sensor readings/samples from a transmitter vehicle. A frame is classified as *highly-likely anomalous* if the mean of the predicted conditions for each (anomalous or non-anomalous) sample in the frame is above a certain per-frame threshold $\theta_{f,a}$. Similarly, a frame is said to be *highly-likely non-anomalous* if the mean of the predicted conditions for each sample in the frame is smaller than $\theta_{f,na}$. Frame predictions between these two thresholds will be sent to Phase 2 for fine-grained classification. Phase 1 will temporarily consider such frames *potentially anomalous* if the frame prediction is greater than 0.5, and *potentially non-anomalous* otherwise, until a classification result from Phase 2 becomes available. Thus, the goal in Phase 1 is to maximize the safety-critical True Positive Rate (TPR) as well as to minimize the False Positive Rate (FPR). However, Phase 1 is optimized for frames that consist of mainly anomalous samples (i.e., from an attacker vehicle) or mainly non-anomalous samples (i.e., from a benign vehicle). Frames consisting of a mixture of anomalous and non-anomalous samples (i.e., a 50/50 split) are sent to Phase 2 for *per-sample detection*. Our ego vehicle can temporarily suspend its communication with vehicles marked as anomalous as a result of anomalies in consecutive BSMs.

Phase 2 provides computationally powerful per-sample detection strictly for frames where Phase 1 is uncertain, preserving V2V usability without sacrificing safety. Both phases exploit inter-sensor correlations: Phase 1 via vehicle dynamics, Phase 2 via ML, relying exclusively on standard BSM data and RSSI to eliminate expensive sensor fusion. This hybrid architecture of CARdea satisfies OEM constraints by allowing flexible Phase 2 deployment (cloud, edge, or super-ECU) to balance cost, bandwidth, and performance.

4.2 Anomalies under Consideration

It is challenging to evaluate anomaly detection schemes for CAVs due to the lack of anomaly datasets for BSMs. Thus, we evaluate CARdea using the Vehicular in Network Simulation (Veins) framework [25]. By using a simulation instead of generating anomalies on top of a static dataset, we ensure that false data may propagate to the control system over time and affect the behavior of the ego vehicle and neighboring vehicles. A simulation can thus account for vehicle dynamics.

We evaluate CARdea on 7 attack types. AT1–AT4 falsify position (lat/lon), speed, and acceleration, extended from [11,12] and consistent with literature [2,21]. AT5 (Sybil), AT6 (Data Replay), and AT7 (Stealthy) target real-world scenarios; AT5–AT6 are extended from [11], while AT7 is motivated by stealthy attacks on robotic vehicles [5, 6, 16]. AT7 specifically maximizes spoofing damage while evading detection, stress-testing CARdea against a sophisticated adversary consistent with the threat model in Sec. 2. The superscript i denotes the sensor; x_t^i is the true measurement at time t: **(AT1) Constant:** broadcasts a fixed $x_t^i = x_c$; **(AT2) Constant Offset:** adds a fixed offset, $x_t^i = x_t^i + \Delta x_c$; **(AT3) Random:** broadcasts $x_t^i = \mathcal{U}([x_{min}, x_{max}])$ bounded by the simulation space; **(AT4) Random Offset:** adds random noise, $x_t^i = x_t^i + \mathcal{U}([-x_c, x_c])$; **(AT5) Sybil:** generates ghost vehicles via $x_t^i = S_x \cdot (x_l^i + x_r^i)$ for map grid length S_x, local measurement x_l^i, and random measurement x_r^i; **(AT6) Data Replay:** resends delayed target BSM data $x_t^i \in_R [x_0^i, x_{t-1}^i]$; and **(AT7) Stealthy:** injects the maximum relative false data without crossing the optimal per-sample threshold θ_s^i via $x_t^i = x_t^i + (1 - \theta_s^i \cdot b_i)$ for $b_i > 0$.

4.3 Phase 1: Local Anomaly Detection

Overview. Phase 1 performs real-time detection of anomalies inside the vehicle by leveraging statistical models of inherent sensor correlations. Among the sensor data included in the BSM [18], we select three fundamental sensors to express vehicle dynamics: (GPS) position $\vec{p}$, velocity $\vec{v}$, and longitudinal acceleration a. BSMs are transmitted by vehicles every 100 ms, indicated by dt. Our ego vehicle will receive BSMs from M surrounding vehicles which vary with traffic and driving behavior of the other cars. The objective of Phase 1 is to perform the plausibility verification

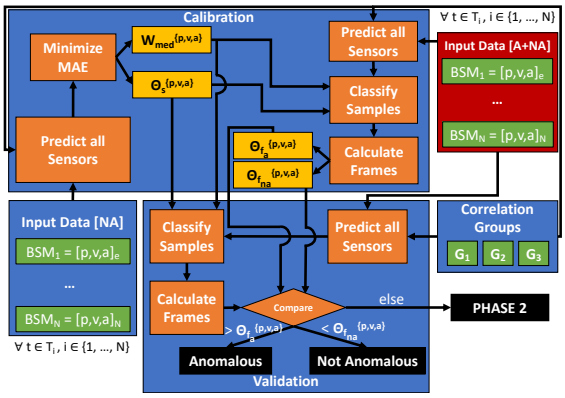


Figure 2: Phase 1 (A: Anomalous, NA: Non-Anomalous)

of each BSM upon its reception by the OBU and to determine if the analyzed BSM is anomalous. Let t be the absolute timestamp; the correlated sensor groups follow basic physics equations: $\vec{p}_{t+dt} = f(p_t, \vec{v}_t) = \vec{p}_t + \int_t^{t+dt} \vec{v}_t dt$, $v_t = f(\vec{p}_t, \vec{p}_{t+dt}) = \frac{|\vec{p}_{t+dt} - \vec{p}_t|}{dt}$, and $a_t = f(v_t, v_{t+dt}) = \frac{v_{t+dt} - v_t}{dt}$.

As shown in Fig. 2, Phase 1 is decomposed into threshold calibration and validation modules. The *correlation groups* G1–G3 characterize the correlation between the three sensors as expressed above, respectively. Namely, G1 describes the relationship between position $\vec{p}$ and velocity vector $\vec{v}$; G2 the relationship between speed v and position $\vec{p}$; and G3 the relationship between longitudinal acceleration a and speed v. Speed v is defined as the magnitude of two-dimensional velocity $\vec{v} = (v_x, v_y)^T$. Below we elaborate on the threshold calibration and validation of Phase 1.

Calibration. The calibration module is configured to operate offline, presumably at the time of vehicle manufacturing. The objective of calibration is to compute four thresholds for each sensor $i \in \{p, v, a\}$: the median window size W_{med}^i, per-sample threshold θ_s^i, per-frame anomaly threshold $\theta_{f,a}^i$, and per-frame non-anomaly threshold $\theta_{f,na}^i$. A single anomalous sensor reading in a BSM can be detected by using the aforementioned correlation groups. For example, if only the speed v is anomalous, G2 can be used to calculate the correct speed from the GPS readings. If v is not anomalous, G2 should yield a value similar to the speed in the received BSM. Since GPS readings can be noisy/erroneous (like any other sensor reading), the difference between *calculated* and *received* values will have a small error. The mean of all errors across all non-anomalous datasets is characterized by the per-sample threshold θ_s^i for each correlation group. This threshold is determined by computing the *Mean Absolute Error* (MAE) for each sample t between the received signal x_t and estimated signal $\hat{x}_t$. Ideally, estimated and received signals should be near-identical for non-anomalous data. However, estimation is prone to noise, which a rolling mean filter of length W mitigates by smoothing the raw signal. The optimal window W_{med}^i for each sensor i minimizes MAE across all non-anomalous trips; the averaged minimized MAEs yield θ_s^i. Using only past samples up to the current frame, the filter preserves real-time detection.

These two calibrated parameters W_{med}^i and θ_s^i may be sufficient to detect the anomaly in a BSM. However, we introduce further statistical techniques and design choices to reduce the false alarm rate in Phase 1. Similar to a cumulative sum, we compute an average $\theta_{f,a}^i$ for all anomalous frames, for each sensor group. A frame prediction is the mean of predicted conditions (1 for anomalous or 0 for non-anomalous) based on θ_s^i for samples in a frame f. This same calculation is computed for non-anomalous frames, $\theta_{f,na}^i$. The per-frame threshold computation can be generalized in Eq. (1), where N is the number of samples and $|f|$ the frame size:

$$\theta_{f,*}^i = \frac{1}{N} \sum_{j=1}^{N} \frac{1}{|f|} \sum_{k=j\cdot|f|}^{j\cdot|f|+|f|} x_k^i. \quad (1)$$

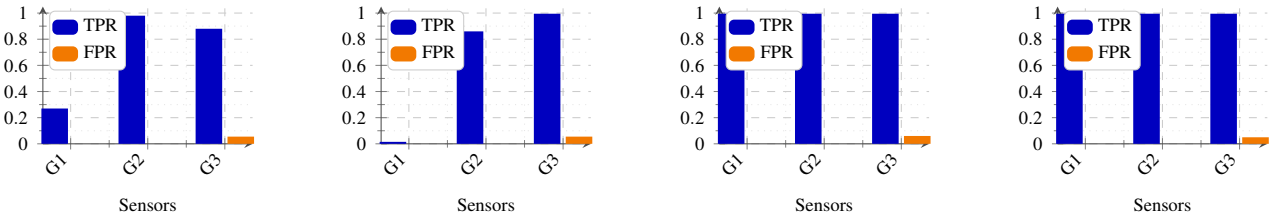
The introduction of the above thresholds helps ensure truly anomalous or truly non-anomalous frames are detected immediately by Phase 1. Therefore, Phase 1 considers frames, and the computed frame value is compared to the respective $\theta_{f,*}^i$ for anomalous and non-anomalous vehicles to make a decision. Trustworthiness of calibration is ensured through continuous backend validation. OEMs can aggregate non-anomalous data from the fleet to periodically refine W_{med} and θ values, pushing optimized thresholds to vehicles via secure over-the-air (OTA) updates to account for aging sensors or changing road conditions.

Validation. The anomaly detection of Phase 1 operates online and extracts the relevant signal data from each BSM. Using G1-G3 and the four calibration parameters W_{med}^i, θ_s^i, $\theta_{f,a}^i$, and $\theta_{f,na}^i$, it first calculates the MAE for the three correlation groups and compares it with the threshold θ_s^i computed during training. If the computed MAE is larger than the threshold, the predicted condition for the respective sample is marked as *anomalous*. Likewise, if the computed MAE is smaller than the threshold, the predicted condition for the sample is marked as *non-anomalous*. After collecting W_{med}^i samples, the samples are smoothed by a rolling mean filter of window size W_{med}^i, and the MAE is re-calculated and compared again with the threshold to update the appropriate markings.

We then compute the mean of all predicted conditions for samples in a frame, such that if this computed frame prediction is greater than, or equal to $\theta_{f,a}^i$, we mark all samples within this frame as *highly-likely anomalous*, and if this computed frame prediction is less than, or equal to $\theta_{f,na}^i$, this frame is marked as *highly-likely non-anomalous*. If the frame prediction lies between these two thresholds, the affected frames are sent to Phase 2 for further investigation.

4.4 Phase 2: Remote Anomaly Detection

Overview. Phase 2 employs ML-based anomaly detection designed for opportunistic, remote deployment (e.g., edge



(a) AT1 (b) AT2 (c) AT3 (d) AT4

Figure 3: Phase 1 per-frame performance with frame size $f = 10$

or cloud, see Fig. 1), operating without strict real-time constraints. Its primary goal is to resolve uncertain frames forwarded from Phase 1, specifically those falling within the interval $[\theta^i_{f,na}, \theta^i_{f,a}]$. For each sensor group, an associated ML model executes in parallel. Similar to Phase 1, a final logical-OR is applied, meaning an anomaly is flagged if any single model detects it.

Feature Extraction. Motivated by [24], the features extracted are dependent on a sensor of interest. More precisely, for each correlation group, we extract the rolling mean $\overline{X^i_t}$, rolling mean displacement, estimated displacement and plausibility check for each vehicle communicating with the ego vehicle.

Training and Validation. We evaluate three models for Phase 2: Random Forest (RF) uses 5-fold cross-validation with 100 estimators; Support Vector Machine (SVM) standardizes features to zero mean and unit variance, utilizing an RBF kernel ($C = 10$) to balance performance with training time; and a Deep Neural Network (DNN) relies on a 60/20/20 train/val/test split, trained for 25–50 epochs (batch size 32) with early stopping. The DNN architecture comprises 6 hidden layers (32–128 nodes, ReLU activations), a 0.2 dropout rate, and a sigmoid output, optimized via Adam using binary cross-entropy loss.

5 Evaluation

5.1 Experimental Setup

We evaluate CARdea using the Veins simulation framework [26] with the Luxembourg SUMO Traffic (LuST) scenario [4] over a 6.63 km^2 topology. Our dataset comprises 108 hours of simulations encompassing over 49,000 vehicles. Attacker vehicles were spawned with a probability $\alpha = 0.05$ and broadcast contiguous sequences of anomalous messages (30–70 messages) with probability $\beta = 0.01$ [27]. These adversaries executed the seven defined attack types (AT1–AT7) manipulating position, speed, and longitudinal acceleration. To demonstrate deployment feasibility, Phase 1 was evaluated in Python 3 on a resource-constrained Raspberry Pi 3 Model B (1.2 GHz, 1 GB RAM) representative of an in-vehicle ECU. Phase 2 was evaluated on a desktop server (64-core Intel Xeon, 128 GB RAM) representing an edge or cloud backend.

5.2 Phase 1

Performance. We first calculated the ROC (see Fig. 4a in Appendix) using our training data for each correlation group to determine a satisfactory θ^i_s. To select an optimal θ^i_s, we selected the first threshold with at least 90% TPR, in accordance with the ROC curve. Then we used θ^i_s to determine $\theta^i_{f,*}$ values for each correlation group. We calculated the means over all different configurations of the anomalous datasets. The calibrated thresholds for each correlation group are as follows: for G_1, $\theta^1_{f,a} = 0.95$ and $\theta^1_{f,na} = 0.015$; for G_2, $\theta^2_{f,a} = 0.83$ and $\theta^2_{f,na} = 0.021$; for G_3, $\theta^3_{f,a} = 0.90$ and $\theta^3_{f,na} = 0.04$.

Furthermore, when computing the $\theta^i_{f,a}$ and $\theta^i_{f,na}$ values, we evaluated them on frames of size $3, 5, 7$, and 10. As frame size increases, performance generally improves, as shown in Figs. 4b and 4c (in Appendix). G3 FPR, for example, decreases from $\approx 9\%$ for a frame size of 3 to $\approx 5.5\%$ for a frame size of 7, and then plateaus. We selected a frame size of 10 for performance, latency, and resource metrics.

Phase 1 targets high TPR for anomalous frames and low FPR for non-anomalous ones. As Fig. 3 shows, confident Phase 1 classifications can trigger mitigation without engaging Phase 2. However, G1 AT1 and AT2 are a known limitation: because G1 uses its previous (potentially spoofed) value to compute estimates, MAEs spike at anomalous/non-anomalous transitions and fail to stay consistently below the determined threshold θ^i_s. These position spoofing attacks are better addressed via RSSI-based plausibility checks [23], which can boost TPR by up to 40% when combined with Phase 1 via logical OR. For AT5 and AT6, Phase 1 achieves 99% TPR and 0.005% FPR. Stealthy attacks, however, are explicitly designed to evade Phase 1, making `CARdea` vulnerable to them, if undetected in Phase 1, all frames are forwarded to Phase 2, negating the hybrid system's efficiency benefits.

Detection Latency. Phase 1 mean latency was evaluated on a Raspberry Pi 3 Model B (comparable to in-vehicle ECUs). As Table 6 (in Appendix) shows, processing one frame takes 0.10 ms across all attack types and correlation groups, negligible given BSMs arrive every 100 ms. In an AV platooning scenario, Phase 1 latency is $\approx$2000x smaller than the delay required to restore full autonomy after compromise, which ranges from 247 ms (2 vehicles) to 1457 ms (8 vehicles) [8].

Memory Consumption. Table 6 (in Appendix) shows an

Table 2: Phase 2 per-sample performance for frame size $f = 10$. AT5 and AT6 are independent of correlation group G* and are displayed for G1 merely for visualization purposes. Best values per attack type are bolded.

	G1			G2			G3		
Attack Type	TPR	FPR	F-1	TPR	FPR	F-1	TPR	FPR	F-1
AT1	93.46%	3.24%	90.30%	**96.90%**	0.57%	**94.77%**	91.36%	**0.02%**	91.44%
AT2	**76.06%**	**0.53%**	**77.81%**	68.62%	1.30%	69.35%	68.30%	0.83%	69.60%
AT3	94.53%	9.06%	87.45%	97.14%	**1.00%**	**94.03%**	75.04%	13.02%	75.06%
AT4	**91.16%**	1.21%	**89.50%**	88.53%	1.26%	87.04%	52.16%	0.89%	52.02%
AT5	**92.75%**	**0.23%**	**93.80%**	-	-	-	-	-	-
AT6	**87.43%**	2.89%	**86.16%**	-	-	-	-	-	-
AT7, $b = 0.1$	62.93%	1.08%	64.57%	**86.37%**	1.71%	**84.29%**	70.29%	**0.50%**	71.15%
AT7, $b = 0.01$	60.01%	**0.32%**	61.35%	**78.99%**	1.87%	**78.20%**	64.68%	0.46%	65.49%
AT7, $b = 0.001$	61.12%	**0.25%**	62.02%	**83.56%**	0.83%	**83.74%**	67.70%	0.40%	68.41%

Table 3: Phase 2 results on frames from Phase 1

	G1				G2				G3				**Average**
	AT1	**AT2**	**AT3**	**AT4**	**AT1**	**AT2**	**AT3**	**AT4**	**AT1**	**AT2**	**AT3**	**AT4**	
TPR	100%	99.75%	99.67%	99.51%	100%	85.8%	99.06%	76.5%	98.64%	98.37%	95.9%	91.0%	95.35%
FPR	0.0%	0.015%	0.0%	0.0%	0.0%	1.3%	0.16%	1.9%	0.09%	0.3%	1.06%	1.8%	0.55%

average RAM usage of 39 MB across the four attack types. While this may seem high for in-vehicle ECUs with limited RAM [17], the current Python implementation inflates usage. A C-based version would require far less memory, given the minimal parameters and equations involved.

5.3 Phase 2

We evaluate TPR, FPR, and F-1 score of Phase 2, both independently and jointly with Phase 1. We first evaluate Phase 2 in a scenario where it receives *all* frames from Phase 1.

Performance. Compared to Phase 1's *per-frame* performance evaluation, Phase 2 focuses on *per-sample* classification, which explains certain lower TPR values, despite its generally favorable performance. For data spoofing attacks (AT1—AT4), RFs almost exclusively display the best performance, particularly for G1 and G2, frequently achieving high TPR and F-1 scores (e.g., $>$ 90% F-1 for AT1, AT3, AT4). Compared to Van *et al.* [29] who inject anomalies on a static dataset, instead of conducting a simulation that makes their evaluation less realistic, they report performance comparable to `CARdea`. Additionally, Sun *et al.* [28] implemented the constant offset attacker, yet relied on AoA and Doppler shift to achieve comparable results. One limitation of Phase 1 was the need to incorporate RSSI to detect such attacks, which might also be required for Phase 2, as AT2 performance suggests. However, AT1 performance is significantly better than in Phase 1, even without considering RSSI. For Sybil (AT5) and Data Replay (AT6) attacks, RFs outperform SVMs and DNNs with $\approx$ 90%+ average F-1 score. For stealthy attacks (AT7), there was no observed correlation in detection performance and the attacker's b_i selection, with RFs being the superior ML model again. The performance numbers for AT7 were only slightly lower than the other attack types, demonstrating that `CARdea` succeeded in thwarting this advanced attack type. Table 2 summarizes the performance of the best performing model (Random Forest) while Table 4 depicts the full performance of all three models for all attack types.

Computation Time. Table 6 (in Appendix) reports Phase 2 inference times for RF, SVM, and DNN across frame sizes 3–10 on both the RPi and desktop. RPi latency substantially exceeds desktop: e.g., RF at f=10 takes 101.9 ms on the RPi vs. 16.0 ms on the desktop, exceeding the BSM period. Compared to Phase 1's 0.10 ms, Phase 2's ML overhead is not ideal for direct ECU deployment.

Memory Consumption. Table 6 (in Appendix) shows that ML models consume significantly more RAM than statistical methods, about 5× more in Phase 1, making them less suitable for ECUs. RFs and SVMs exceed 100 MB, and deep learning models require up to 3× more.

Interactions between Phase 1 and 2. We now perform a combined evaluation of Phases 1 and 2 jointly to evaluate Phase 2 on the frames sent from Phase 1. We can express this real-time interaction between the two phases where the input X are Phase 1's estimated sensor readings of frame size N. Phase 2 implements an RF model, and we select a frame size of 10, consistent with our preceding evaluation. Phase 2 reports an average 95.35% TPR and 0.55% FPR (see Table 3).

6 Conclusion

`CARdea` demonstrates that combining a lightweight, in-vehicle statistical check with a powerful ML-based approach offers a highly performant, practical solution for real-time V2V anomaly detection. By bridging the gap between high detection accuracy and deployment feasibility on resource-constrained ECUs, our two-phase architecture proves that security and latency need not be a compromise. Future work will explore adaptive thresholding, dynamically shrinking the uncertain interval $[\theta_{f,na}, \theta_{f,a}]$ as Phase 1 learns from Phase 2. This will progressively reduce offloading rates and further minimize cellular bandwidth over the vehicle's lifecycle.

Acknowledgments

This work was supported in part by NSA H98230-21-1-0317 through INSuRE+C, as well as the US National Science Foundation under grants 2245223 and 2608885.

References

[1] Ahmed Abdo, Sakib Md Bin Malek, Zhiyun Qian, Qi Zhu, Matthew Barth, and Nael Abu-Ghazaleh. Application level attacks on connected vehicle protocols. In *22nd International Symposium on Research in Attacks, Intrusions and Defenses ({RAID} 2019)*, pages 459–471, 2019.

[2] Monowar H Bhuyan, Dhruba Kumar Bhattacharyya, and Jugal K Kalita. Network anomaly detection: methods, systems and tools. *Ieee communications surveys & tutorials*, 16(1):303–336, 2013.

[3] Norbert Bißmeyer, Sebastian Mauthofer, Kpatcha M Bayarou, and Frank Kargl. Assessment of node trustworthiness in vanets using data plausibility checks with particle filters. In *2012 IEEE Vehicular Networking Conference (VNC)*, pages 78–85. IEEE, 2012.

[4] Lara Codeca, Raphaël Frank, and Thomas Engel. Luxembourg sumo traffic (lust) scenario: 24 hours of mobility for vehicular networking research. In *2015 IEEE Vehicular Networking Conference (VNC)*, pages 1–8. IEEE, 2015.

[5] Pritam Dash, Mehdi Karimibiuki, and Karthik Pattabiraman. Out of control: stealthy attacks against robotic vehicles protected by control-based techniques. In *Proceedings of the 35th Annual Computer Security Applications Conference*, pages 660–672, 2019.

[6] Pritam Dash, Guanpeng Li, Zitao Chen, Mehdi Karimibiuki, and Karthik Pattabiraman. Pid-piper: Recovering robotic vehicles from physical attacks. In *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 26–38. IEEE, 2021.

[7] Stefan Dietzel, Rens van der Heijden, Hendrik Decke, and Frank Kargl. A flexible, subjective logic-based framework for misbehavior detection in v2v networks. In *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, pages 1–6. IEEE, 2014.

[8] Jeremy Erickson, Shibo Chen, Melisa Savich, Shengtuo Hu, and Z Morley Mao. Commpact: Evaluating the feasibility of autonomous vehicle contracts. In *2018 IEEE Vehicular Networking Conference (VNC)*, pages 1–8. IEEE, 2018.

[9] TCITS ETSI. Intelligent transport systems (its); vehicular communications; basic set of applications; part 2: Specification of cooperative awareness basic service. *Draft ETSI TS*, 20(2011):448–51, 2011.

[10] M. Hasan, S. Mohan, T. Shimizu, and H. Lu. Securing vehicle-to-everything (v2x) communication platforms. *IEEE Transactions on Intelligent Vehicles*, 5(4):693–713, 2020.

[11] Joseph Kamel, Mohammad Raashid Ansari, Jonathan Petit, Arnaud Kaiser, Ines Ben Jemaa, and Pascal Urien. Simulation framework for misbehavior detection in vehicular networks. *IEEE transactions on vehicular technology*, 69(6):6631–6643, 2020.

[12] Joseph Kamel, Michael Wolf, Rens Wouter van der Heijden, Arnaud Kaiser, Pascal Urien, and Frank Kargl. VeReMi Extension: A Dataset for Comparable Evaluation of Misbehavior Detection in VANETs. In *IEEE International Conference on Communications (ICC)*, Dublin (virtual), Ireland, June 2020. Virtual conference.

[13] Charlie Miller and Chris Valasek. Adventures in automotive networks and control units. *Def Con*, 21(260-264):15–31, 2013.

[14] Mujahid Muhammad and Ghazanfar Ali Safdar. Survey on existing authentication issues for cellular-assisted v2x communication. *Vehicular Communications*, 12:50–65, 2018.

[15] Orange Travel. How much data does spotify use?, March 28 2025. Accessed: 2026-02-24.

[16] Raul Quinonez, Jairo Giraldo, Luis Salazar, Erick Bauman, Alvaro Cardenas, and Zhiqiang Lin. {SAVIOR}: Securing autonomous vehicles with robust physical invariants. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 895–912, 2020.

[17] ResearchAndMarkets.com. Automotive microcontroller unit (mcu) industry report 2023: Automotive high-end mcus will still be in short supply, and how oems can break the situation, March 2023. Accessed: 2026-01-28.

[18] SAE. *Dedicated Short Range Communications (DSRC) Message Set Dictionary*, March 2016.

[19] Fatih Sakiz and Sevil Sen. A survey of attacks and detection mechanisms on intelligent transportation systems: Vanets and iov. *Ad Hoc Networks*, 61:33–50, 2017.

[20] Md Hasan Shahriar, Mohammad Raashid Ansari, Jean-Philippe Monteuuis, Cong Chen, Jonathan Petit, Y Thomas Hou, and Wenjing Lou. Vehigan: Generative adversarial networks for adversarially robust v2x

misbehavior detection systems. In *2024 IEEE 44th International Conference on Distributed Computing Systems (ICDCS)*, pages 1294–1305. IEEE, 2024.

[21] Abhishek B Sharma, Leana Golubchik, and Ramesh Govindan. Sensor faults: Detection methods and prevalence in real-world datasets. *ACM Transactions on Sensor Networks (TOSN)*, 6(3):1–39, 2010.

[22] Junjie Shen, Jun Yeon Won, Zeyuan Chen, and Qi Alfred Chen. Drift with devil: Security of multi-sensor fusion based localization in high-level autonomous driving under {GPS} spoofing. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 931–948, 2020.

[23] Steven So, Jonathan Petit, and David Starobinski. Physical layer plausibility checks for misbehavior detection in v2x networks. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, pages 84–93, 2019.

[24] Steven So, Prinkle Sharma, and Jonathan Petit. Integrating plausibility checks and machine learning for misbehavior detection in vanet. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 564–571. IEEE, 2018.

[25] Christoph Sommer, David Eckhoff, Alexander Brummer, Dominik S Buse, Florian Hagenauer, Stefan Joerer, and Michele Segata. Veins: The open source vehicular network simulation framework. In *Recent Advances in Network Simulation*, pages 215–252. Springer, 2019.

[26] Jan Peter Stotz, Norbert Bißmeyer, Frank Kargl, Stefan Dietzel, Panos Papadimitratos, and Christian Schleiffer. Preserve d1. 1 security requirements of vehicle security architecture. *Deliverable, PRESERVE consortium*, 2011.

[27] Beshr Sultan and Mike McDonald. Assessing the safety benefit of automatic collision avoidance systems (during emergency braking situations). In *Proceedings of the 18th International Technical Conference on the Enhanced Safety of Vehicle.(DOT HS 809 543)*, 2003.

[28] Mingshun Sun, Ming Li, and Ryan Gerdes. A data trust framework for vanets enabling false data detection and secure vehicle tracking. In *2017 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9. IEEE, 2017.

[29] Franco van Wyk, Yiyang Wang, Anahita Khojandi, and Neda Masoud. Real-time sensor anomaly detection and identification in automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 21(3):1264–1276, 2019.

[30] Xiaoyang Wang, Ioannis Mavromatis, Andrea Tassi, Raul Santos-Rodriguez, and Robert J Piechocki. Location anomalies detection for connected and autonomous vehicles. In *2019 IEEE 2nd Connected and Automated Vehicles Symposium (CAVS)*, pages 1–5. IEEE, 2019.

[31] Wufei Wu, Renfa Li, Guoqi Xie, Jiyao An, Yang Bai, Jia Zhou, and Keqin Li. A survey of intrusion detection for in-vehicle networks. *IEEE Transactions on Intelligent Transportation Systems*, 21(3):919–933, 2019.

[32] Yuan Yao, Bin Xiao, Gaofei Wu, Xue Liu, Zhiwen Yu, Kailong Zhang, and Xingshe Zhou. Multi-channel based sybil attack detection in vehicular ad hoc networks using rssi. *IEEE Transactions on Mobile Computing*, 18(2):362–375, 2018.

[33] Chaitanya Yavvari, Zoran Duric, and Duminda Wijesekera. Vehicular dynamics based plausibility checking. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8. IEEE, 2017.

A Appendix

A.1 Bandwidth

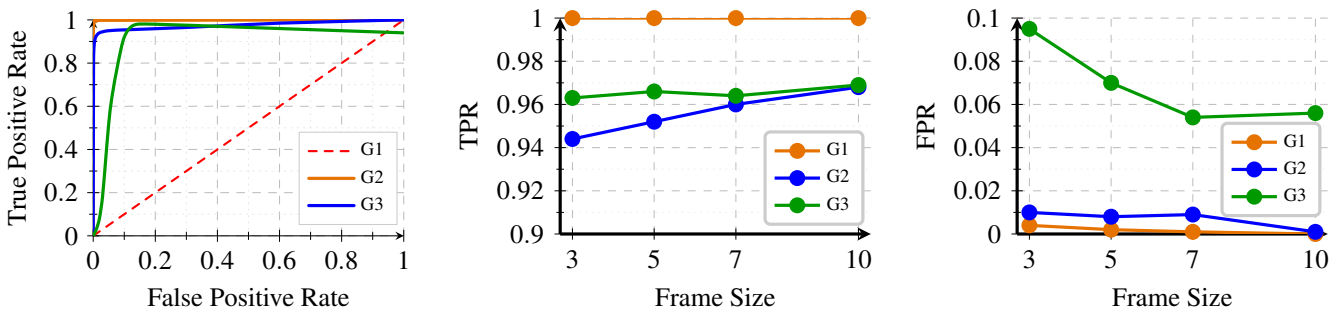
We would like to discuss how many of the *potentially anomalous*-flagged frames from Phase 1 have to be sent to Phase 2. This will have an impact on bandwidth which is another important metric for an OEM since it drives up cost. The car will very likely use the built-in cellular connection. Each BSM has a default size of 254 bytes and will be sent using the Transmission Control Protocol (TCP). This is necessary since we cannot assume that the network connection during a trip will always be reliable. Together with the TCP/IP header overhead, each packet including the BSM will have a total size of 318 bytes. Table 5 (in Appendix) summarizes the mean percentage of frames in each experiment to be sent to Phase 2 (a mean number of frames to be sent to Phase 2 d_{p2} over total number of frames d_{total} for given vehicle), as well as the required bandwidth b.

$$b = \frac{d_{p2} \cdot 318bytes}{d_{total} \cdot 0.1s} \tag{2}$$

The required bandwidth is always < 0.56 kB/s if only every flagged frame is transmitted. At most, 1.8 MB is transmitted per hour, less than one minute of high-quality Spotify streaming (320 kbps) [15]. Table 5 uses $f = 10$; smaller frame sizes reduce bandwidth by sending fewer frames to Phase 2. Thus, the table shows an upper bound. Larger frames improve performance but increase bandwidth, making frame size a trade-off between performance and cost.

Table 4: Phase 2 per-sample performance for frame size $f = 10$. AT5 and AT6 are independent of correlation group G* and are displayed for G1 merely for visualization purposes. Best values per attack type are bolded.

Model	Attack Type	G1			G2			G3		
		TPR	FPR	F-1	TPR	FPR	F-1	TPR	FPR	F-1
RF	AT1	93.46%	3.24%	90.30%	**96.90%**	0.57%	**94.77%**	91.36%	**0.02%**	91.44%
	AT2	**76.06%**	**0.53%**	**77.81%**	68.62%	1.30%	69.35%	68.30%	0.83%	69.60%
	AT3	94.53%	9.06%	87.45%	97.14%	**1.00%**	**94.03%**	75.04%	13.02%	75.06%
	AT4	**91.16%**	1.21%	**89.50%**	88.53%	1.26%	87.04%	52.16%	0.89%	52.02%
	AT5	**92.75%**	**0.23%**	**93.80%**	-	-	-	-	-	-
	AT6	**87.43%**	2.89%	**86.16%**	-	-	-	-	-	-
	AT7, $b = 0.1$	62.93%	1.08%	64.57%	**86.37%**	1.71%	**84.29%**	70.29%	**0.50%**	71.15%
	AT7, $b = 0.01$	60.01%	**0.32%**	61.35%	**78.99%**	1.87%	**78.20%**	64.68%	0.46%	65.49%
	AT7, $b = 0.001$	61.12%	**0.25%**	62.02%	**83.56%**	0.83%	**83.74%**	67.70%	0.40%	68.41%
SVM	AT1	72.56%	2.01%	71.79%	77.91%	1.39%	77.14%	62.66%	0.98%	62.45%
	AT2	68.65%	9.76%	61.23%	56.12%	3.61%	56.76%	64.25%	1.02%	64.41%
	AT3	**97.79%**	1.79%	91.85%	95.61%	1.03%	90.86%	69.44%	14.56%	69.98%
	AT4	88.70%	1.20%	86.59%	84.55%	1.52%	82.85%	50.12%	**0.0009%**	49.37%
	AT5	76.40%	3.99%	76.72%	-	-	-	-	-	-
	AT6	77.38%	**0.36%**	82.03%	-	-	-	-	-	-
	AT7, $b = 0.1$	51.54%	0.94%	50.56%	66.27%	11.22%	56.75%	54.03%	1.15%	53.27%
	AT7, $b = 0.01$	50.41%	0.72%	50.24%	64.83%	7.73%	61.88%	52.60%	1.09%	51.15%
	AT7, $b = 0.001$	50.50%	0.49%	50.08%	68.18%	8.23%	62.83%	55.19%	0.84%	54.12%
DNN	AT1	82.90%	2.85%	78.94%	85.31%	1.25%	83.71%	62.32%	2.23%	61.54%
	AT2	69.52%	5.75%	65.08%	55.01%	3.32%	55.14%	59.88%	0.56%	60.71%
	AT3	94.65%	1.58%	90.85%	95.37%	**1.00%**	90.72%	73.38%	2.74%	70.35%
	AT4	83.20%	3.80%	76.37%	86.09%	2.72%	80.06%	50.00%	**0.0009%**	49.05%
	AT5	76.68%	6.14%	76.39%	-	-	-	-	-	-
	AT6	79.91%	**0.36%**	84.56%	-	-	-	-	-	-
	AT7, $b = 0.1$	51.26%	0.51%	50.46%	66.07%	20.82%	58.38%	54.81%	1.67%	55.45%
	AT7, $b = 0.01$	50.21%	0.47%	49.71%	66.68%	8.65%	62.57%	55.02%	1.98%	53.93%
	AT7, $b = 0.001$	50.20%	**0.25%**	49.94%	66.68%	8.65%	62.57%	54.05%	0.48%	54.67%



(a) Combined ROC Curves for θ_s^i (b) TPR vs Frame Size (c) FPR vs Frame Size

Figure 4: Phase 1 ROC curves and per-frame performance based on frame size.

Table 5: Bandwidth considerations

Sensor Group G_i	**Attack Type** AT^*	**Mean Percentage (%)**	**Bandwidth (kB/s)**
G_1	AT_1	5.3	0.17
	AT_2	5.2	0.17
	AT_3	5.2	0.17
	AT_4	5.3	0.17
	AT_5	8.7	0.28
	AT_6	7.6	0.24
G_2	AT_1	3.9	0.12
	AT_2	4.2	0.13
	AT_3	4.5	0.14
	AT_4	4.7	0.15
	AT_5	3.0	0.09
	AT_6	2.7	0.08
G_3	AT_1	14.5	0.46
	AT_2	14.0	0.45
	AT_3	15.0	0.48
	AT_4	15.6	0.50
	AT_5	17.9	0.56
	AT_6	17.8	0.56

Table 6: Detection Latency and RAM Usage

Phase	**Model**	**Frame Size**	**Latency (ms)**		**RAM Usage (MB)**	
			RPi	Desktop	RPi	Desktop
Phase 1	N/A	1	0.10	N/A	38.8	N/A
Phase 2	RF	3	98.9	15.1	106	153
		5	95.1	15.2	106	153
		7	99.4	15.6	106	153
		10	101.9	16.0	106	153
	SVM	3	30.1	3.8	102	147
		5	31.9	4.2	102	147
		7	33.5	4.6	102	147
		10	38.8	4.9	102	147
	DNN	3	570	41.5	357	485
		5	532	39.5	357	485
		7	528	39.9	357	485
		10	538	40.4	357	485