Adversarial Attacks on Plausibility Checks in V2X Security*

Jungwoo Park, Kunho Park, and Sanghoon Jeon†

Kookmin University, Seoul, Republic of Korea
{caesar6679, darklie13, sh.jeon}@kookmin.ac.kr

Abstract

Vehicle-to-Everything (V2X) communication is a core technology that enhances road safety and efficiency through the periodic transmission of Basic Safety Messages (BSMs). To ensure message reliability, the system adopts a multi-layered security architecture composed of digital signature–based authentication (ECDSA or PQC), L1/L2 Plausibility Checks, and a machine learning–based Misbehavior Detection System (ML-MDS). However, this architecture remains vulnerable to subtle adversarial perturbations within permissible physical limits. This study proposes a novel conditional adversarial perturbation model that can bypass both cryptographic verification and rule-based plausibility checks in existing V2X systems while evading ML-MDS detection. Experiments using the VeReMi dataset demonstrate that an insider attacker possessing a legitimate certificate can conditionally manipulate speed, acceleration, and yaw rate, causing all transmitted messages to pass the L1/L2 Plausibility Checks while significantly degrading the detection performance of the ML-MDS. In particular, perturbations activated when the Time-to-Collision (TTC) falls below a predefined threshold distort the reported TTC to appear longer than the actual value, thereby concealing collision risks. These findings indicate that "cryptographic integrity combined with rule-based verification" alone cannot ensure the semantic integrity of V2X communications, underscoring the necessity of enhancing the adversarial robustness of ML-MDS and developing MLSecOps-based defense strategies.

Adversarial Attacks; Vehicle-to-Everything (V2X); Basic Safety Message (BSM); L1/L2 Plausibility Checks; Machine Learning-based Misbehavior Detection Systems (ML-MDS); Adversarial Perturbations; Time-to-Collision (TTC)

1 Introduction

With the proliferation of Intelligent Transportation Systems (ITS) and autonomous driving technologies, Vehicle-to-Everything (V2X) communication has emerged as a core infrastructure ensuring road safety and traffic efficiency [1]. V2X periodically exchanges Basic Safety Messages (BSMs) to share dynamic vehicle information such as position, velocity, acceleration, heading, and yaw rate in real time [2]. This information forms the foundation of various safety and efficiency services—including collision avoidance, cooperative driving, and traffic optimization—and thus ensuring message integrity and trustworthiness is critical to overall traffic safety [3].
To achieve this, the V2X receiver pipeline is typically designed as a multi-layered architecture comprising (i) cryptographic verification based on IEEE 1609.2 [4], (ii) L1/L2 Plausibility Checks [5], and (iii) a machine learning–based Misbehavior Detection System (MDS) [6]. While this structure effectively mitigates tampered messages and overt spoofing attacks, each layer has inherent limitations. Cryptographic verification ensures the authenticity and integrity of

*Proceedings of the 9th International Conference on Mobile Internet Security (MobiSec'25), Article No. 76, December 16-18, 2025, Sapporo , Japan. © The copyright of this paper remains with the author(s).
†Corresponding author

messages but cannot validate the semantic truth of their contents [7]. Similarly, Plausibility Checks only verify value ranges (L1) or simple kinematic consistency (L2: $a_{\text{lat}} = v \cdot r$), making it difficult to detect subtle perturbations ($\pm\boldsymbol{\varepsilon}$) introduced within physical tolerance limits. Furthermore, the ML-MDS, introduced to complement these checks, remains vulnerable to adversarial attacks due to its sensitivity to small input perturbations [8].
Existing studies have primarily focused on explicit attacks such as GPS spoofing, location falsification, and message dropping [9]. However, current systems are designed under the assumption that messages passing "cryptographic verification + rule-based checks" can be inherently trusted, leading to a lack of empirical analysis on whether perturbations within plausibility bounds can also bypass ML-MDS. Therefore, the critical research gap lies in systematically validating the feasibility and safety implications of micro-scale perturbations that can evade both rule-based and learning-based detection.
This study investigates the question: "Can small perturbations within tolerance bounds ($\|\delta\|_\infty \le \varepsilon$) simultaneously evade L1/L2 Plausibility Checks and ML-based Misbehavior Detection Systems (MDS), while systematically distorting the Time-to-Collision (TTC)?"
To explore this, we design a *conditional (triggered)* ε-perturbation. The attack is not applied indiscriminately but is activated only when the TTC falls below a threshold T_0 (e.g., 3 seconds) [10]. This setup models a realistic and threatening scenario that specifically targets imminent collision situations while remaining undetectable under normal driving conditions.
The attack design is based on three principles. First, instead of inserting large perturbations into single frames, we distribute them across multiple frames (cross-frame perturbation) to conceal abrupt changes [11]. Second, we control jerk variance to emulate realistic sensor noise levels. Third, by maintaining the constraint $a_{\text{lat}} = v \cdot r$, the attack reliably passes the L1/L2 checks [12]. These techniques make the adversarial signals indistinguishable from natural sensor noise, effectively deceiving detection systems.
Consequently, while individual messages appear legitimate, the reported TTC becomes systematically biased to longer values compared to the actual ones. This not only enables evasion from detection but also conceals imminent collision risks, leading to severe safety degradation.
The main contributions of this paper are as follows.

1. **Empirical demonstration of ε-perturbation degrading ML-MDS detection performance (Recall):** By introducing small perturbations within the tolerance range ($\pm\boldsymbol{\varepsilon}$), messages can pass the L1/L2 Plausibility Checks. Experimental results show that when such perturbed messages are used, the detection performance (Recall) of ML-based detectors (Random Forest, MLP) significantly declines.

2. **Design of noise-like perturbation mechanisms through cross-frame insertion and jerk variance control:** Perturbations are distributed over multiple frames, and jerk variations are smoothed to resemble natural sensor noise. Quantitative analysis confirms that such perturbations effectively distort the Time-to-Collision (TTC) toward longer values.

3. **Quantitative evaluation of TTC-aware attacks concealing collision risks:** Using the VeReMi dataset [13, 14], we construct Random Forest and MLP-based MDS models to empirically validate that the perturbations not only reduce Recall but also bias TTC estimates upward. This result highlights the need for future studies on SOTA attack and defense mechanisms for ML-based V2X security.

2 Background

This section provides the essential background knowledge required to understand the proposed study. Specifically, it (i) introduces the structure of the Basic Safety Message (BSM) defined in SAE J2735 and the IEEE 1609.2–based Secured Protocol Data Unit (SPDU), (ii) explains the definitions, representative rules, and roles of the L1/L2 Plausibility Checks, (iii) presents the definition of the Time-to-Collision (TTC) used for collision risk assessment along with the trigger conditions defined in this study, and (iv) clearly specifies the scope and assumptions of the research.

2.1 Overview of BSM, SPDU, and IEEE 1609.2

The **Basic Safety Message (BSM)** is the core V2X message defined in the SAE J2735 standard. It periodically broadcasts key dynamic vehicle information—such as position, velocity, acceleration, heading, and yaw rate—at a frequency of approximately 10 Hz. These messages serve as the foundational data for situational awareness and collision avoidance applications. As shown in **Table 1**, the BSM structure is divided into two parts: Part I, which contains the mandatory core data elements, and Part II, which includes optional extended data elements.

Table 1: BSM Part I & Part II Fields (SAE J2735)

Part	Field	Unit
BSM Part I	Message ID (msgID)	-
	Message Count (msgCount)	-
	Temporary ID (id)	-
	Second Mark (secMark)	ms (0–65535)
	Latitude (lat)	$1/10^7$ degree
	Longitude (long)	$1/10^7$ degree
	Elevation (elev)	0.1 m
	Positional Accuracy (accuracy)	m
	Transmission State (transmission)	enum
	Speed (speed)	0.02 m/s
	Heading (heading)	0.0125°
	Steering Wheel Angle (angle)	1.5°
	Acceleration Set (accelSet)	m/s^2, deg/s
	Brake System Status (brakes)	on/off (bit field)
	Vehicle Size (size)	cm
BSM Part II	Path History	lat/long/elev/time
	Path Prediction	offset (m), confidence (%)
	Exterior Lights	bit field (on/off)
	Vehicle Safety Extensions	status flags
	Vehicle Status	various (fuel level %, wiper on/off)
	Event Flags	bit field (e.g., crash, emergency stop)
	Tire Pressure Monitoring	kPa
	Stability/Control Data	on/off, enum

2.2 Secured Protocol Data Unit (SPDU)

The IEEE 1609.2 standard specifies that application messages must be transmitted in the form of a secure encapsulation known as the **Secured Protocol Data Unit (SPDU)**. As illustrated in **Figure 1**, the overall structure of an SPDU consists of the protocol version, security header, certificate, payload, and digital signature. Among these, the three core components are (i) **Payload**, (ii) **Certificate**, and (iii) **Digital Signature**.

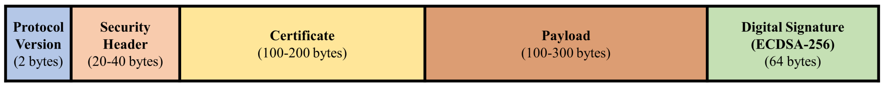
Protocol Version (2 bytes)
Security Header (20-40 bytes)
Certificate (100-200 bytes)
Payload (100-300 bytes)
Digital Signature (ECDSA-256) (64 bytes)

Figure 1: SPDU Structure

- **Payload**: Contains the Basic Safety Message (BSM), which includes essential dynamic data such as the sender vehicle's position, speed, and acceleration. This information provides the core kinematic parameters necessary for collision avoidance and traffic safety functions.
- **Certificate**: Represents the sender vehicle's public key certificate, typically implemented as a pseudonym certificate. This certificate is issued by a Public Key Infrastructure (PKI), allowing the receiver to verify that the message sender is a legitimate entity. Periodic certificate renewal also supports privacy protection by preventing long-term linkability of messages.
- **Digital Signature**: The sender vehicle signs the payload and header using its private key before transmission. The receiver verifies the message using the public key contained in the certificate, ensuring both message integrity and sender authenticity. Currently, the Elliptic Curve Digital Signature Algorithm (ECDSA) is predominantly used, while recent research actively explores Post-Quantum Cryptography (PQC)–based digital signatures to ensure resilience against quantum attacks.

In summary, the SPDU combines *BSM data (what to transmit)* with the *authentication and signature structure (how to transmit securely)* to ensure both the integrity and authenticity of V2X messages.

2.3 L1/L2 Plausibility Checks

The Plausibility Check is a rule-based procedure designed to quickly verify whether the received kinematic values are physically feasible. It is generally categorized into two types: L1 (range check) and L2 (consistency check).

L1 rules validate the physical limits of individual variables. For example, the vehicle's speed v_t, acceleration a_t, and jerk ($\text{jerk}_t = \frac{a_t - a_{t-1}}{\Delta t}$) must not exceed their respective thresholds $v_{\max}, a_{\max}, j_{\max}$. Similarly, parameters such as position, altitude, and vehicle dimensions must remain within reasonable ranges.

L2 rules verify dynamic consistency among related variables. A representative rule is the lateral acceleration–velocity–yaw rate relation, which must satisfy $|a_{\text{lat},t} - v_t \cdot r_t| \leq \kappa$, based on the curvature motion approximation.

The Plausibility Check operates as a preliminary filter between IEEE 1609.2 cryptographic verification and the ML-based Misbehavior Detection System (MDS). While it offers the advantages of low computational cost and interpretability, it cannot detect subtle perturbations or insider attacks within permissible bounds. Therefore, a multi-layered defense architecture combining L1/L2 Plausibility Checks with ML-MDS is required to ensure robust V2X security.

2.4 Definition of TTC and Trigger

We define the leader vehicle as the vehicle in the same lane with the minimum longitudinal distance d to the ego vehicle, and the relative speed as $v_{\text{rel}} = v_{\text{ego}} - v_{\text{lead}}$. The Time-to-Collision (TTC) is computed as

$$\text{TTC} = \begin{cases} \min\bigl(\text{TTC}_{\max},\ \max(\varepsilon_t,\ d/v_{\text{rel}})\bigr), & v_{\text{rel}} > 0, \\ \text{TTC}_{\max}, & \text{otherwise}, \end{cases}$$

where the lower bound ε_t prevents numerical singularity for very small denominators and $\text{TTC}_{\max}$ caps excessively large values. In this study we adopt the default parameters $\varepsilon_t = 10^{-3}$ s, $\text{TTC}_{\max} = 10$ s, and set the attack trigger threshold to $T_0 = 3.0$ s.

2.5 Scope & Assumptions

Scope

- All received frames are required to pass the L1/L2 Plausibility Checks (rule-evasion constraint).
- The attack is conditional: we consider a *TTC-aware attack* that activates only when the Time-to-Collision (TTC) falls below the threshold T_0.

This configuration is chosen because, in realistic vehicular communication environments, an attacker that injects perturbations indiscriminately across all situations would be readily detectable. By contrast, restricting perturbations to imminent-collision situations (TTC $< T_0$) allows the attacker to bypass L1/L2 Plausibility Checks while exerting a more severe and targeted impact on detectors and safety, thereby representing a realistic and threatening attack scenario.

Assumptions This study is based on the following reasonable assumptions:

- **Cryptographic verification assumption:** All messages are assumed to successfully pass IEEE 1609.2–based cryptographic verification (digital signature and certificate validation). In other words, message integrity and authenticity are guaranteed, while the semantic truth of message contents remains a separate evaluation target.
- **Attacker model:** The attacker is assumed to be an insider vehicle possessing legitimate certificates. Consequently, signatures still verify correctly after message manipulation.
- **Detector composition:** The Misbehavior Detection System (MDS) is assumed to be implemented using representative ML models such as Random Forest, Multi-Layer Perceptron (MLP), and Long Short-Term Memory (LSTM). In particular, LSTM models are employed to exploit the temporal patterns inherent in V2X message sequences for anomaly detection. In this paper we use these models at a proof-of-concept level; future work may extend the evaluation to architectures such as Transformers, CNN–LSTM hybrids, and Temporal Convolutional Networks (TCN), as well as ensemble techniques.

- **Communication environment:** Vehicles are assumed to transmit and receive BSMs at approximately 10 Hz during driving, and attack injections are synchronized to this standard transmission rate.
- **Physical-rule tolerance:** Fine-grained perturbations within the thresholds defined by L1/L2 Plausibility Checks are assumed to be difficult to distinguish from realistic sensor noise. This assumption is accepted as reasonable for real-world vehicular environments.

These assumptions are not intended as excessive simplifications but rather reflect practical constraints commonly adopted in V2X security research. Under these premises, this study evaluates the impact of potential *semantic attacks* that can arise from legitimately signed messages and examines the possibility that existing L1/L2 Plausibility Checks and ML-based MDS may fail to detect such attacks.

3 Related Work

This section reviews prior studies conducted in the context of Vehicle-to-Everything (V2X) security. Specifically, it examines related work on cryptographic verification and rule-based checks, machine learning–based misbehavior detection systems, and adversarial attack research within intelligent transportation systems, before highlighting the distinct contributions of this study.

V2X messages undergo integrity and authentication verification using digital signatures in accordance with IEEE 1609.2. This ensures the origin and tamper resistance of the message but does not guarantee the **semantic truth** of the data. To address this limitation, Leinmüller et al. [5] proposed the **L1/L2 Plausibility Checks**. L1 verifies the physical range of values such as speed, acceleration, and jerk, while L2 validates the consistency between lateral acceleration a_{lat} and $v \cdot r$. In this paper, IEEE 1609.2 and L1/L2 checks are collectively referred to as the **Dual Defense** framework. This rule-based approach offers low computational cost and real-time applicability; however, minor **micro-perturbations (ε perturbations)** within acceptable physical limits can pass L1/L2 checks, thereby increasing the overall attack success rate. Recent studies have demonstrated the feasibility of *physics-constrained attacks* that disrupt communication or perception while maintaining kinematic consistency, emphasizing that temporally smooth perturbations—undetectable by rule-based checks—pose a greater threat to real systems [15].

To overcome these limitations, recent studies have developed **Machine Learning–based Misbehavior Detection Systems (MDS)**. Various models such as Random Forests, Neural Networks, and Autoencoders have been explored, with Boualouache et al. [6] providing a comprehensive survey. These methods exhibit superior detection accuracy compared to rule-based approaches; however, systematic analysis of their **adversarial robustness** remains scarce. Because ML-based models are sensitive to input perturbations, carefully crafted adversarial Basic Safety Messages (BSMs) may easily bypass detection, leading to a significantly higher **Attack Success Rate (ASR)**. This study empirically investigates such a scenario by proposing a conditional ε-perturbation that simultaneously bypasses rule-based filters and evades ML-MDS.

Adversarial attacks have been extensively studied in computer vision and natural language processing, where even small perturbations can significantly mislead classifiers [8]. However, adversarial research in V2X security remains relatively limited. Existing work has mainly focused on **explicit attacks** such as GPS spoofing, message falsification, or position manipulation, often evaluating detection evasion or alert delay performance. In contrast, **implicit adversarial attacks**—which inject perturbations within plausibility limits to evade rule-based checks while simultaneously compromising ML-MDS—have received little attention. This reveals a critical security gap in current V2X architectures, which inherently trust messages that have passed "cryptographic integrity + rule-based verification." The present study addresses this gap by proposing a conditional ε-perturbation attack that passes the Dual Defense system while systematically biasing the reported Time-to-Collision (TTC) toward longer values, effectively concealing imminent collision risks.

In summary, (i) previous research has addressed the design of L1/L2 Plausibility Checks and ML-MDS but has not verified the potential for **conditional adversarial perturbations to bypass the Dual Defense system** and (ii) the **safety impact (concealment effect)** of adversarial perturbations on **Time-to-Collision (TTC)** estimation in V2X contexts has not been adequately explored. To fill this gap, this study proposes a ε-perturbation–based **conditional TTC-aware attack algorithm** and, using the VeReMi dataset [13], empirically demonstrates (1) simultaneous evasion of rule-based and ML-MDS defenses and (2) a quantifiable upward bias in the reported TTC.

4 Main Method

4.1 Cryptographic Verification (IEEE 1609.2)

The receiver pipeline is summarized in this section and the plausibility specifications, and then presents the **Attack Design**. The receiver pipeline is composed of three sequential stages: (i) cryptographic verification (IEEE 1609.2), (ii) L1/L2 Plausibility Checks, and (iii) ML-based Misbehavior Detection System (ML-MDS).

4.1.1 Cryptographic Verification (IEEE 1609.2)

This study assumes that all received messages successfully pass IEEE 1609.2–based digital signature and certificate verification. In other words, the receiving vehicle validates signatures and certificates to ensure message integrity and sender authenticity; however, the semantic truthfulness of the message contents is not assessed at this stage. Consequently, this paper evaluates the detectability of manipulated messages by L1/L2 Plausibility Checks and ML-MDS under the condition that cryptographic verification has already succeeded.

Algorithm 1 Cryptographic Verification Pseudocode

```
Require: message (payload bytes), cert, signature
Ensure: True / False
1: function VerifySignature(message, cert, signature)
2:     pk ← ExtractPublicKey(cert)
3:     if Verify(pk, message.payload, signature) then
4:         return True
5:     else
6:         return False
7:     end if
8: end function
```

Algorithm 1 provides a simplified representation of the IEEE 1609.2 verification procedure. Upon reception, the vehicle extracts the public key from the sender's certificate and uses that key to verify the message signature. If the signature verification succeeds, the message is considered trustworthy and is accepted for further processing; otherwise, the message is discarded. For the purposes of this study, we assume that all messages successfully pass this cryptographic verification step.

4.2 L1/L2 Plausibility Checks

Following the definitions provided in Section 2 and Section 3, the L1/L2 Plausibility Checks are implemented to compute Boolean indicators, `pass_L1` and `pass_L2`, for each received frame. The threshold parameters employed in this study are as follows:

Experimental Thresholds

$$\tau_v = 40 \text{ m/s}, \quad \tau_a = 7 \text{ m/s}^2, \quad \tau_j = 15 \text{ m/s}^3, \quad \kappa = 5 \text{ m/s}^2.$$

Here, τ_v, τ_a, and τ_j represent the L1 thresholds for velocity, acceleration, and jerk, respectively, while κ denotes the tolerance for L2 dynamic consistency. These thresholds are selected to reflect the physical constraints of real vehicles: $\tau_v = 40\,\text{m/s}$ corresponds to the highway speed limit plus a safety margin; $\tau_a = 7\,\text{m/s}^2$ reflects the typical acceleration and braking capability of passenger vehicles; $\tau_j = 15\,\text{m/s}^3$ represents a realistic upper limit on jerk; and $\kappa = 5\,\text{m/s}^2$ defines the allowable tolerance for dynamic consistency, considering sensor noise and measurement uncertainty.

Algorithm 2 L1 / L2 Plausibility Check Pseudocode

```
Require: sequence of frames with timestamps, sampling interval $dt$ (e.g., 0.1s for 10Hz), thresholds $\tau_v, \tau_a, \tau_j, \kappa$, small velocity threshold $\epsilon_v$
Ensure: per-frame booleans pass_L1, pass_L2
 1: function CheckL1(frame_t, frame_t-1, dt)
 2:     $v_t \leftarrow$ frame_t.v;   $a_t \leftarrow (v_t -$ frame_t-1.v$)/dt$;   $jerk_t \leftarrow (a_t -$ frame_t-1.a$)/dt$
 3:     return $|v_t| \le \tau_v \wedge |a_t| \le \tau_a \wedge |jerk_t| \le \tau_j$
 4: end function
 5: function CheckL2(frame_t)
 6:     $alat, v_t, r_t \leftarrow$ frame_t.alat, frame_t.v, frame_t.r
 7:     return $|alat - v_t \cdot r_t| \le \kappa$
 8: end function
 9: function PlausibilityPipeline(frames, dt)
10:     for each frame $t$ in frames do
11:         pass_L1 ← CheckL1(frame_t, frame_t-1, dt)
12:         pass_L2 ← pass_L1 ? CheckL2(frame_t) : False
13:         Emit or log (pass_L1, pass_L2, failed_rule)
14:     end for
15: end function
```

Algorithm 2 illustrates the frame-level procedure of the plausibility checking process. First, `CheckL1()` computes acceleration a_t from the velocity difference between the current and previous frames, and subsequently derives jerk$_t$ using the previous frame's acceleration. The L1 check passes only if all three conditions hold: $|v_t| \le \tau_v$, $|a_t| \le \tau_a$, and $|\text{jerk}_t| \le \tau_j$. Only frames that satisfy the L1 conditions proceed to `CheckL2()`, which verifies the physical consistency among lateral acceleration, yaw rate, and velocity according to $|a_{\text{lat}} - v \cdot r| \le \kappa$. When the vehicle's velocity is extremely low ($|v_t| < \epsilon_v$), numerical instability is handled separately. For every frame, both `pass_L1` and `pass_L2` results—as well as any violated rule—are logged for post-analysis. The thresholds $\tau_v, \tau_a, \tau_j, \kappa$, and ϵ_v are initialized using the baseline values defined in the text.

4.3 ML-Based Misbehavior Detection System (MDS)

This section describes the structure, training, and evaluation of the Machine Learning–based Misbehavior Detection System (MDS), which operates on messages that have already passed the L1/L2 Plausibility Checks. The objective is to quantitatively evaluate how effectively the proposed ε-perturbation attack can evade detection even after satisfying all L1/L2 physical constraints.

Random Forest. Random Forest is an ensemble classifier that combines multiple decision trees using bootstrap sampling and random feature selection to reduce inter-tree correlation. The final output is determined by majority voting across all trees, providing stable performance and interpretable feature importance. In this study, it serves as a baseline model using static features.

Multi-Layer Perceptron (MLP). The MLP is a feedforward neural network with two hidden layers and ReLU activations. Dropout regularization prevents overfitting, and the sigmoid output produces binary anomaly probabilities. Unlike Random Forest, the MLP captures nonlinear feature interactions and can detect distributed perturbations within L1/L2 thresholds.

Long Short-Term Memory (LSTM). The LSTM, a variant of recurrent neural networks (RNNs), is designed to learn long-term dependencies in sequential data. While standard RNNs suffer from vanishing gradient issues, the LSTM incorporates cell states and gate mechanisms (input, forget, and output gates) to selectively retain and update important information.

In this study, the LSTM learns the temporal patterns of speed, acceleration, and yaw rate across consecutive frames to identify **temporal anomalies** that are difficult to detect with single-frame analysis. Under normal driving conditions, velocity and acceleration vary smoothly within physical limits, whereas during an attack, small but consistent perturbations accumulate across the sequence. By capturing such temporal inconsistencies, the LSTM achieves higher detection performance than Random Forest or MLP models.

4.4 Attack Design

The attacker injects a per-frame perturbation δ_t into the original payload $x_t = (v_t, a_t, r_t)$ to produce the modified payload $\tilde{x}_t = x_t + \delta_t$. This section provides a formal definition of the attack objective, the design stages, and the execution procedure.

4.4.1 Threat objective (formal definition)

The goal is to construct a sequence of perturbations $\{\delta_t\}$ that simultaneously satisfies the following three requirements:

1. Preserve L1/L2 plausibility for every perturbed frame:

$$\forall t:\ \mathrm{L1}(x_t + \delta_t) = \mathrm{true}, \quad \mathrm{L2}(x_t + \delta_t) = \mathrm{true}.$$

2. Evade (or induce misclassification in) the ML-MDS:

$$\mathrm{MDS}\big(x_{t-W+1} + \Delta, \ldots, x_t + \delta_t\big) = \mathrm{normal},$$

where W denotes the window length referenced by the ML-MDS and Δ denotes accumulated perturbations applied to past frames.

3. Systematically increase the reported TTC (risk concealment):

$$\text{TTC-aware} \geq \text{TTC} + \Delta_{\text{TTC}},$$

i.e., the TTC computed after perturbation (TTC-aware) should be larger than the original TTC by at least Δ_{TTC}.

4.4.2 Attack components (two-stage design)

The attack is designed in two stages.

(A) Micro-perturbation generation to pass L1/L2 Let $g_i(\cdot)$ denote the function representing the i-th plausibility rule and θ_i the maximum permissible deviation for that rule. Each perturbation δ_t must satisfy the constraints

$$\forall i, \quad |g_i(x_t + \delta_t)| \leq \theta_i,$$

which enforces that the perturbed sample remains within each rule's allowable tolerance. Additionally, each frame's perturbation magnitude is bounded by $|\delta_t| \leq \varepsilon$ (e.g., $\varepsilon = 0.1$ m/s). This construction ensures L1/L2 compliance at the frame level.

(B) ML-MDS evasion — frame-splitting (accumulated perturbation) Because an abrupt large perturbation in a single frame is more likely to be detected by L1/L2 checks or by ML-MDS, the attacker distributes the same total perturbation Δv across K consecutive frames:

$$\text{Total perturbation } \Delta v \text{ distributed over } K \text{ frames: } \Delta v = \sum_{k=0}^{K-1} \delta^{(k)}, \quad |\delta^{(k)}| \leq \varepsilon_k.$$

For example, distributing $\Delta v = +0.2$ m/s over $K = 4$ yields increments of $+0.05$ m/s per frame. Splitting patterns may be uniform, alternating (e.g., $+\varepsilon, -\varepsilon, \ldots$), or otherwise engineered to minimize detectability.

Constraints and safety mechanisms During perturbation injection, the following constraints are continuously checked; violations trigger rollback or reduction of the perturbation (using a rollback factor $\gamma \in (0, 1)$):

- Jerk limit: enforce $|\text{jerk}| \leq \text{jerkCap}$.
- Re-verify L1/L2 Plausibility: call CheckL1 and CheckL2 for the perturbed frame(s).
- Numerical stability around near-zero speed: handle $|v| < \epsilon_v$ by clamping yaw-rate computations or using fallback procedures.

These safeguards ensure the attack remains stealthy (i.e., stays within the Dual Defense tolerances) while achieving the cumulative effect required to bias TTC and evade ML detectors.

Algorithm 3 Conditional ε-Attack

```
Require: frames, parameters: $\varepsilon, K$, jerkCap, $\kappa, T_0, dt$, rollback $\gamma \in (0,1)$, small vel $\epsilon_v$
Ensure: modified frames with applied perturbations
1: for each time $t$ do
2:     $x_t \leftarrow$ frames[t]; compute TTC$_t$
3:     if TTC$_t < T_0$ and CheckL1($x_t$) and CheckL2($x_t$) then
4:         choose target bias $\delta_v$; $v' \leftarrow v_t + \delta_v$; $a' \leftarrow (v' - v_{t-1})/dt$; jerk$' \leftarrow (a' - a_{t-1})/dt$
5:         if $|\text{jerk}'| >$ jerkCap then
6:             $\delta_v \leftarrow \gamma \cdot \delta_v$; recompute $v', a', \text{jerk}'$
7:         end if
8:         $r' \leftarrow \begin{cases} r_t & \text{if } |v'| < \epsilon_v \\ \text{clamp}(a_{\text{lat},t}/v',\ r_{\min}, r_{\max}) & \text{otherwise} \end{cases}$
9:         if CheckL1($v', a', \text{jerk}'$) and CheckL2($a_{\text{lat},t}, v', r'$) then
10:            $v_t, a_t, r_t \leftarrow v', a', r'$
11:        end if
12:    end if
13: end for
```

Algorithm 3 illustrates the procedure for distributing a total velocity bias Δv over K consecutive frames after confirming the TTC trigger condition (T_0). Each split perturbation $\delta^{(j)}$ is subjected to jerk and L1/L2 checks; on violation the perturbation is attenuated by the rollback factor γ and retried, and if it still fails the perturbation is set to zero. When vehicle speed is near zero ($|v| < \epsilon_v$), yaw-rate computations are preserved or clamped to avoid numerical instability. This lightweight, practical procedure is designed to achieve the three attack goals: evasion of L1/L2 Plausibility Checks, degradation or evasion of ML-MDS detection, and systematic upward biasing of the reported TTC.

5 Implementation

This section presents the proposed attack and detection pipeline in a *reproducible* manner. Specifically, we describe a consistent pipeline covering (i) data, features, and preprocessing, (ii) ML-MDS model architectures and training/validation procedures, (iii) generation and injection of conditional ε-perturbations, and (iv) the overall experimental procedure, metrics, and statistical tests.

5.1 Environment, Data and Preprocessing

Environment. All experiments were conducted under Windows 11 using an Anaconda Python 3.11.2 environment. The pipeline is organized into seven modular stages (`step1_load` $\rightarrow$ `step7_ttc_analysis`), each executable independently from data loading to final evaluation.

Dataset. We use the **VeReMi (Vehicular Reference Misbehavior)** dataset [13], a widely adopted benchmark for misbehavior detection in V2X research. VeReMi is generated via SUMO–OMNeT++ simulation; its controlled scenarios and reproducibility make it a de facto standard for V2X security evaluations. Original logs were linearly interpolated to **10 Hz** to match the typical BSM transmission period (0.1 s). This resampling ensures numerical stability for jerk (rate of change of acceleration) computation while preserving temporal resolution.

Dataset quality and statistics. After 10 Hz resampling in Step 1, the dataset contains **100,000** frames. Class distribution is balanced with **54,800** benign frames (54.8%) and **45,200** attack frames (45.2%). The attack set comprises 19 scenarios including GridSybil, DoSRandom, and ConstSpeedOffset. Mean velocity is 10.72 m/s ($\approx$ 38.6 km/h) with a maximum of 54.7 m/s, consistent with realistic driving ranges. We use the `attack` column (0 = benign, 1 = attack) as the detection label. Because VeReMi is simulation-based, jerk variability is relatively larger than in many real-world traces; this is attributable to simulator acceleration/deceleration sampling characteristics and represents a pragmatic trade-off between physical realism and experimental reproducibility. In our Step 3 experiments, **98.4%** of L1 failures were attributable to jerk threshold violations, indicating a strong influence of this characteristic on plausibility-check pass rates.

Input features: In Step 2 we designed features to include both single-frame dynamics and short-term temporal statistics.

- *Primary kinematic variables (7):* velocity v, acceleration a, yaw rate r, lateral acceleration a_{lat}, distance d, TTC, and heading.
- *Window statistics (25):* For a window $W_f = 1.0$ s (10 frames), compute mean, std, min, max, and median for five base features (5 features $\times$ 5 statistics = 25).
- *L1/L2 residuals (2):* $|a_{\text{lat}} - vr|$ and $|a - \Delta v/\Delta t|$.
- *TTC-derived features (3):* TTC, TTC_mean, TTC_std.
- *Additional variables:* jerk, Δt, jerk_median, etc.

Feature engineering outcome. At the end of Step 2 a total of **60 features** were produced (7 primary kinematics + 25 window statistics + 2 L1/L2 residuals + 3 TTC + additional variables). The output file (`veremi_features_final.csv`, 85.33 MB) was validated to fall within physically plausible ranges: velocity ≤ 56.5 m/s, acceleration ≤ 4.5 m/s^2, and jerk ≤ 20 m/s^3.

Preprocessing.

- **Standardization:** All numeric features were standardized to zero mean and unit variance using training-set statistics.
- **Missing-value handling:** Time-based interpolation or zero-imputation was applied where necessary.
- **Class balancing:** Training used `class_weight='balanced'` to mitigate class imbalance between benign and attack labels.

5.2 Models & Training

Training data preparation (based on Step 3): A key experimental design choice is to train ML models only on samples that have *passed* the L1/L2 Plausibility Checks. From the Step 3 results, out of 100,000 total frames:

- L1 passed: **52,940** frames (52.94%)
- L2 passed: **33,100** frames (33.10%)
- **Both L1 and L2 passed**: benign **11,543** frames, attack **9,286** frames

Thus, approximately **79.58%** of original attack frames (45,200) were blocked by L1/L2, and only **20.42%** (9,286 frames) passed the rule checks. While L1/L2 serve as an effective initial filter, they are not exhaustive. This study focuses on the subset of attack samples that remain **physically plausible** after rule-based filtering to evaluate ML-MDS detection performance and adversarial vulnerability.

Splitting strategy: We adopt a *scenario hold-out* split. Training: S0, S1, S2, S5; Validation: S3; Test: S4. This ensures that models are evaluated on scenarios unseen during training, and all final results are reported on the S4 test set.

Model architectures and hyperparameters: In Step 4 we trained and evaluated three representative detector models:

- **Random Forest:**
  - Configuration: $n\_estimators = 200$, `max_depth=None`, `class_weight='balanced'`, `random_state=42`.
  - Rationale: Use a tree-based model as a baseline for stability and interpretability (feature importance).
- **MLP (Multi-Layer Perceptron):**
  - Configuration: hidden layers [128, 64], ReLU activations, dropout=0.2, batch size=256, Adam (lr=1×10^{-3}), max epochs=100, early stopping (patience=10).
  - Rationale: Lightweight neural network able to learn nonlinear patterns; serves as a baseline for non-tree models.
- **LSTM (Long Short-Term Memory):**
  - Configuration: 2 hidden layers (hidden size=128), sequence length 10 frames (1.0 s), Adam (lr=5×10^{-4}), dropout=0.3, batch size=128.
  - Rationale: Representative RNN structure for learning temporal patterns in speed/acceleration dynamics to evaluate sequence-based detectors.

All models were optimized via hyperparameter tuning on the validation set (S3) with early stopping. Recall and ROC-AUC were used as primary criteria for model selection.

Evaluation metrics: Given the focus on detection evasion, **Recall** (detection rate) and **FNR** (false negative rate) are prioritized. We also compute Precision, F1-score, ROC-AUC, and PR-AUC to assess threshold sensitivity and overall classification robustness.

Reproducibility: All library versions (e.g., scikit-learn, PyTorch), random seeds (`random_state=42`), hyperparameters (ε, T_0, jerkCap, K, etc.), and per-frame logs are published in the public repository to guarantee full reproducibility [16].

5.3 Attack Design and Injection

Trigger. The attack is activated conditionally only within intervals where the Time-to-Collision satisfies TTC $< T_0$ (conditional trigger), with the default threshold set to $T_0 = 3.0$ s.

- (i) *Frame-splitting and cross-injection*: distribute total velocity and acceleration biases (Δv, Δa) over $K = 4$ consecutive frames to minimize instantaneous deviations.
- (ii) *Jerk-dispersion control*: suppress abrupt changes by enforcing a jerk cap (jerkCap = 2.0 m/s^3).
- (iii) *L2 correction*: maintain the lateral-acceleration vs. velocity–yaw-rate constraint approximately as $a_{\text{lat}} \approx vr$ to preserve L2 plausibility.

Key parameters used in experiments: perturbation amplitude set $\varepsilon \in \{0.05, 0.1, 0.15\}$ (m/s), trigger threshold $T_0 = 3.0$ s, jerkCap= 2.0 m/s^3, and split length $K = 4$. Perturbations are injected at the payload level and then the modified messages traverse the full receiver pipeline (cryptographic verification $\rightarrow$ L1/L2 Plausibility Checks $\rightarrow$ ML-MDS).

5.4 Experimental Pipeline & Metrics

Overview of the full pipeline: The experimental workflow is organized into seven steps:

1. **Step 1 – Data loading and resampling:** Resample the original VeReMi logs to 10 Hz, yielding 100,000 frames.

2. **Step 2 – Feature computation:** Generate 60 features (7 core features + 25 window statistics + 2 residuals + 3 TTC features + additional features).

3. **Step 3 – L1/L2 Plausibility Checks:** Apply thresholds $\tau_v = 40$ m/s, $\tau_a = 7$ m/s^2, $\tau_j = 15$ m/s^3, and $\kappa = 5$ m/s^2. Results: L1 pass rate 52.94%, L2 pass rate 33.10%. Final training set: benign 11,543 frames, attack 9,286 frames.

4. **Step 4 – ML-MDS training and evaluation:** Train Random Forest, MLP, and LSTM models (Train/Val/Test splits) and measure baseline detection performance.

5. **Step 5 – Conditional ε-perturbation generation:** Apply perturbations to test cases in S4 where TTC $< T_0 = 3.0$ s. Generate perturbations according to Algorithm 3 and verify L1/L2 pass rates.

6. **Step 6 – ASR evaluation:** Compare ML-MDS performance before and after perturbation (ASR, Recall, ROC-AUC, etc.).

7. **Step 7 – TTC bias analysis:** Compare actual TTC versus reported TTC and perform statistical tests (paired t-test, Wilcoxon signed-rank test, bootstrap).

Core evaluation metrics:

- **Attack Success Rate (ASR):** Proportion of perturbed samples that are still classified as normal after the attack.

- **Recall / FNR:** Change in detection rate and false-negative rate.

- **ROC-AUC / PR-AUC:** Detection robustness across threshold settings.

- **TTC bias:** Distributional statistics (mean, standard deviation, quantiles) of the difference between actual and reported TTC.

- **Risk concealment rate:** Proportion of cases where reported TTC falls below specified thresholds (e.g., TTC $< \theta$) after perturbation.

Statistical testing: Use paired t-test when normality holds, Wilcoxon signed-rank test for non-normal cases, and bootstrap 95% confidence intervals with $n = 10{,}000$ resamples. Significance level is set to $\alpha = 0.05$.

5.5 Reproducibility

This study scripts all stages from preprocessing to evaluation (`step1_load.py` through `step7_ttc_analysis.py`) and fixes configuration via a YAML file (`base.yaml`) and a deterministic random seed (`random_state=42`) to ensure reproducibility. Core parameters (ε, T_0, jerkCap, κ, W_f) and library versions are documented in the repository, and per-frame logs plus intermediate artifacts are provided to enable exact replication of the experiments [16].

6 Evaluation

This chapter quantitatively evaluates whether the proposed conditional ε-perturbations can simultaneously evade rule-based and learning-based detectors and distort safety metrics (Time-to-Collision, TTC), based on the results of the seven-step experimental pipeline. First, we report the baseline performance (Steps 3 and 4), and then we provide answers to RQ1 (attack effectiveness) and RQ2 (safety impact).

6.1 Research Questions

This study addresses two research questions to verify whether the proposed conditional ε-perturbation can simultaneously evade rule-based and learning-based detectors.

- **RQ1 (Attack Success Rate):** How effectively does the proposed ε-perturbation attack neutralize existing defense mechanisms (L1/L2 Plausibility Checks and ML-MDS)?
  - Apply the conditional ε-perturbation defined in Step 5 to the test set (S4).
  - Evaluate detection metrics in Step 6 (ASR, Precision, Recall, F1, ROC-AUC, PR-AUC).
- **RQ2 (Safety Impact):** To what extent does the attack bias the reported TTC upward relative to the true TTC, and what are the implications for vehicle collision risk assessment?
  - Compute the bias between true TTC and reported TTC in Step 7 (mean, standard deviation, quantiles).
  - Assess statistical significance using paired t-test, Wilcoxon signed-rank test, and bootstrap 95% confidence intervals.
  - Quantify risk concealment rates for thresholded TTC events (e.g., TTC $< 1.0\,\mathrm{s}$, $< 1.5\,\mathrm{s}$, $< 2.0\,\mathrm{s}$).

6.2 RQ1: Attack Success Rate (Results for Steps 5–6)

Step 5 — Conditional ε-perturbation generation. We generated conditional ε-perturbations for **3,250** original attack samples in the test set (S4) that satisfied the trigger condition TTC < 3.0 s.

- Perturbation parameters: $\varepsilon \in \{0.05, 0.1, 0.15\}$ m/s, $T_0 = 3.0$ s, $K = 4$, jerkCap $= 2.0$ m/s^3.
- L1/L2 pass rate after injection: **99.8**% (3,242 / 3,250).
- Mean perturbation magnitudes: $\delta_v \approx 0.08$ m/s, $\delta_r \approx 0.05$ rad/s.

In other words, the proposed attack successfully bypassed the plausibility checks in nearly all targeted frames (99.8%).

Step 6 — ASR evaluation results. The comparison of ML-MDS performance before and after perturbation is summarized in Table 2.

Experimental results.

1. **Random Forest:** The baseline recall declined marginally from 76% to 73% after perturbation (ROC-AUC $\approx$ 0.51).
2. **MLP:** Performance was poor from the outset (recall 7%, ROC-AUC $\approx$ 0.50) and showed little change after perturbation.
3. **LSTM:** The baseline recall dropped markedly from **98.7%** to **42.2%** (a decline of 56.5 percentage points), and the ASR increased from 1.3% to **57.8%**.

Table 2: ML-MDS performance before and after conditional ε-perturbation (S4 test set)

Model	**Scenario**	**ASR (%)**	**Recall**	**F1**	**ROC-AUC**
Random Forest	Before	24.3	0.76	0.62	0.51
	After	27.5	0.73	0.59	0.50
	Δ	+3.2	–0.03	–0.03	–0.01
MLP	Before	93.0	0.07	0.12	0.50
	After	97.8	0.02	0.04	0.49
	Δ	+4.8	–0.05	–0.08	–0.01
LSTM	**Before**	**1.3**	**0.987**	**0.978**	**0.9987**
	After	**57.8**	**0.422**	**0.434**	**0.8317**
	Δ	**+56.5**	**–0.565**	**–0.544**	**–0.167**

Answer to RQ1. The proposed conditional ε-perturbation achieves the following:

- Near-complete evasion of L1/L2 plausibility checks: **99.8% pass rate**.
- Substantial degradation of the LSTM detector: recall reduced from **98.7%** to **42.2%**.
- Increase of attack success rate to **57.8%**, causing over half of attacks to be misclassified as benign.

Conclusion: The proposed attack can simultaneously defeat rule-based plausibility filters and learning-based detectors, demonstrating a fundamental vulnerability in current V2X security architectures.

6.3 RQ2: Safety Impact (Step 7 TTC Analysis)

This section quantitatively evaluates the safety impact of the proposed attack by measuring the difference between the **true TTC** and the **reported TTC**. In Step 7, we analyzed **3,242 perturbed samples** to measure TTC bias, distributional shifts, and the rate of concealed hazardous events.

Evaluation metrics.

- **TTC bias:** Defined as (reported TTC – true TTC), reported in terms of mean, median, and standard deviation.
- **TTC distribution shift:** Comparison of lower 5%, 25%, and 75% quantiles before and after perturbation.
- **Hazard concealment rate:** Number and ratio of hazardous events (TTC $< \theta$) that became concealed after perturbation.

Step 7 — Experimental results on TTC bias. The statistics of TTC before and after perturbation are summarized in Table 3.

Experimental results.

1. **TTC bias:** After applying the perturbation, the mean Time-to-Collision increased from 2.79 s to 9.80 s, corresponding to an absolute change of +**7.02** seconds (a relative increase of 251%).

Table 3: TTC statistics before and after perturbation ($n = 3{,}242$, Step 7)

Metric	**Original TTC**	**Perturbed TTC**	**Bias (Δ)**
Mean (s)	2.79	9.80	**+7.02**
Median (s)	2.82	10.00	**+6.99**
Std. Dev (s)	1.30	0.69	–0.61
Min (s)	0.50	5.13	+0.99
Max (s)	5.00	10.00	+9.50
25 % quantile (s)	1.66	10.00	+5.93
75 % quantile (s)	3.90	10.00	+8.23
Statistical tests:			
Paired t-test: $t(3241) = 271.13$, $p < 0.001$			
Wilcoxon signed-rank: $W = 0.00$, $p < 0.001$			
Effect size (Cohen's d): **4.76** (very large)			

Answer to RQ2. The proposed conditional ε-perturbation yields the following effects:

- Systematic upward bias of the mean TTC by +**7.02** seconds, causing reported TTC to be substantially larger than actual TTC.
- Complete concealment of hazardous events across all evaluated risk thresholds (i.e., **100**% concealment rate for the thresholds considered).
- Strong statistical significance for the observed bias ($p < 0.001$) with a very large effect size (Cohen's $d = 4.76$).

Conclusion: The attack severely distorts the safety-critical metric (TTC), effectively neutralizing collision-avoidance assessments and demonstrating a fundamental safety vulnerability in V2X systems.

7 Conclusion

This study empirically presents a novel insider threat scenario in which an attacker possessing a legitimate certificate can **fully comply with L1/L2 plausibility constraints while evading a machine learning–based Misbehavior Detection System (ML-MDS)**. We designed a **Conditional ε-Perturbation** technique that activates only when the Time-to-Collision (TTC) falls below a predefined threshold, and derived the following results through quantitative experiments based on the VeReMi dataset.

1. The proposed attack successfully passed the L1/L2 plausibility checks with a rate of 99.8%, while degrading the detection performance (Recall) of the LSTM-based ML-MDS from **98.7% to 42.2%** (a drop of 56.5 p.p.). This demonstrates that the attack can simultaneously evade both rule-based and learning-based detection mechanisms.
2. By applying **cross-frame injection** and **smooth perturbation (jerk control)**, the perturbations remained indistinguishable from actual sensor noise, proving the feasibility of stable and stealthy evasion.
3. During activated attack periods, the reported TTC was on average **+7.02 s longer** than the actual TTC, and all hazardous events (TTC $<$ 3 s) were **completely concealed (100%)**. This demonstrates that the attack systematically obscure collision risks, posing a critical threat to the safety assurance of autonomous decision-making modules.

These findings provide the **first empirical evidence that the conventional "cryptographic integrity + plausibility check" paradigm alone cannot guarantee V2X security**, and reveal that even ML-based detectors possess **structural vulnerabilities to adversarial perturbations**. Therefore, future V2X security frameworks should evolve in the following three directions:

- Formalize the concept of **semantic-valid adversarial attacks**, demonstrating that physically plausible perturbations can undermine overall system trustworthiness.
- Experimentally implement TTC-aware conditional ε-perturbations and quantitatively prove that they can simultaneously disable existing L1/L2 rules and ML-MDS frameworks.
- Redesign ML-MDS architectures with **adversarial robustness in mind**, integrating **MLSecOps-based randomization, data sanitization, and adversarial training** techniques.

This work represents a proof-of-concept (PoC) simulation-level study. Future work should include (i) **real-vehicle validation**, (ii) **integration with PQC-hybrid authentication frameworks**, and (iii) **extended evaluation on advanced deep learning–based MDS architectures (LSTM, GNN, Transformer, etc.)**. A limitation of this work concerns the absence of analysis on the **transferability across models**—whether perturbations generated for one ML-MDS architecture remain effective against others. Addressing this will lay the foundation for the design of a next-generation V2X security framework that bridges the current **semantic integrity gap**.

References

[1] H. Hartenstein and L. Laberteaux, “A tutorial survey on vehicular ad hoc networks,” *IEEE Communications magazine*, vol. 46, no. 6, pp. 164–171, 2008.

[2] J. B. Kenney, “Dedicated short-range communications (dsrc) standards in the united states,” *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.

[3] T. Yoshizawa, D. Singelée, J. T. Muehlberg, S. Delbruel, A. Taherkordi, D. Hughes, and B. Preneel, “A survey of security and privacy issues in v2x communication systems,” *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–36, 2023.

[4] M. Hasan, S. Mohan, T. Shimizu, and H. Lu, “Securing vehicle-to-everything (v2x) communication platforms,” *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 4, pp. 693–713, 2020.

[5] T. Leinmüller, E. Schoch, and F. Kargl, “Evaluation of plausibility checks for vanets,” in *6th International Conference on ITS Telecommunications*, pp. 1–6, 2008.

[6] A. Boualouache and T. Engel, “A survey on machine learning-based misbehavior detection systems for 5g and beyond vehicular networks,” *IEEE Communications Surveys & Tutorials*, vol. 25, no. 2, pp. 1128–1172, 2023.

[7] R. W. Van Der Heijden, S. Dietzel, T. Leinmüller, and F. Kargl, “Survey on misbehavior detection in cooperative intelligent transportation systems,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 779–811, 2018.

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

[9] A. A. Andrade Salazar, P. D. McDaniel, R. Sheatsley, and J. Petit, “Physics-based misbehavior detection system for v2x communications,” *SAE International Journal of Connected and Automated Vehicles*, vol. 5, no. 3, 2022.

[10] N. Aksan, L. Sager, S. Hacker, R. Marini, J. Dawson, S. Anderson, and M. Rizzo, “Forward collision warning: clues to optimal timing of advisory warnings,” *SAE International journal of transportation safety*, vol. 4, no. 1, p. 107, 2016.

[11] C. Xiao, R. Deng, B. Li, T. Lee, B. Edwards, J. Yi, D. Song, M. Liu, and I. Molloy, “Advit: Adversarial frames identifier based on temporal consistency in videos,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3968–3977, 2019.

[12] J. Jagelčák, J. Gnap, O. Kuba, J. Frnda, and M. Kostrzewski, “Determination of turning radius and lateral acceleration of vehicle by gnss/ins sensor,” *Sensors*, vol. 22, no. 6, p. 2298, 2022.

[13] R. W. Van Der Heijden, T. Lukaseder, and F. Kargl, “Veremi: A dataset for comparable evaluation of misbehavior detection in vanets,” in *International conference on security and privacy in communication systems*, pp. 318–337, Springer, 2018.

[14] V. D. Contributors, “Veremi dataset.” `https://github.com/VeReMi-dataset`, 2018. Accessed: 2025-09-30.

[15] Z. H. Khattak, “Cybersecurity vulnerability and resilience of cooperative driving automation for energy efficiency and flow stability in smart cities,” *Sustainable Cities and Society*, vol. 106, p. 105368, 2024.

[16] J. Park, “Adversarial attacks on plausibility checks in v2x security code repository.” `https://github.com/YeormDal/Adversarial-Attacks-on-Plausibility-Checks`, 2025. Accessed: 2025-09-30.