Detecting Injection Attacks on Cooperative Adaptive Cruise Control

Marco Iorio, Fulvio Risso, Riccardo Sisto
Politecnico di Torino, Torino, Italy
{name.surname}@polito.it

Alberto Buttiglieri, Massimo Reineri
Italdesign, Moncalieri, Italy
{name.surname}@italdesign.it

Abstract—**Although vehicle platooning promises to improve transportation efficiency and safety by leveraging communication between convoy members, preliminary results in previous work suggest that cyber-attacks could deceive many Cooperative Adaptive Cruise Control algorithms, hence endangering the safety of every participant. This paper deeply analyzes the case of injection attacks. First, we introduce an extensive security analysis carried out through realistic simulations, to demonstrate how even slight and smooth falsification attacks do succeed in fooling the CACC controllers and cause numerous vehicle crashes. Second, we present a novel misbehavior detection technique. It leverages the correlation between multiple motion parameters concerning both single and consecutive vehicles to evaluate the plausibility of the information received from the other members. Extensive validation confirms the effectiveness of the technique proposed: overall, it succeeds to detect all the attacks simulated and prevents the occurrence of safety-critical situations.**

Index Terms—**Platooning, V2V security, Injection attacks, Misbehavior detection technique, CACC, VANET**

I. Introduction

Vehicle platooning, characterized by groups of either commuters or heavy-duty vehicles traveling very closely together, safely at high speed, has been widely recognized as a promising application in the scope of Intelligent Transportation Systems [1]. Shortened reaction times, limited distances, reduced speed oscillations and aerodynamic drags are all essential factors contributing to an increase in safety, comfort and roadway capacity, while reducing fuel consumption and the consequent emission of harmful pollutants [2], [3].

To enable vehicular platooning, one key component is the automated longitudinal control, a control algorithm maintaining the desired distance from the preceding vehicle and adapting the speed to the convoy pace. To this end, Adaptive Cruise Control (ACC) algorithms, based on the information gathered from local sensors, such as radar, lidar and cameras, are attractive due to their simplicity. However, it has been shown they are unable to provide string stability [4] if a constant spacing policy is adopted (i.e. the distance between two vehicles is independent from the current speed) and they are associated with fairly high headways at highway speeds [5, Chapter 6]. To face these issues, a wide range of Cooperative ACC (CACC) algorithms has been proposed. They leverage vehicle-to-vehicle (V2V) communications to obtain extensive information about positions, speeds and accelerations from the other members of the platoon, and especially from the leader, to allow for quicker reactions and guarantee string stability, even when a small and constant headway is desired [6].

Although previous research confirmed that a performance increase can be achieved by coupling vehicle platooning and V2V communications, the reliance upon remotely peer-generated data opens, at the same time, the possibility for malicious attacks. Indeed, since CACC algorithms compute the currently desired longitudinal motion based both on locally available information and some knowledge received from the other platoon members, such as their speed and acceleration, erroneous or falsified pieces of data may cause incorrect computations that, in the extreme case, might eventually lead to a crash. During recent years, a great amount of effort has been put in by standardization bodies to define the various network layers necessary to enable VANETs, along with the associated security properties that need to be guaranteed. To this extent, the most relevant standard is IEEE 1609.2, which defines the methods to secure application messages guaranteeing their authenticity, integrity and confidentiality, if required. Nonetheless, it targets exclusively at the protection from the so-called external attackers, those not authorized to participate in the communication, and guarantees that only the messages originated from legitimate vehicles are processed by the application layer of the others.

This paper, instead, focuses on the attacks carried on by internal adversaries, those that are currently granted access to the network as legitimate vehicles. In such a situation, traditional cryptography-based methods are not suitable to detect misbehavior, since the attackers themselves do also own the keys necessary to establish the communication: complementary protections, based on the analysis of the content of the beacon messages received, need to be established. Previous work identified two main categories of attacks that can be perpetrated against CACC algorithms, namely *jamming* and *data injection*, and showed they can seriously endanger the safety of the victims [7], [8]. Jamming encompasses different techniques aiming to forbid or severely delay the reception of legitimate beacons: it can be perpetuated by violating on purpose the MAC layer, forcing the other participants to wait, or at higher layers, by means of DoS attacks that cause the victims to perform useless tasks. Although not being strictly limited to internal adversaries, jamming attacks cannot be in any case prevented by traditional cryptography. The second malicious technique, on the other hand, requires a vehicle controlled by an attacker to send modified beacons containing fallacious information; nonetheless, similar side-effects can be involuntarily originated by undetected malfunctioning sensors.

It is worth pointing out that the injection attack needs to be perpetrated directly on the originating vehicle, since the integrity of the network messages is assumed to be guaranteed by the security measures of the communication protocol. In the remainder of this paper, we will concentrate exclusively on injection attacks, being deemed to be more interesting and challenging, especially from the detection point of view.

Although van der Heijden et al. [8] already studied the effects of both jamming and injection attacks on CACC algorithms by means of simulations, some interesting facets still remain to be explored. In particular, they did not feed the CACC algorithms with the information obtainable from local sensors, such as a radar, even when available (e.g. to assess the speed of the predecessor); indeed, they always leveraged the measurements obtained through V2V, intrinsically easier to be falsified. Additionally, the values applied during the data injection were rather extreme, to such an extent that, in most cases, the attack could have been detected by even trivial validity checks, as acknowledged by the authors themselves. Moreover, to the best of our knowledge, no misbehavior detection algorithms have been proposed so far that exploit the peculiarities of vehicular platooning, and in particular the correlation between different physical variables (i.e. position, speed and acceleration), to detect attempts to perform injection attacks. In this paper, we fill this gap through our twofold contribution. First, we extend the analysis about the effectiveness of data injection attacks, overcoming the identified limitations. Second, we propose and validate a novel misbehavior detection algorithm that, combining the information coming from different sources (i.e. V2V and radar, when available) and referred to different but correlated physical characteristics (i.e. position, speed and acceleration), aims to identify ongoing injection attacks.

The remainder of this paper is organized as follows. Section II discusses the existing studies about different security attacks targeting vehicle platooning and possible misbehavior detection techniques. In Section III we present an overview of the Plexe simulator, to outline some preliminary modifications, and a brief comparison between the different car-following models therein implemented. Section IV describes the attacker model adopted and illustrates the results obtained by simulating the injection attacks. In Section V we detail our novel detection technique to identify ongoing injection attacks, experimentally validated in Section VI. Finally, Section VII draws the main conclusions and proposes directions for further research.

II. Related Work

During recent years, the analysis of security requirements in the broad context of VANETs has been a proficient field of research. In 2016, Kerrache et al. [9] presented a taxonomy of possible attacks, together with a survey of available countermeasures, encompassing both cryptography-based and trust management-oriented solutions. Nonetheless, the peculiarities of vehicular networks, such as the completely distributed and peer-to-peer architecture, the strict timing constraints and the privacy requirements, still leave different open points to be studied, especially when focusing on specific applications.

Zooming in on the field of vehicle platooning, most initial research focused on either control theory aspects or the network protocols and requirements, leaving the associated security facets partially unexplored. In 2015, Amoozadeh et al. [7] presented one of the first investigations about a series of possible attacks perpetrated against connected vehicles streams, both at the network and application layer. Their simulations, performed using the VENTOS platform [10], showed how platoon instability emerged as a result of message falsification, while radio jamming attacks succeeded in forcing the CACC algorithm to degrade to non-cooperative ACC. Finally, they pointed out the necessity for misbehavior detection techniques to identify compromised vehicles and secure CACC controllers. Subsequently, van der Heijden et al. [8] continued along this line of research by simulating with greater detail the effects of both jamming attacks and malicious data injections. They considered the three car-following models implemented at that time by the selected simulator, Plexe, and concluded that all the algorithms are affected when under attack, leading in many cases to potentially fatal crashes. Although being already rather complete as an analysis, we identified some possible aspects and limitations still worth of investigation. In particular, their work neglected the possibility to gather some of the physical measurements through a radar, intrinsically harder to fool, rather than using the ones obtained through V2V. Additionally, they unfairly compared the behavior of the controllers using different spacing settings and caused easy-to-detect abrupt and extreme changes in the values received when starting the injection attacks: in this paper, instead, we implemented smoother variations, to make the attack much harder to detect. Finally, they did not study the effects of falsifying at the same time multiple fields in the transmitted beacons.

Concerning misbehavior detection in the scope of VANETs, multiple researchers already presented different sensor fusion techniques and plausibility checks aiming to detect misbehaving nodes and internal attackers [11]–[14]. Nonetheless, to the best of our knowledge, the only proposal directly exploiting the peculiarities of vehicle platoons for this purpose is the paper recently presented by Lu et al. [15]. In their work, the authors leveraged the spatial relations between multiple members to obtain an additional piece of information that can be evaluated for the detection of possible injection attacks. However, they did not take advantage from the existing correlations between the different physical measurements present in V2V beacons.

III. Simulating Vehicle Platooning

Given the prohibitively high costs and the safety risks associated with real-world prototypes, in the following we will exploit a simulator to evaluate the effectiveness of the injection attacks and validate our proposed countermeasures. Among the different available alternatives, we identified Plexe [16], version 2.1, as the best candidate, having it been explicitly designed to simulate vehicle platooning and being easily extensible. Plexe implements a set of CACC algorithms on the top of Veins, a vehicular network simulation framework coupling the well-established OMNeT++ and SUMO simula-

TABLE I
COMPARISON BETWEEN DIFFERENT ACC/CACC ALGORITHMS.

Controller	Policy	Predecessor			Leader		
		d	s	a	p	s	a
ACC	CTH	✓	✓				
PATH	CVS	✓	✓	📶		📶	📶
Consensus	Both	✓			📶	📶	
Flatbed	CVS	✓	✓			📶	
Ploeg	CTH	✓	✓	📶			

tors. The former models the communication between different nodes, implementing a complete vehicular communication stack based on IEEE 802.11p, while the latter provides realistic microsimulations of physical vehicles.

Concerning the high-level longitudinal controllers, i.e. the ones computing the desired motion parameters given the local measurements and the messages received from the other platoon members, we focus on the car-following models implemented by Segata et al. in Plexe. Along with a non-cooperative Adaptive Cruise Control (ACC) [5, Chapter 6], the simulator provides four main cooperative algorithms, each one associated with some peculiar characteristics. The first car-following model is a constant spacing controller based on classical control theory and developed within the scope of the California PATH program [5, Chapter 7]. The second one is a consensus algorithm proposed by di Bernardo et al. [17], characterized by a reconfigurable topology and capable of exploiting the information (i.e. positions and speeds) coming from all the other members of the platoon. However, we maintained the default implementation, characterized by the leader- and predecessor-following topology (i.e. every vehicle considers only the messages received from the leader and its predecessor). The Flatbed controller [18], on the other hand, exploits a model considering the platooning vehicles loaded on a virtual flatbed tow truck to emulate a constant spacing policy. Finally, the last controller considered is the algorithm designed by Ploeg et al. [19], which is characterized by a constant time-gap headway policy (i.e. the distance between two vehicles depends on their current speed) and requires only the information concerning the preceding vehicle. Table I summarizes the main characteristics of the different algorithms, highlighting for each one the associated spacing policy (CVS: Constant Vehicle Spacing, CTH: Constant Time Headway, Both). Additionally, it points out the measurements, regarding both the predecessor and the leader vehicles, required for their operation (d: distance, p: position, s: speed, a: acceleration) and whether they need to be obtained through V2V (📶). This comparison is especially important to predict in advance the types of injection attacks that could affect each controller.

Before simulating the injection attacks, we performed some preliminary modifications to Plexe, in order to make the implementation and the behavior of the different controllers more uniform. In particular, we included the possibility to select whether to use the measurements obtained from the radar or the values gathered from V2V communications, in case both are available. Finally, we implemented a model to simulate the uncertainties associated with each sensor: once an exact value is obtained from SUMO, it is replaced by another one uniformly extracted from the uncertainty interval.

IV. ASSESSING THE EFFECTS OF INJECTION ATTACKS

This section presents the enriched simulations performed to evaluate the impact of different injection attacks on the CACC algorithms. We begin describing the attacker model adopted, then we proceed with an overview of the simulation setup and we conclude by presenting the outcome of the experiments and a brief discussion about the results obtained.

A. Attacker Model

Since we are concentrating on the effects of security attacks targeting CACC algorithms, we assume an already established platoon traveling along a straight road. The different members exchange periodic beacons secured by means of well-established cryptographic primitives (e.g. IEEE 1609.2) to enforce message authentication, integrity and replay protection. Consequently, we assume the attacker having already gained the control over a legitimate member of the platoon and, thus, being able to modify the information contained in the beacons without being detected by means of integrity checks. For the purpose of this analysis, it is not relevant how the attacker obtained its access, e.g. through software compromise or by physical manipulation of the vehicle itself (either at sensor, network or software level). However, we assume the attacker to be precluded from performing any physical action (i.e. accelerating or braking) on the vehicle under his control.

In our model, as in [8], the primary goal of the attacker is to cause a crash within the platoon, by destabilizing the string of vehicles and provoking oscillations in the distance maintained between consecutive members. Concerning the attacker's position, we consider two distinct situations. First, we assume the attacker to control a *non-leader* vehicle, a situation in which it is supposed to be able to directly influence only the behavior of its immediate follower. Second, we analyze the possibility it gained the control over the *leader* vehicle, so to potentially endanger all the other platoon members.

All the injection attacks have been simulated by replacing the content of legitimate beacons with fallacious values before their transmission. The different variables considered during the analysis encompassed the ones required by the longitudinal controllers (i.e. position, speed and acceleration) both individually and all at the same time. To make the attacks harder to detect, we opted for smooth variations in the advertised values, instead of abrupt and instantaneous changes. Figure 1 graphically depicts one illustrative attack applied to speed values, which is characterized by two main phases. Initially, the advertised values slowly deviate from the legitimate ones at the configured linear rate. Once the maximum specified divergence is reached, the injection attack comes to a steady state and it is no longer increased. Clearly, the smaller the increase rate is configured, the longer it takes to reach the attack limit and, potentially, the harder the misbehavior detection is. Finally, it is worth pointing out that, although the legitimate values were

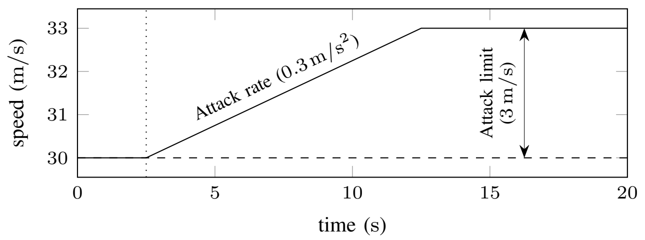

Fig. 1. Graphical representation of an injection attack applied to speed values advertised through V2V (dashed line: legitimate, solid line: falsified).

supposed constant for graphical reasons, a similar behavior would have been obtained with a varying speed profile.

B. Simulation setup

All the simulations to assess the effects of the various injection attacks have been performed in Plexe, comparing the reaction exhibited by the four car-following models considered for the evaluation. Although maintaining the main controller parameters to their default values, as provided in the respective papers, we decided to level out the desired distance between consecutive vehicles, in order to guarantee as uniform results as possible: in [8], instead, the authors unfairly compared the controllers maintaining different spacing settings for each of them. In detail, we elected $10\,\mathrm{m}$ (corresponding to a time-headway gap of $0.36\,\mathrm{s}$ at $100\,\mathrm{km/h}$) as our predefined choice, considering a balance between a too packed setup, which cannot be safely maintained by all the considered controllers in case of hard braking, and excessive distances, strongly limiting the advantages of vehicle platooning.

Concerning the overall scenario, we opted to simulate a platoon of eight vehicles traveling along a straight highway at a constant desired speed of $100\,\mathrm{km/h}$. Albeit being very simple, this scenario is deemed to represent the most frequent operating condition in realistic situations. Moreover, it aims to highlight as much as possible the effect of the attacks themselves, without the interference caused by external perturbations. Nonetheless, for the sake of completeness, we also evaluated the outcome of the injection attacks when the speed of the leader varies, as simulated by the sinusoidal scenario provided by Plexe.

Every vehicle is assumed to be equipped with all the sensors necessary to gather the measurements required by the CACC algorithms, including a GPS, which is nowadays considered to be mostly ubiquitous in vehicular networks, and a radar. Regard the latter, it is always leveraged to measure the relative distance from the preceding vehicle, thanks to its much higher precision and reliability compared to GPS triangulation. Concerning the assessment of the predecessor speed, on the other hand, we analyzed and compared two alternative solutions, encompassing the usage of either the values received through V2V or the measurements gathered using the radar.

Table II summarizes the parameters associated with the simulated injection attacks, performed considering both the different variables independently and all at the same time. One may complain about the specific selection, arguing that different values could have led to somehow different results. However, it is worth pointing out that, besides focusing on the exact

TABLE II
INJECTION PATTERNS ANALYZED DURING THE SIMULATIONS.

Variable	Attack rate	Attack limit
Position	$2.5\,\mathrm{m/s}$	$50\,\mathrm{m}$
Speed	$0.5\,\mathrm{km/h/s}$	$10\,\mathrm{km/h}$
Acceleration	$0.05\,\mathrm{m/s^3}$	$1\,\mathrm{m/s^2}$

values, the attacks have been configured to be smooth, slowly incrementing in time and characterized by variations compatible with normal operation: thus, they are assumed to be hard to detect. Concluding, all the injection patterns encompass positive values: preliminary analyses, in fact, revealed how these tend to cause a reduction in the maintained distances. Hence, they are associated with much higher safety risks compared to the opposite behavior inducted by negative variations, simply increasing the space occupation of the platoon.

C. Numerical results

The full results of the simulations concerning the constant scenario are presented in Fig. 2. Each line corresponds to one single run, characterized by a specific combination of car-following model, falsified variable and radar configuration, when relevant. As an evaluation metric to measure the effectiveness of the different attacks, we concentrated on the instabilities introduced in the distance between consecutive vehicles, due to its strong safety implications. Hence, for each combination, we represented the interval $[\min_{t,i}(d), \max_{t,i}(d)]$, where t is simulation time, i the index of the considered vehicle and d corresponds to the distance between the i^{th} vehicle and its predecessor. In other words, the interval is bounded by the minimum and maximum distances measured between any two vehicles during the whole simulation. It is worth remembering that the default distance in absence of attacks is $10\,\mathrm{m}$, while reaching the $0\,\mathrm{m}$ spacing line corresponds to a crash.

Figure 2a depicts the outcome of the falsification attacks when perpetrated by a non-leader vehicle. Concerning the different variables, the position injection was clearly ineffective, since the distance from the preceding vehicle was directly evaluated using the radar; similar considerations hold also for the speed falsification, in case the measurements are gathered through this sensor. Both speed (if the radar is not exploited) and acceleration forgery, on the other hand, succeeded in fooling the PATH and Ploeg controllers, causing in most situations a crash, while they introduced no perturbations when the other controllers were adopted. In most cases the outcome of the injection attack can be easily predicted looking at whether the falsified variable is considered or not by each controller. Instead, the low sensitivity to the speed falsification demonstrated by the Flatbed controller depends on the default settings, assigning very limited importance to such variable. Finally, the attack encompassing all the variables originated slightly larger oscillation intervals than the most successful injection alone. It is worth mentioning that crashes always occurred between the attacker and its direct follower.

Moving to the attacks perpetrated by the leader (Fig. 2b), it is evident the higher success rate compared to the previous

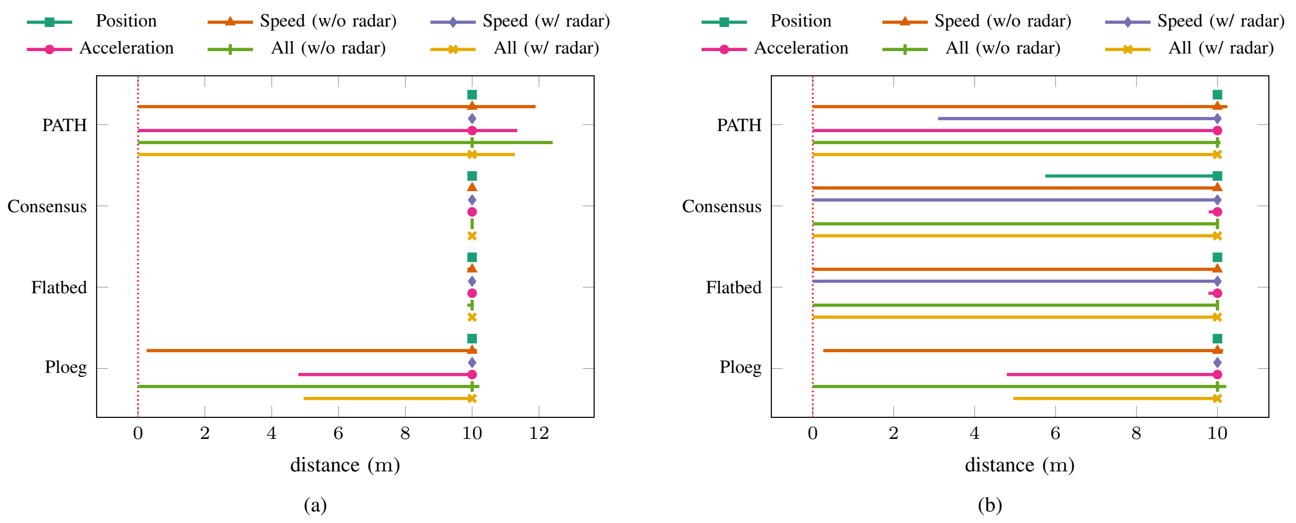

Fig. 2. Distance from the preceding vehicle under different injection attacks perpetrated by (a) a non-leader vehicle, (b) the leader vehicle. Each interval represents the minimum and maximum distances measured between any two vehicles (expected distance: 10 m); the zero line corresponds to an occurred crash.

situation. Nonetheless, exactly the same results are associated with the Ploeg algorithm in both cases, since it leverages only the information concerning the preceding vehicle. Regarding the other controllers, a crash was almost always the outcome of both speed and acceleration falsifications, with the availability of the radar being insignificant unless the PATH algorithm was selected. The Consensus controller, on the other hand, was the only one susceptible to the position injection attack, due to its peculiar characteristics. As before, the synchronous attack caused the most significant perturbations, combining the effects of the individual attacks. Finally, the crashes occurred mostly between the attacker itself and the second vehicle, although in some cases, especially when the radar was used to measure relative speeds, the third vehicle bumped into its predecessor.

D. Discussion

The analysis presented above confirmed the effectiveness of the examined injection attacks and the associated security implications. As expected considering the comparison matrix introduced in Table I, each controller is susceptible to different attacks, depending on its peculiar characteristics. The PATH algorithm is the one that suffers the most when receiving counterfeit beacons, due to its dependence on multiple variables. Both the Consensus and the Flatbed controllers, on the other hand, are only susceptible to speed falsifications perpetrated by the leader vehicle. Nonetheless, the former relies on the knowledge of the GPS position of the convoy leader, a piece of information that may not be always available with the required precision even in the absence of attackers. The latter, on the other hand, simply requires all members to share a common speed value: although the default implementation exploits the one obtained from the leader for simplicity, agreement algorithms may be developed to mitigate the presence of attackers. Finally, the Ploeg algorithm, leveraging only the information coming from the predecessor, shows a more limited susceptibility to the considered injection attacks, suffering only when bogus acceleration values are received (if the relative speed is evaluated using the radar). However, it is the only controller analyzed based on a CTH spacing policy, implying a different desired distance depending on the current convoy pace. Concerning the sinusoidal scenario, no relevant differences emerged compared to the already presented simulations, with only very little additional perturbations introduced by the oscillations in the speed of the leader vehicle.

V. Detecting Injection Attacks

No widespread diffusion of vehicle platooning can occur until viable countermeasures are developed to mitigate the effects of possible attacks. This section presents our novel misbehavior detection technique that, continuously analyzing different pieces of information, possibly coming from multiple uncorrelated sources, evaluates the plausibility of the messages received to detect as soon as possible ongoing attacks. The algorithm proposed leverages the peculiarities of vehicle platooning and it is based on two main insights. First, it acknowledges that the different physical variables regarding a single vehicle (i.e. position, speed and acceleration) cannot evolve independently: instead, they are bound by the well-known kinematic equations. Second, moving in a convoy, consecutive vehicles tend to react to the same stimuli, thus showing very similar motion patterns: inconsistencies could be a clear clue of a falsification attack.

A. Correlation between physical variables

Let us initially delve into the relationship between the correlated variables and consider a 1-D scenario, as represented by a straight highway. For a sufficiently small time interval ΔT, we can approximate the evolution of the physical quantities as:

$$\begin{cases} p(t+\Delta T) = p(t) + \Delta T \cdot v(t) + 0.5 \cdot \Delta T^2 \cdot a(t) \\ v(t+\Delta T) = v(t) + \Delta T \cdot a(t) \end{cases} \tag{1}$$

where $p(t)$, $v(t)$ and $a(t)$ represent respectively the position, speed and acceleration of one vehicle at time t. Thus, the intuition behind this first pillar of the misbehavior detection technique consists in comparing the values received through

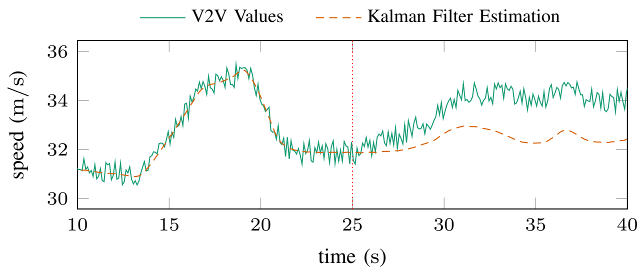

Fig. 3. Graphical comparison between the speed values received through V2V and the ones estimated using the Kalman Filter. The red vertical line indicates the beginning of the injection attack: at this point, the estimation soon starts to diverge from the values received.

V2V with the ones estimated by means of (1), to understand whether subsequent beacons contain plausible pieces of data or not. If a single variable is injected, in fact, it would immediately start to diverge from the predicted values, acting as a warning signal of a possible misbehavior. This approach could be easily bypassed in case a coordinated attack is performed, i.e. the three correlated variables are modified respecting the same equations used for the prediction. Nonetheless, forcing the attack to encompass all the variables contained in a beacon makes the injection itself easier to spot using complementary techniques, as detailed later on.

Although promising, the naïve technique just presented, and in particular the estimation of the physical variables by means of (1), is characterized by an intrinsic limitation. In fact, due to approximations, sensor uncertainties and real-world non-linearity, the estimation would immediately start to diverge from the correct values, becoming soon useless to be compared with the received ones. To face this issue, the bare estimation is replaced with the values obtained by means of a Kalman Filter [20]. It is an algorithm estimating the state of a process by combining theoretical predictions and measurements coming from noisy sensors, to correct the real-world drifts that cannot be captured by simplified models. In other words, the filter tends to mediate between the estimation and the actual measurements depending on the associated confidence intervals, converging towards the true values very rapidly.

Summarizing the general idea, during normal operation the values received from the network and the estimations generated by the Kalman Filter are expected to be similar, confined in a very limited tolerance band. Instead, injected beacons are supposed to cause an increasing divergence, thus being an indicator of an ongoing attack. Figure 3 provides a graphical representation of this expected behavior. On the left side of the graph, when legitimate beacons are sent, the Kalman Filter is perfectly able to follow the measurements received through V2V, even if characterized by an evident noise. Soon after the beginning of the injection attack, on the other hand, the values received and the estimation start to become increasingly different, immediately highlighting the inconsistency.

B. Correlation between different vehicles

Considering the correlation between different physical variables does not prevent an attacker from injecting all of them in a

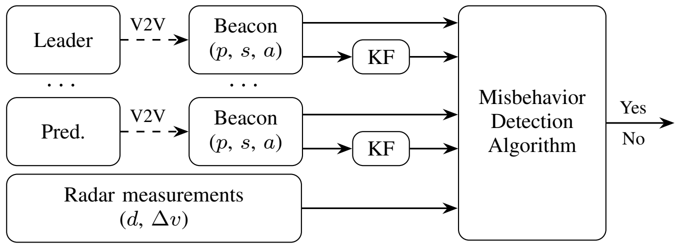

Fig. 4. High-level representation of the inputs required the misbehavior detection algorithm, including the V2V beacons, the estimations computed by the Kalman Filters and the radar measurements (if available). The output is a boolean value indicating whether an attack is currently detected or not.

coordinated way. To this end, the second pillar of the proposed misbehavior detection technique is based on the comparison between the behavior shown by different vehicles belonging to the same platoon. Indeed, being constrained by identical CACC algorithms, consecutive members are expected to act coherently. Moreover, as an additional validity check, it is possible to exploit the fact that each vehicle is constrained to maintain the configured distance from its predecessor: a violation of this fundamental rule is a clear indication of a possible misbehavior. Although one may leverage only this last, seemingly trivial validity check to identify ongoing attacks, it would delay the detection until the injection already caused most of its effects, by strongly reducing the safety margins. Instead, the combination of multiple evaluations aims to anticipate as much as possible the detection instant.

Figure 4 summarizes at a high-level the information required by the misbehaviour detection algorithm to operate. Assuming the attacker controls at most one member of the platoon, every vehicle can independently execute the misbehavior detection algorithm on its own, by leveraging only the messages received and the local estimations computed through Kalman Filters. Nonetheless, every vehicle can use its radar to gather privileged information concerning its own predecessor. Therefore, by comparing and combining independently gathered measurements, every vehicle is able to enrich its own knowledge, making it harder to be deceived by its predecessor and, consequently, increasing the safety of the whole platoon.

C. Formalization of the misbehavior detection technique

In the following, we formalize the different components of the misbehavior detection technique described so far. Let l_i, p_i, v_i and a_i represent respectively the length, position, speed and acceleration of the i^{th} vehicle in the platoon, with $d_i = p_i - p_{i+1} - l_i$ and $\Delta v_i = v_i - v_{i+1}$ denoting the relative distance and speed from its follower; moreover, the superscripts indicate whether each value is obtained through V2V communication ($\cdot^{\text{V2V}}$), Kalman Filter estimation ($\cdot^{\text{EST}}$) or radar measurement ($\cdot^{\text{RAD}}$). Additionally, let δ be the expected distance between any two vehicles, ϵ_x the sensor uncertainty associated with the variable x and $\sigma^{\text{EST}}_{d_i} = \sigma^{\text{EST}}_{p_i} + \sigma^{\text{EST}}_{p_{i+1}}$, $\sigma^{\text{EST}}_{\Delta v_i} = \sigma^{\text{EST}}_{v_i} + \sigma^{\text{EST}}_{v_{i+1}}$ the standard deviations as provided in output by the Kalman Filter along with the estimation. Finally, let $|\cdot|$ indicate the absolute value and $\bar{\cdot}$ the simple moving average of the previous n data, for some user-configured n. Then, for each vehicle i, a misbehavior is detected if any of the following inequalities is violated for more than a specified tolerance interval ΔT_{th}:

$$|d_i^{\text{EST}} - \delta| < \alpha_1 \cdot \delta \tag{2}$$
$$\left|\overline{d_i^{\text{V2V}} - d_i^{\text{EST}}}\right| < \alpha_2 \cdot 3\sigma_{d_i}^{\text{EST}} \tag{3}$$
$$\left|\overline{v_i^{\text{V2V}} - v_i^{\text{EST}}}\right| < \alpha_3 \cdot (\epsilon_v^{\text{V2V}} + 3\sigma_{v_i}^{\text{EST}}) \cdot (1 + \alpha_8|a_i|) \tag{4}$$

$$|d_i^{\text{RAD}} - \delta| < \alpha_4 \cdot \delta \tag{5}$$
$$\left|\overline{d_i^{\text{RAD}} - d_i^{\text{EST}}}\right| < \alpha_5 \cdot (\epsilon_d^{\text{RAD}} + 3\sigma_{d_i}^{\text{EST}}) \tag{6}$$
$$\left|\overline{\Delta v_i^{\text{RAD}} - \Delta v_i^{\text{V2V}}}\right| < \alpha_6 \cdot (\epsilon_{\Delta v}^{\text{RAD}} + 2\epsilon_v^{\text{V2V}}) \cdot (1 + \alpha_8|a_i|) \tag{7}$$
$$\left|\overline{\Delta v_i^{\text{RAD}} - \Delta v_i^{\text{EST}}}\right| < \alpha_7 \cdot (\epsilon_{\Delta v}^{\text{RAD}} + 3\sigma_{\Delta v_i}^{\text{EST}}) \cdot (1 + \alpha_8|a_i|) \tag{8}$$

where α_k for $k = 1 \ldots 8$ are all non-negative user-defined constants, $\alpha_1 < 1$ and $\alpha_4 < 1$.

Inequalities (2) to (4) are based only on the information received through V2V communication and estimated by means of Kalman Filters; hence, they can be evaluated independently by and for each member of the platoon. They aim to assess the coherence between the readings concerning one vehicle (4) and across consecutive vehicles (2) and (3). Instead, inequalities (5) to (8) do require additional measurements obtained through the radar and, thus, are suitable to be verified only by the follower of each vehicle. Their purpose, besides the trivial safety check provided by (5), is to compare pieces of information coming from different sources, to verify their soundness.

The left-hand side of every inequality is always composed by a difference, to measure how similar the quantities of interest are. In most cases, a moving average is leveraged to mediate between consecutive values and attenuate the oscillations caused by noisy readings: the larger the window size is selected, the smoother the average is obtained, at the cost of an increase in the detection delay. Differently, the right-hand side represents a threshold, depending on user-defined parameters and the uncertainties associated with the different measurements. Additionally, inequalities (4), (7) and (8) also encompass a correcting factor based on the current acceleration: indeed, velocity changes are expected to temporarily increase the oscillations also during normal operation.

VI. Validation

To evaluate the validity of the misbehavior detection technique herein proposed, we implemented the constraints expressed by the inequalities (2) to (8) as an extended application within Plexe. In particular, every member of the platoon continuously inspects the behavior of both the leader vehicle and its own predecessor, to verify the coherence of the information received. Whenever an injection attack is identified, being this paper focused on the detection phase, we simply switch the control algorithm to non-cooperative ACC. Although increasing the distance between consecutive vehicles, and consequently reducing the advantages of vehicle platooning, we deem this approach to be sufficiently conservative and able to prevent possible safety issues. At the same time, we leave as a future work the evaluation of more complex countermeasures.

Similarly to the analysis presented in Section IV, we experimented with injection attacks targeting both the three variables of interest (i.e. position, speed and acceleration) independently and all at the same time; in the latter case, we simulated both a non-coordinated attack, where the three variables are falsified without a particular relation between one another, and a coordinated one, respecting the kinematic equations expressed by (1). To account for a large number of distinct situations, we generated a different motion pattern for the leader vehicle in every simulation, by randomly combining acceleration and deceleration steps. Each step is characterized by a particular duration and intensity, drawn from exponential distributions with configurable means. Finally, we reproduced the uncertainties associated to the different sensors, to make the simulations more realistic. Table III summarizes the most relevant simulation parameters, concerning both the motion pattern generation, the attacker behavior and the detection algorithm. To increase the validation coverage, we executed a thousand simulations, randomly extracting most parameters from a uniform interval at the beginning of each run.

TABLE III
An Excerpt of the Main Simulation Parameters.

Simulation duration	90 s
Repetitions	1000
Controller	CACC ($\delta = 10\,\text{m}$)
Beacon frequency	10 Hz
Initial speed	uniform (90 km/h, 110 km/h)
Maximum speed	uniform (130 km/h, 150 km/h)
Acceleration (min, avg, max)	$0.1\,\text{m/s}^2$, $0.5\,\text{m/s}^2$, $2.0\,\text{m/s}^2$
Acceleration probability	uniform (0.15, 0.25)
Deceleration (min, avg, max)	$0.1\,\text{m/s}^2$, $0.75\,\text{m/s}^2$, $4.0\,\text{m/s}^2$
Deceleration probability	uniform (0.15, 0.25)
Step duration (min, avg)	0.5 s, uniform (1.5 s, 3.0 s)
Attacker	Leader vehicle
Attack start	uniform (15 s, 75 s)
Position inj. (rate, limit)	uniform (1 m/s, 5 m/s), uniform (25 m, 75 m)
Speed inj. (rate, limit)	uniform ($0.05\,\text{m/s}^2$, $0.25\,\text{m/s}^2$), uniform (1 m/s, 5 m/s)
Acceleration inj. (rate, limit)	uniform ($0.025\,\text{m/s}^3$, $0.1\,\text{m/s}^3$), uniform ($0.25\,\text{m/s}^2$, $1\,\text{m/s}^2$)
Detection parameters	$\Delta T_{\text{th}} = 1\,\text{s}$, $n = 10$, $\alpha_1 = 0.33$, $\alpha_2 = 1$, $\alpha_3 = 1$, $\alpha_4 = 0.25$, $\alpha_5 = 1$, $\alpha_6 = 1$, $\alpha_7 = 1$, $\alpha_8 = 0.05$
Sensor uncertainties	$\epsilon_p^{\text{V2V}} = 1\,\text{m}$, $\epsilon_v^{\text{V2V}} = 0.1\,\text{m/s}$, $\epsilon_a^{\text{V2V}} = 0.01\,\text{m/s}^2$, $\epsilon_d^{\text{RAD}} = 0.1\,\text{m}$, $\epsilon_{\Delta v}^{\text{RAD}} = 0.1\,\text{m/s}$

Table IV presents in an aggregated form the outcome of the simulations, highlighting the percentage of injection attacks correctly identified along with the average delay between the attack start and the actual detection instant. Overall, it is evident the effectiveness of the misbehavior detection technique herein presented: all falsification attempts have been correctly spotted within very few seconds. As expected, however, the coordinated attacks were the hardest to detect. While coherence analyses, alone, were not sufficient to recognize these more advanced injection attempts, the availability of complementary measurements coming from the radar made their identification

TABLE IV
OUTCOME OF THE SIMULATIONS: PERCENTAGE OF ATTACKS CORRECTLY IDENTIFIED AND AVG DETECTION DELAY (WITHOUT AND WITH RADAR).

Injection Attack	w/o radar		w/ radar	
	Detect. [%]	Delay [s]	Detect. [%]	Delay [s]
None	1.2	-	0.6	-
Position	99.8	1.75	100.0	1.75
Speed	99.8	2.81	100.0	2.81
Acceleration	99.8	3.80	100.0	3.79
All	99.8	1.76	100.0	1.75
Coordinated	4.4	6.00	100.0	3.90

possible. Although only the direct follower of the attacker can obtain the necessary radar measurements, simulations in Section IV pointed out that the most affected vehicle is indeed the follower itself. Thus, this protection is deemed to be completely sufficient to prevent the safety implications arisen as a consequence of the attack. Concerning the different inequalities presented, (3) and (4) accounted for the detection of almost all attacks but in the case of coordinated ones, which required the contributions of (7) and (8). Finally, after having enabled the "degrade to ACC" countermeasure, no attacks succeeded in causing a crash but in a couple of very specific cases (i.e. if the leader vehicle is hard braking while the other members switch to ACC), ascribable to the limitations of the naïve countermeasure. Overall, it is the clear demonstration of the effectiveness of the misbehavior detection technique proposed.

VII. CONCLUSIONS AND FUTURE WORK

Vehicle platooning is one of the most promising applications in the scope of Intelligent Transportation Systems and cooperation has been widely recognized as a key feature to advantage from all its benefits. However, previous research suggested CACC algorithms can be easily fooled by various cyber-attacks. As previous simulations were done with limited and rather extreme scenarios, this paper proposed an extended security analysis by means of simulation, focusing on a wider and more realistic range of injections attack scenarios. As expected, different longitudinal controllers reacted differently to various message falsifications, depending on the variables leveraged by each algorithm and their specific peculiarities. Nonetheless, the overall results confirmed the effectiveness of the attacks, as proved by the considerable number of crashes simulated between vehicles. Starting from these considerations, the second part of the paper advanced the state of the art by presenting a novel misbehavior detection technique. Its aim is to continuously evaluate the coherence of the beacons received, by comparing their content with local estimations computed by means of Kalman Filters. Additionally, whenever available, measurements coming from a different sensor (i.e. a radar) are leveraged to enrich the local knowledge and simplify the detection of attacks encompassing multiple variables. Extensive simulations confirmed the validity of the approach proposed that, combined with degradation to non-cooperative ACC as a simple countermeasure, succeeded to detect all the considered attacks and prevented catastrophic crashes. As future work, we plan to examine alternative and more sophisticated countermeasures to be adopted once an attack is identified. Additionally, we will focus on the development of an algorithm to automatically fine tune the different detection parameters depending on the specific situation, to further reduce the probability of false positives and shorten the sensing time.

REFERENCES

[1] L. D. Baskar, B. De Schutter, J. Hellendoorn, and Z. Papp, "Traffic control and intelligent vehicle highway systems: a survey," *IET Intelligent Transport Systems*, vol. 5, no. 1, pp. 38–52, Mar. 2011.

[2] Z. Wang, G. Wu, and M. J. Barth, "A review on cooperative adaptive cruise control (CACC) systems: Architectures, controls, and applications," in *Proc. 21st International Conference on Intelligent Transportation Systems (ITSC)*, Nov. 2018, pp. 2884–2891.

[3] A. A. Alam, A. Gattami, and K. H. Johansson, "An experimental study on the fuel reduction potential of heavy duty vehicle platooning," in *Proc. 13th International IEEE Conference on Intelligent Transportation Systems*, Sep. 2010, pp. 306–311.

[4] D. Swaroop and J. K. Hedrick, "String stability of interconnected systems," *IEEE Transactions on Automatic Control*, vol. 41, no. 3, pp. 349–357, Mar. 1996.

[5] R. Rajamani, *Vehicle Dynamics and Control*. Springer US, 2011.

[6] K. C. Dey *et al.*, "A review of communication, driver characteristics, and controls aspects of cooperative adaptive cruise control (CACC)," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 491–509, Feb. 2016.

[7] M. Amoozadeh *et al.*, "Security vulnerabilities of connected vehicle streams and their impact on cooperative driving," *IEEE Communications Magazine*, vol. 53, no. 6, pp. 126–132, Jun. 2015.

[8] R. van der Heijden, T. Lukaseder, and F. Kargl, "Analyzing attacks on cooperative adaptive cruise control (CACC)," in *Proc. IEEE Vehicular Networking Conference (VNC)*, Nov. 2017, pp. 45–52.

[9] C. A. Kerrache, C. T. Calafate, J. Cano, N. Lagraa, and P. Manzoni, "Trust management for vehicular networks: An adversary-oriented overview," *IEEE Access*, vol. 4, pp. 9293–9307, 2016.

[10] M. Amoozadeh, B. Ching, C.-N. Chuah, D. Ghosal, and H. M. Zhang, "VENTOS: Vehicular network open simulator with hardware-in-the-loop support," *Procedia Computer Science*, vol. 151, pp. 61–68, 2019.

[11] T. H.-J. Kim *et al.*, "VANET alert endorsement using multi-source filters," in *Proc. 7th ACM International Workshop on VehiculAr InterNETworking*, Sep. 2010, pp. 51–60.

[12] S. Ruj, M. A. Cavenaghi, Z. Huang, A. Nayak, and I. Stojmenovic, "On data-centric misbehavior detection in VANETs," in *Proc. IEEE Vehicular Technology Conference (VTC Fall)*, Sep. 2011, pp. 1–5.

[13] R. van der Heijden, A. Al-Momani, F. Kargl, and O. M. F. Abu-Sharkh, "Enhanced position verification for VANETs using subjective logic," in *Proc. IEEE 84th Vehicular Technology Conference (VTC Fall)*, Sep. 2016, pp. 1–7.

[14] S. So, P. Sharma, and J. Petit, "Integrating plausibility checks and machine learning for misbehavior detection in VANET," in *Proc. 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec. 2018, pp. 564–571.

[15] P. Lu, L. Zhang, B. B. Park, and L. Feng, "Attack-resilient sensor fusion for cooperative adaptive cruise control," in *Proc. 21st International Conference on Intelligent Transportation Systems (ITSC)*, Nov. 2018, pp. 3955–3960.

[16] M. Segata *et al.*, "PLEXE: A platooning extension for Veins," in *Proc. 6th IEEE Vehicular Networking Conference (VNC)*, Dec. 2014, pp. 53–60.

[17] S. Santini *et al.*, "A consensus-based approach for platooning with intervehicular communications and its validation in realistic scenarios," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 1985–1999, Mar. 2017.

[18] A. Ali, G. Garcia, and P. Martinet, "The flatbed platoon towing model for safe and dense platooning on highways," *IEEE Intelligent Transportation Systems Magazine*, vol. 7, no. 1, pp. 58–68, 2015.

[19] J. Ploeg, B. T. M. Scheepers, E. van Nunen, N. van de Wouw, and H. Nijmeijer, "Design and experimental evaluation of cooperative adaptive cruise control," in *Proc. 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Oct. 2011, pp. 260–265.

[20] G. Welch and G. Bishop, "An introduction to the kalman filter," University of North Carolina, Dept. of Computer Science, Tech. Rep. 95–041, 2006.