Severity of Attacks in a Vehicle Platoon by Model-Based Simulation

Cinzia Bernardeschi[1]✉[0000−0003−1604−4465], Adriano Fagiolini[2][0000−0001−9943−1975], Giuseppe Lettieri[1][0000−0003−1005−7441], Vittoria Nardone[3][0000−0001−7888−6620], Dario Pagani[1][0009−0005−4663−6666], Christian Quadri[4][0000−0002−3608−8142], and Antonella Santone[3][0000−0002−2634−4456]

[1] Department of Information Engineering, University of Pisa, 56100 Pisa, Italy
cinzia.bernardeschi@unipi.it, giuseppe.lettieri@unipi.it, dario.pagani@ing.unipi.it

[2] Department of Engineering, University of Palermo, 90133 Palermo, Italy
adriano.fagiolini@unipa.it

[3] Department of Medicine and Health Sciences "Vincenzo Tiberio", University of Molise, 86100 Campobasso, Italy
vittoria.nardone@unimol.it, antonella.santone@unimol.it

[4] Computer Science Department, University of Milano, 20133 Milan, Italy
christian.quadri@unimi.it

Abstract. This work explores model-based attack injection technique to analyse the severity of attacks to a platoon made of autonomous vehicles. Once the model of the cyber-physical system has been defined and built, we proceed to the identification of the threats that may be used to exploit the vulnerabilities of the control system of the autonomous car. Then we design driving and attack scenarios, and run several simulations in case of platoon under attack. We explore different combinations of attack parameters and driving scenarios by using the Design Space Exploration. Finally, we use the collected simulation traces to improve the knowledge on the resiliency of the system to the attacks; as well as to gauge the severity of the attacks and the possible relations between the attack parameter and the effect on the overall behavior of the system. The methodology provides quantitative evidence for risk assessment and reveals critical vulnerabilities in platoon coordination mechanisms.

Keywords: Cyber-physical systems · Model-based design · Cyber-attacks · Co-simulation.

1 Introduction

Autonomous driving systems are complex Cyber-Physical Systems (CPS) [22] that perceive surrounding environment via sensors and actuators; and combine computation and physical processes. In last years, several researches investigated about the safety and security in modern computerized cars; for instance, the paper [4] shows that remote exploitation is feasible through a broad range of

attack vectors (i.e., mechanical tools, CD players, Bluetooth and cellular radio). Various attacks have been developed and tested on a vehicle, proving that the considered vulnerabilities could actually be exploited by a potential attacker. In case of attacks through Media Player, for example, after studying the firmware of a media player of a vehicle the authors were able to create a CD containing malicious data that are then elaborated by the car, creating a dangerous scenario.

A platoon is a convoy of co-operating vehicles traveling at a short distance from one another [16]. The control of the platoon is realized by means of a control law, which is constructed to guarantee inter-vehicular distance and string stability properties. The string stability guarantees that perturbations are not propagated along the platoon [23]. The control law is fed with data read from the vehicles' sensors and provides instructions (e.g., the longitudinal acceleration) to preserve the platoon properties. Platoon control laws are designed to tolerate moderate sensor errors and communication delays. However, they cannot function properly when exposed to corrupted data, which poses significant threats to the platoon system and can lead to system failure. Based on the degree of data alteration, the unaware controller could lead the platoon to dangerous conditions like a sudden emergency braking and collisions.

In model-based development [21], simulation is one of the techniques that are usually applied, together with testing, in the analysis of systems behaviors. In the case of cyber-physical systems, simulation often takes place in the form of co-simulation [7,14], which allows sub-systems, each modeled with its most appropriate languages and tools, to be composed together. In particular, co-simulation can help developers to discover hidden issues arising from different assumptions on the models of digital controllers and those of physical components [2].

This work reports on analysis of the severity of data-alteration attacks in a vehicle platoon through model-based simulation. The model is extended with attacks that can be triggered and the effects of the attack are part of the system model. Traces of behaviour of the platoon under attack are generated under different operational scenarios. By analysing the simulation traces, attacks are subsequently classified according to the severity of consequences on the safety of the platoon. In particular, a design space exploration tool is used to vary the attack parameters within a certain domain and to generate and collect simulation traces.

The work is organized as follows: Section 2 introduces co-simulation technology and the platooning application; Section 3 describes the model of the platoon in terms of vehicle's dynamics, the control law and the communication network; Section 4 describes the attacks considered in the paper; Section 5 contains the result derived by the simulation traces; and Section 6 contains discussion and conclusions.

2 Background

This section reports basic knowledge on co-simulation technique and platooning.

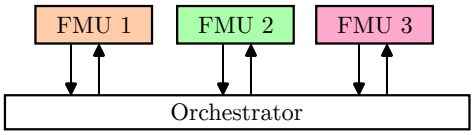

Fig. 1. Communication between the FMUs and the orchestrator.

2.1 Co-simulation

An extensive survey on co-simulation has been published by Gomes et al. in [7], providing definitions of the fundamental concepts and a taxonomy of the literature based on the discrete events and continuous time computational models. The main advantage of co-simulation is modeling flexibility, because it does not require a single modeling language for all system parts (e.g., discrete and continuous parts). The Functional Mockup Interface (FMI) [3] is an emerging standard for co-simulation of cyber-physical systems. Many modeling and simulation tools can export their models as Function Mock-up Units (FMU) that can be run under the supervision of an FMI-based orchestration engine. A scheme of co-simulation is shown in Fig. 1.

In our work we will use the INTO-CPS framework [10] for the co-simulation of the platoon. An important feature that INTO-CPS provides is the Design Space Exploration tool (DSE) [6], which is designed to solve optimization problems related to the system's parameters. The DSE tool is used for the data gathering process, required to implement the attack detection rules.

2.2 Platoon

Platooning [11] is a driving strategy where multiple vehicles travel closely together. The platoon is made of two or more cars, the one in front, called the *leader*, dictates the pace of the platoon; the other cars, called *followers*, follow behind trying to keep the same pace as the leader.

The platoon coordination is realized through network communications, which can be decentralized or centralized. In the following, we briefly present these two approaches.

Decentralized Platoon Coordination Traditionally, platooning systems rely on dedicated short-range communications (DSRC) for managing the platoon in a decentralized fashion using the IEEE 802.11p and ETSI ITS-G5 standards [18]. Another decentralized approach is the C-V2X (cellular vehicular-to-everything) standard that operates under the cellular network spectrum, allowing the vehicles to directly communicate [19].

Both decentralized approaches require that each vehicle compute the control law on board, using the data received from the other platoon members. Periodically, each vehicle broadcasts its own data, which will be received by the other members that store it on board. The decentralized approach is suitable for short

platoons and does not require any supporting network infrastructure. However, the limited communication range, the high sensitivity to the radio interferences and shadowing effects, pose serious threats to the effectiveness of inter-vehicular communications, leading to severe degradation of the overall platoon performance.

Centralized Platoon Coordination With the emergence of 5G, centralized approaches based on cellular communications and multi-access edge computing (MEC) become possible [16]. In the centralized platoon coordination approach, each vehicle sends its own status to the remote controller through the mobile network. The controller collects the status of all cars, after which the CACC control law is applied to compute the desired acceleration of each follower. The desired accelerations are then sent back to the cars.

The centralized approach supports long platoons and does not suffer from the communication limitations of the decentralized systems; moreover, the centralized platoon controller has a global and unified view of the entire platoon. Nevertheless, the network infrastructure must always be present to guarantee stable and low-latency communication. The centralized knowledge about the status of each vehicle in the platoon can be exploited to detect potential conflicting and dangerous behaviors resulting from the compromised cars.

Recently, hybrid approaches have been presented [9,19,15], combining the benefits of different radio technologies and dynamically selecting the best coordination system over time.

Our Use Case In this work we consider a centralized vehicle platoon, controlled by the Cooperative Adaptive Cruise Control (CACC) [17]. Fig. 2 shows a platoon made of 4 vehicles. In particular, each vehicle $i = 0, 1, \ldots$ is characterized by four scalar values $(u, x, v, a)_i$, these are the state variables of the vehicle. The first value $u_i,\ i = 1, 2, \ldots$ is the car's input value and represents the vehicle's desired acceleration — that is the command sent by the controller. For the leader, value u_0 is a known function part of the problem's hypothesis.The other values are the output values and represent, for each vehicle, respectively, the position x_i, the velocity v_i and the current acceleration a_i; and are sent back to the controller, completing the feedback loop.

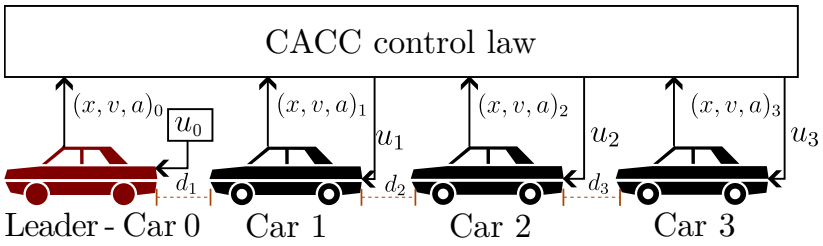

Fig. 2. Example of a platoon made of 4 cars.

Let D be the desired distance between each pair of vehicles. The acceleration command u for the i-th vehicle, with $i > 0$, is given by the formula

$$u_i = \alpha_1 a_{i-1} + \alpha_2 a_0 + \alpha_3 (v_i - v_{i-1}) + \alpha_4 (v_i - v_0) + \alpha_5 (D - d_i), \quad (1)$$

where d_i is the distance to the vehicle in front ($d_i = x_{i-1} - x_i - l$), with l the length of the car), and α_i are control gain parameters. We assume, in our case-study, $\alpha = (0.5, 0.5, -0.3, -0.1, -0.04)$, as they are the values suggested by the literature [20], to guarantee stability. This choice allows us to focus on the impact of attacks on a standard CACC configuration, while the investigation of alternative or adaptive tuning strategies is left for future work.

The desired acceleration for vehicle i, depends on the acceleration of the vehicle in front; the acceleration of the leader; the difference of velocity between the vehicle and the vehicle in front; the difference of velocity between the vehicle and the leader; the gap between the desired distance D and the distance to the vehicle in front. The overall behavior of the platoon is imposed by u_0, the acceleration command given to the leader, which is a signal dependent only on time.

3 Platoon CPS Model

We made use of three FMUs to implement the platoon model and the driving scenario:

- **Vehicle's dynamics FMU**, this FMU implements the dynamics of the vehicle; it was made in MATLAB Simulink using the Single Track Vehicle Body [12] at its core to implement the physics.
- **Driver FMU**, this FMU is a specialised version of the former that includes a driver for the platoon leader
- **Controller FMU**, this FMU implements the control law and the network medium; it was made in Python

We have an instance of the vehicle's dynamics FMU for each follower car in the platoon; and a single instance of the Driver FMU and the Controller FMU.

From now, the parameters of FMU — that is the constants of the system — will be stylized using the typewriter typeface, i.e. `start_time`.

3.1 Vehicle's Dynamics

We have one single input, the *desired acceleration*, that is a physical value, expressed in meter per squared second, that represents the desired acceleration u to be imposed on the car. Firstly, the u signal passes through a first-order filter with $\tau = 0.2$, this is done to simulate the actuation delay of the throttle of the car. The time constant τ represents how slowly the filter allows the signal to pass through.

The signal u is then used as the reference of a simple feedback control system that generates the longitudinal force F_{lon} to impose on the cars.

Finally, the physical values (x, v, a) generated by the car body, with F_{lon} fed as input signal, are used as outputs of the FMU.

3.2 Driver of the Leader

In the case of the leader car the u_0 signal is internally generated instead of being an input of the FMU. The generated signal can be tweaked via the Driver FMU's parameters.

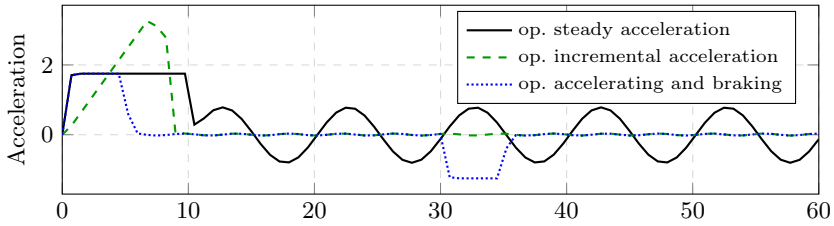

Fig. 3. Example of possible operational modes for the leader.

We consider three possible driving scenarios, the driving scenario can be chosen via the `operational_mode` parameter. The scenarios are:

- **steady acceleration**, the leader accelerates at constant acceleration for the first 10 seconds of the simulations then follows a sinusoidal signal
- **incremental acceleration**, the leader accelerates following a ramp for the first 8 seconds then switches to no acceleration
- **accelerating and braking**, the leader alternates periods of acceleration and deceleration with periods of no acceleration

Fig. 3 shows the possible driving behaviors for the leader, one for each of the three operational modes.

3.3 Controller and Network Medium

The centralized CACC FMU is implemented in Python [16], it also implements the transmission delays. There is only one instance of this FMU, reading data from all the cars and sending back the desired acceleration u_i to the follower cars. In our study, the delay is drawn from an exponential distribution with an average delay of 33 ms in both directions, uplink and downlink, leading to a total delay between the data reading and instruction reception of 66 ms, on average. These settings simulate a typical 5G-Edge scenario that relies on low-latency 5G slice implementation and a platoon controller hosted on edge servers geographically close to the platoon. As reported in [16], the chosen delay settings guarantee the platoon stability, enforcing the respect of the inter-vehicle distance policy and the string stability property.

4 Attacks

We consider data alteration attacks. The attacks are implemented by modifying the signals directly inside the vehicle FMU, in a similar manner as shown in Fig. 4, that is the original signal u is altered in a tainted one $\tilde{u}$ when the *attack enabled* condition is satisfied. In our case the attack is activated when $t \geq t_{\mathrm{A}} =$ `attack_time`.

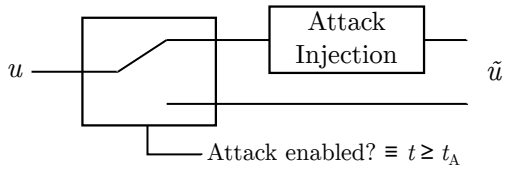

Fig. 4. How the attack is injected at runtime: u, the input signal, is altered into $\tilde{u}$.

In our scenario, the attacker targets a single vehicle within a platoon aiming to manipulate its sensory or control systems to disrupt the regular operation of the entire formation. The target vehicle can be any of the platoon's vehicles, whether a leader or a follower. The primary objective is to investigate the cascading effects of such a disruption on the whole platoon, focusing on the implications for system safety.

The attacks aim to explore vulnerabilities within the connected vehicle platoon system, where vehicles are interlinked through communication and control mechanisms. The attacker seeks to undermine vehicle coordination by attacking just one vehicle and — as the control systems are interconnected — this can result in significant disturbances in the overall driving behavior, vehicle spacing, and system stability, potentially compromising safety.

The impact of the attacks varies depending on the position of the targeted vehicle. If the vehicle under attack is the leader, the disruptions can propagate downstream, affecting all subsequent vehicles in the platoon. Conversely, if a follower vehicle is targeted, the disruption may be more localized but still influences the vehicles trailing behind.

4.1 Attacks on Actuators

In general, we should expect that an alteration to the original desired acceleration u may cause an instability of the system, as the control law still computes the correct value, but then the attacker modifies it, possibly making the feedback loop of the control system behave erroneously.

In our study, we modify the desired acceleration u, coming from the CACC, into an altered one called $\tilde{u}$; this causes the car to accelerate or brake by a different amount than the correct one. Possible attacks on the car's actuators,

modeled in this work, are the shift of the control signal by a certain constant value or the scale of the control signal by a certain factor.

We consider two possible scenarios. In the first scenario, the **Shift attack**, a constant acceleration term A is added to the desired acceleration u_i of a car i, thereby modifying the signal applied to the vehicle as

$$\tilde{u}_i = u_i + A. \tag{2}$$

In the second scenario, the **Scale attack**, the desired acceleration u_i of a car i is scaled by a factor A, resulting in a modified signal given by

$$\tilde{u}_i = (1 + A)u_i. \tag{3}$$

4.2 Attacks on Sensors

In the context of sensor attacks, the attacker can alter sensor readings by injecting a spurious signal, which may consist of N sinusoidal waves with chosen frequencies (f) and amplitudes (A). Once N, f, A are determined, the attacker's main objective is to identify the frequencies at which the spurious signal significantly impacts the system.

It is important to choose the value of the frequencies properly. A frequency too high, may lie outside the controller's bandwidth, making the attack completely ineffective; on the other hand, a frequency too low, may be easily identifiable by reasonable error-detection checks.

Of this kind of attack, we consider and implement a single scenario in which we replace the signals a, v, x with altered ones $\tilde{a}, \tilde{v}, \tilde{x}$, starting from $\tilde{a}$ and then updating in cascade the $\tilde{v}, \tilde{x}$, as described by the following:

$$\tilde{a}(t) = g(t) + a(t) \tag{4}$$

$$\tilde{v}(t) = v(0) + \int_0^t \tilde{a}(\tau)\, d\tau \tag{5}$$

$$\tilde{x}(t) = x(0) + \int_0^t \tilde{v}(\tau)\, d\tau. \tag{6}$$

In such a way, the physical values of the car are altered consistently with each other, making it difficult, for an observer, to detect that the system is under attack.

We model three attacks, as shown by the (7). The switch between the modes is done by setting the FMU's parameter `attack_mode`; A, that is the attack amplitude, expressed in $\mathrm{m} \times \mathrm{s}^{-2}$; and f_L, f_H, that are respectively the frequencies of the low frequency and high frequency signal.

$$g(t) = \begin{cases} A \sin(f_{\mathrm{L}}\, 2\pi t) & \text{if } \texttt{attack\_mode} = 1 \\ A \sin(f_{\mathrm{H}}\, 2\pi t) & \text{if } \texttt{attack\_mode} = 2 \\ A \left(\sin(f_{\mathrm{L}}\, 2\pi t) + \sin(f_{\mathrm{H}}\, 2\pi t)\right) & \text{if } \texttt{attack\_mode} = 3. \end{cases} \tag{7}$$

5 Results

In this section we present examples of single simulation traces for the previously described attacks; then we categorize, present and discuss the results obtained from the aggregation of the data from all the simulations. The raw data are available in a public repository [13].

5.1 Overview of the Configuration

The DSE was configured to vary the parameters to simulate different driving scenarios and attack conditions on a platoon composed of 9 vehicles: 1 driver and 8 followers. Each trace represents a simulation of 180s of length and a simulation step size of 10ms. The attack starts after 60 s of simulation. All vehicles start from a standstill, spaced 1 m from each other. The CACC is set to maintain a safety distance of $D = 10\text{m}$. Using the DSE tool, we gathered a grand total of 2,016 simulation traces, divided in five main batches, one without attacks, the other with different kinds of attacks.

Regarding the driving scenario, 96 possible cases were considered, as described in Sec. 3.2. In particular, we tried, for each operational mode, several possible values for the amount of acceleration and deceleration performed by the leader; and of the frequency of the acceleration, in the case of sinusoidal operation.

Regarding the attacks, we chose the value of A within the domain $|A| < 1$, more specifically $A = 0.08\text{m} \times \text{s}^{-2}$, as such attacks are not easily identifiable; on the contrary, a large value of A might immediately cause an error detector to raise an error flag.

In this work we show attacks on car 1, on car 4 and the leader. For both car 1 and 4, we consider attacks on sensors and actuators, as described in Sec. 4, starting at $t_\text{A} = 60\text{s}$; whereas for the leader we only considered attacks on the sensors, with the same parameters. We do not consider attacks on the actuator of the leader, as modifying the signal u_0 would just change the pace of the overall platoon, so it's like introducing another driving scenario. We chose car 1 and 4 because one is the car at the beginning of the platoon, the other is a car in the middle of it.

5.2 Actuator Attacks

Let us consider an attack on car 1, that is, the control signal u_1 is altered.

Shift Attack Let us consider the inter-vehicular distance between car 1 and the leader, and the inter-vehicular distance between car 2 and car 1; noted as d_1 and d_2 respectively. The distance d_i is computed as $d_i = x_{i-1} - x_i - 4\text{m}$, assuming a vehicle length of 4m.

In Fig. 5 we plot the value of the distances in the nominal case — that is in absence of attacks — and in case of attack given by (2), considering the case of a positive and a negative value of $A = \pm 0.08$.

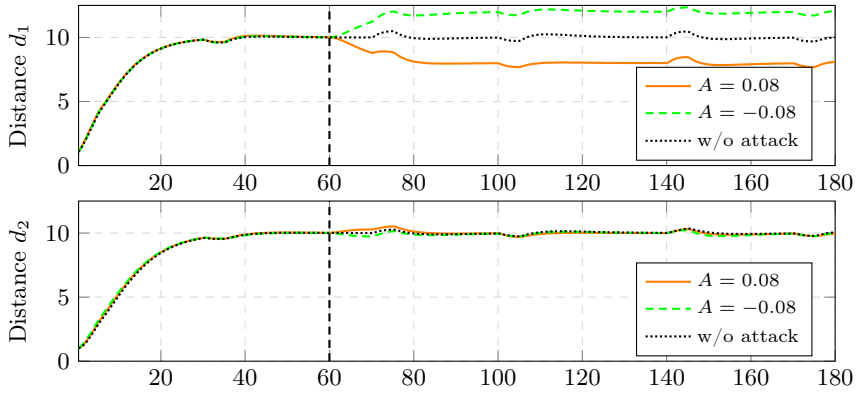

Fig. 5. Shift attack, distance d_1 (b/w leader and car 1) and d_2 (b/w car 1 and 2) plotted over time.

We can observe how such an attack affects only the distance d_1 (top plot), between the leader and the car 1. In particular, a value of $A = 0.08$ definitively reduces the distance to the vehicle in front by around two meters; whereas a value of $A = -0.08$ definitively increases the distance by around two meters. The distance d_2 (bottom plot), between car 1 and 2, is not altered by the attack. We didn't report the value of d_3 as it follows a similar behavior as d_2.

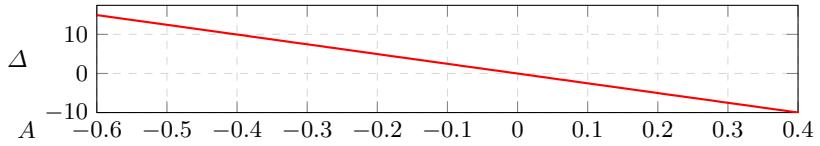

Fig. 6. Δ wrt nominal distance at $t = 120$s over A (the attack parameter).

Let us now, given a certain A, consider the error $\Delta = d_1 - D$ ($D = 10$m) made when car 1 is attacked as in (2). The results are shown in Fig. 6. We initially run the simulation within the domain $A \in [-0.6, 0.6]$ discretized with step 0.01. We exclude from further analysis the interval $[-0.6, 0.4)$ as car 1 ends up rear-ending the leader. Overall, we can see how negative values of A are non fatal, as the gap between vehicles increases, thus causing only an efficiency loss.

As result, we observe that there is a linear relation between A and Δ. After a linear regression, we can show that for the considered domain the relation between A and Δ can be expressed, with $R^2 = 0.996$, as $\Delta \simeq -25.41 \cdot A - 0.06$.

Scale Factor Attack In Fig. 7 we plot d_1 with the attack described by (3). The figure shows that the gap between car 1 and the leader only scales by a certain

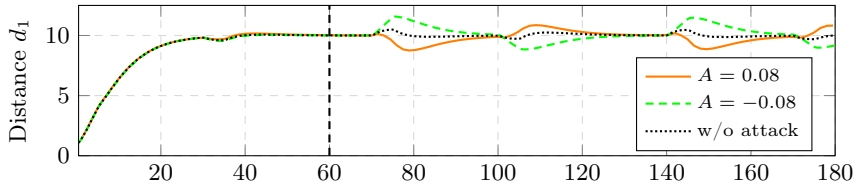

Fig. 7. Scale attack, distance d_1 plotted over time.

factor without impacting the platoon's safety. This shows us that the control law is able to operate in under this kind of attack, although its performance is degraded. For brevity's sake, we omitted d_2, d_3 as they don't show any differences compared to the baseline.

5.3 Sensors Attack

Let us now consider two sample cases of attacks, as formulated by the (4), in which the physical values (x, v, a) are altered. We ran the simulation once with no attack and then two times with $A = \pm 0.08$, starting at $t_A = 60$s and always on car 1.

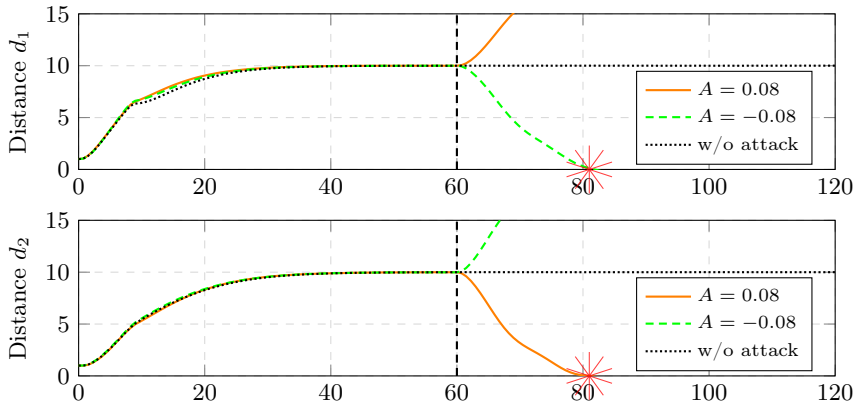

Fig. 8. Sensor, distance d_1 and d_2 between car 2 and 1 plotted over time. The red star denotes the crash between the two vehicles.

In Fig. 8 the results of the three simulations are shown. We can see how in both cases we have a crash at around $t \simeq 80$s. With $A > 0$, we see car 2 rear-ending car 1, as $d_2 \to 0$; whereas with $A < 0$, we see car 1 rear-ending the leader, as $d_1 \to 0$. It can be observed that the behavior persists for any value of A.

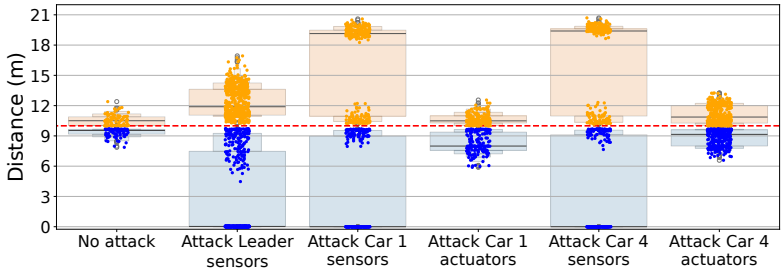

Fig. 9. Distribution of the min/max of all the following distances across all scenarios.

5.4 Severity of Attacks

Data Distribution Let us study the distribution of the minimum and maximum distance d across all simulation traces. For the maxima, for each simulation trace we consider the

$$M = \max_{i=0,\dots,8} \max_{t} d_i(t)$$

and, similarly, the minima are computed using the min operator.

The results are shown in Fig. 9, the red dotted line is the desired distance $D = 10\text{m}$, the orange points are the maxima and the blue ones the minima.

We can see how, without any attack, the value concentrates around the desired distance $D = 10\text{m} \pm 2$ and it does not go below 8 m. In the case of attacks on actuators, as shown in the previous section, with the chosen value of A, there's only a definitive and constant change in inter-vehicular distance but no collisions. As already observed, for the chosen parameters, we have collisions, that is $d = 0$, in case of attacks on sensors. When the leader is under attack (on its sensors), the effect on the distances are less severe, as the data points are more concentrated near the D line; whereas attack on car 1 and 4 generate larger deviations from the baseline.

Trace Labeling We defined three class of severity, in ascending order: *OK*, *TOO CLOSE* and *COLLISION*. Given a simulation step t and a car i, we can define a label $L(t, i)$ whose value is *COLLISION*, in case of collision, *TOO CLOSE* if $d_i(t) \leq 8$, chosen by studying the plot in Fig. 9, as in the case of no attack the distances never go below such a threshold. Otherwise, label *OK* is used.

The 8-meter threshold for the *TOO CLOSE* classification is determined by the operational tolerance of the platoon. The desired inter-vehicle distance is set to $D = 10\text{m}$. Under normal operation, the CACC controller maintains distances within $\pm 20\%$ of this target, as shown by Fig. 9. The 8-meter threshold represents a deviation from the desired distance. At this distance, the system is no longer operating as designed. The objective of platooning is to maintain close inter-vehicular distances at high speeds, making deviations beyond the control tolerance critical safety indicators.

Then, we define a global label for the entire simulation trace as

$$M_{\text{trace}} = \max_{i=0,\ldots,8} \max_{t} L(t,i). \tag{8}$$

Then we aggregate the resulting labels of simulation traces by class, computing the cardinality of each class, the final figures, shown in percentages, are reported in Table 1.

Table 1. Aggregated results from the simulation traces.

Label class	**No attack**	**Leader sensors**	**Car 1 sensors**	**Car 1 actuators**	**Car 4 sensors**	**Car 4 actuators**
OK	100.00%	33.33%	33.33%	50.00%	33.33%	75.00%
TOO CLOSE	0.00%	0.00%	0.00%	50.00%	0.00%	25.00%
COLLISION	0.00%	66.67%	66.67%	0.00%	66.67%	0.00%
Tot. traces	**96**	**576**	**288**	**384**	**288**	**384**

Our labeling might be used as an aid to provide an estimate of the *impact rating*, in the framework of Threat Analysis and Risk Assessment (TARA) activity in the field of cybersecurity engineering in automotive [8].

6 Conclusions

This work has presented a comprehensive model-based approach for analyzing the severity of cyber-attacks in vehicle platooning systems. Through the development of high-fidelity co-simulation models incorporating vehicle dynamics, control laws, and communication networks, we have demonstrated the effectiveness of simulation-based techniques in assessing the impact of various attack scenarios on platoon safety.

Our results reveal several critical insights into attack severity: actuator attacks, particularly the shift attack, show a predictable linear relationship between attack magnitude and inter-vehicle distance deviation; sensor data alteration attacks prove particularly dangerous, consistently leading to collisions in approximately 2/3 of simulated scenarios regardless of the attacked vehicle's position; and scale factor attacks on actuators, while affecting vehicle spacing, demonstrated lower immediate safety risks compared to other attack types. These results suggest possible detection approaches that might be installed on the network edge and/or the vehicles and will be the topic of future work.

The shown method has the advantage of being modular and a large domain of combinations of parameters and attack scenarios can be explored using the DSE tool. Such an approach might be useful in the design and early prototyping to gauge and assess the risk and the safety countermeasures to adopt in the system to limit the exploitable attack vectors.

Future work will study the resilience of the platoon to the attacks with different CACC control parameters. It will also further examine adverse network

conditions. This could involve longer simulations in which the platoon traverses road segments under specific environmental conditions known to degrade network performance. In addition, more complex scenario will be considered, for instance, platoons in which multiple cars are attacked simultaneously with a heterogeneous set of attacks. Moreover, future work will also involve validation using higher-fidelity simulators like CARLA [5] and BeamNG.tech, [1] which offer more advanced models of sensor data and vehicle dynamics.

Acknowledgments. This work has been supported by the European Union - Next - Generation EU - National Recovery and Resilience Plan (NRRP) - MISSION 4 COMPONENT 2, INVESTMENT N. 1.1, CALL PRIN 2022 PNRR D.D. 1409 14-09- 2022 - (FORESEEN) CUP N.P2022WYAEW and by the Italian Ministry of University and Research (MUR) in the framework of the Crosslab and FoReLab projects (Departments of Excellence). This work has also been supported by the project SERICS (PE00000014) under the MUR NRRP funded by the European Union NextGenerationEU – CUP: G43C22002580001.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. BeamNG GmbH: BeamNG.tech, https://www.beamng.tech/
2. Bernardeschi, C., Domenici, A., Fagiolini, A., Palmieri, M.: Co-simulation and formal verification of co-operative drone control with logic-based specifications. The Computer Journal **66**(2), 295–317 (10 2021). https://doi.org/10.1093/comjnl/bxab161
3. Blochwitz, T., Otter, M., Åkesson, J., Arnold, M., Clauß, C., Elmqvist, H., Friedrich, M., Junghanns, A., Mauß, J., Neumerkel, D., Olsson, H., Viel, A.: Functional Mockup Interface 2.0: The Standard for Tool independent Exchange of Simulation Models. In: Proc. of the 9th Intl. Modelica Conference. pp. 173–184. The Modelica Association (2012), http://dx.doi.org/10.3384/ecp12076173
4. Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., Koscher, K., Czeskis, A., Roesner, F., Kohno, T.: Comprehensive experimental analyses of automotive attack surfaces. In: SEC'11: Proc. of the 20th USENIX conference on Security. USENIX Association (2011)
5. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: An open urban driving simulator. In: Proceedings of the 1st Annual Conference on Robot Learning. pp. 1–16 (2017)
6. Gamble, C.: DSE in the INTO-CPS Platform. Tech. Rep. D5.3e, INTO-CPS Deliverable (2017)
7. Gomes, C., Thule, C., Broman, D., Larsen, P.G., Vangheluwe, H.: Co-simulation: A survey. ACM Comput. Surv. **51**(3) (may 2018). https://doi.org/10.1145/3179993
8. ISO/SAE: Road vehicles — Cybersecurity engineering. Standard ISO/SAE 21434:2021, International Organization for Standardization and SAE International (2021), https://www.iso.org/standard/70918.html
9. Jacob, R., Anwar, W., Fettweis, G., Pohlmann, J.: Exploiting Multi-RAT Diversity in Vehicular Ad-Hoc Networks to Improve Reliability of Cooperative Automated Driving Applications. In: IEEE VTC2019-Fall. pp. 1–7 (2019). https://doi.org/10.1109/VTCFall.2019.8891072

10. Larsen, P.G., Fitzgerald, J., Woodcock, J., Fritzson, P., Brauer, J., Kleijn, C., Lecomte, T., Pfeil, M., Green, O., Basagiannis, S., Sadovykh, A.: Integrated tool chain for model-based design of cyber-physical systems: The into-cps project. In: 2nd International Workshop on Modelling, Analysis, and Control of Complex CPS (CPS Data). pp. 1–6 (2016). https://doi.org/10.1109/CPSData.2016.7496424
11. Maiti, S., Winter, S., Kulik, L.: A conceptualization of vehicle platoons and platoon operations. Transportation Research Part C: Emerging Technologies **80**, 1–19 (2017)
12. MathWorks: Adaptive Cruise Control System Using Model Predictive Control. The MathWorks, Inc., https://it.mathworks.com/help/mpc/ug/adaptive-cruise-control-using-model-predictive-controller.html, mATLAB Documentation
13. Pagani, D., Bernardeschi, C., Lettieri, G., Quadri, C., Fagiolini, A., Santone, A., Nardone, V., Vivani, A.: Longitudinal traces of platoon simulation with and without cyber-attacks and network latency (Jul 2025). https://doi.org/10.5281/zenodo.16529953
14. Palmieri, M., Quadri, C., Fagiolini, A., Bernardeschi, C.: Co-simulated digital twin on the network edge: A vehicle platoon. Computer Communications **212**, 35–47 (2023). https://doi.org/10.1016/j.comcom.2023.09.019
15. Quadri, C., Dileo, M., Mancuso, V., Marsan, M.A.: DNN-controlled multi-technology platooning. In: 2025 IEEE Vehicular Networking Conference (VNC). pp. 1–8 (2025). https://doi.org/10.1109/VNC64509.2025.11054110
16. Quadri, C., Mancuso, V., Marsan, M.A., Rossi, G.P.: Edge-based platoon control. Computer Communications **181**, 17–31 (2022). https://doi.org/10.1016/j.comcom.2021.09.021
17. Rajamani, R., Tan, H.S., Law, B.K., Zhang, W.B.: Demonstration of integrated longitudinal and lateral control for the operation of automated vehicles in platoons. IEEE Transactions on Control Systems Technology **8**(4), 695–708 (2000). https://doi.org/10.1109/87.852914
18. Santini, S., Salvi, A., Valente, A.S., Pescapé, A., Segata, M., Lo Cigno, R.: A consensus-based approach for platooning with intervehicular communications and its validation in realistic scenarios. IEEE Transactions on Vehicular Technology **66**(3), 1985–1999 (2017). https://doi.org/10.1109/TVT.2016.2585018
19. Segata, M., Cigno, R.L., Hardes, T., Heinovski, J., Schettler, M., Bloessl, B., Sommer, C., Dressler, F.: Multi-technology cooperative driving: An analysis based on plexe. IEEE Transactions on Mobile Computing **22**(8), 4792–4806 (2023). https://doi.org/10.1109/TMC.2022.3154643
20. Segata, M., Joerer, S., Bloessl, B., Sommer, C., Dressler, F., Cigno, R.L.: Plexe: A platooning extension for veins. In: 2014 IEEE Vehicular Networking Conference (VNC). pp. 53–60 (2014). https://doi.org/10.1109/VNC.2014.7013309
21. Selic, B.: The pragmatics of model-driven development. IEEE Software **20**(5), 19–25 (Sept 2003). https://doi.org/10.1109/MS.2003.1231146
22. Shi, J., Wan, J., Yan, H., Suo, H.: A survey of cyber-physical systems. In: 2011 International Conference on Wireless Communications and Signal Processing (WCSP). pp. 1–6 (2011). https://doi.org/10.1109/WCSP.2011.6096958
23. Swaroop, D., Hedrick, J.: String stability of interconnected systems. In: Proceedings of 1995 American Control Conference - ACC'95. vol. 3, pp. 1806–1810 vol.3 (1995). https://doi.org/10.1109/ACC.1995.531196