CARACAS: vehiCular ArchitectuRe for detAiled Can Attacks Simulation

# CARACAS: vehiCular ArchitectuRe for detAiled Can Attacks Simulation

Sadek Misto Kirdi[1], Nicola Scarano[1], Franco Oberti[2], Luca Mannella[1], Stefano Di Carlo[1], Alessandro Savino[1]*
[1]*Politecnico di Torino, Department of Control and Computer Engineering, Turin, Italy*
[2]*Dumarey Softronix S.r.l., Torino*
*e-mail:* [1]{*name.surname*}*@polito.it, sadek.mistokirdi@studenti.polito.it,*[2]*franco.oberti@dumarey.com*

***Abstract*—Modern vehicles are increasingly vulnerable to attacks that exploit network infrastructures, particularly the Controller Area Network (CAN) networks. To effectively counter such threats using contemporary tools like Intrusion Detection Systems (IDSs) based on data analysis and classification, large datasets of CAN messages become imperative.**

**This paper delves into the feasibility of generating synthetic datasets by harnessing the modeling capabilities of simulation frameworks such as Simulink coupled with a robust representation of attack models to present CARACAS, a vehicular model, including component control via CAN messages and attack injection capabilities. CARACAS showcases the efficacy of this methodology, including a Battery Electric Vehicle (BEV) model, and focuses on attacks targeting torque control in two distinct scenarios.**

***Index Terms*—Automotive Security, Battery Electric Vehicles, CAN attacks, Cybersecurity, Vehicular Systems**

## I. Introduction

The automotive industry is undergoing a substantial shift in its primary drivers of innovation, moving from combustion engines to Battery Electric Vehicles (BEVs). This transition into a relatively unexplored domain of vehicular systems introduces unprecedented risks that car manufacturers have not previously encountered [1]. Among them, cyberattacks are becoming more frequent, posing significant threats to both the systems' security and the drivers' safety [2].

The Controller Area Network (CAN) [3], which serves as the communication backbone among various components within traditional combustion engine vehicles, continues to be the primary communication technology in future cars. It is essential for enabling communication among multiple Electric Control Units (ECUs), sensors, and intelligent actuators. This facilitates more straightforward vehicle wiring and sensor integration. However, the CAN network's external accessibility provides attackers with multiple exploitation opportunities [4]–[6]. Furthermore, emerging functions such as Adaptive Cruise Control (ACC) and Cooperative ACC (CACC) systems, which are enabled by Vehicle-to-Vehicle (V2V) communication, have significantly enhanced vehicle efficiency and safety. Conversely, these systems expand the vehicle's attack surface and introduce new security vulnerabilities. Once an adversary gains access to a vehicle's CAN, they can manipulate the system by injecting malicious packets. For instance, they could shut off the engine [7] or turn off the immobilizer through a replay attack—a method currently popular for stealing luxury Sport Utility Vehicles (SUVs) [8]. These vulnerabilities require urgent attention and mitigation [9].

Intrusion Detection Systems (IDSs) are countermeasures of great interest due to their simplicity and efficiency in detecting attacks, even on resource-constrained environments [10]–[12]. They monitor network or host activities, raise alarms when unexpected events occur, and help prevent unauthorized access to the system. These techniques are being used extensively in Automotive, and multiple studies on the application of IDS on CAN attacks have emerged, showing interesting results [13]–[15]. Research in CAN IDS has expanded quickly but needs help with reproducing, replicating, and comparing methodologies due to the need for costly infrastructure and extensive expertise [16]. Consequently, due to their unavailability, many proposed detection techniques still need to be tested on suitable data [17].

The paper presents two main contributions. Firstly, it introduces a novel modular approach integrating a vehicular dynamics simulation model in a Simulink environment alongside a CAN bus model and a CAN injection model. It leverages the Simscape [18] framework, a platform for building and simulating physical systems within Simulink [19] to simulate CAN bus attacks. This integration streamlines the analysis and validation of simulated CAN attacks on the modeled vehicle, facilitating the generation of synthetic CAN traffic and simulated attack data, allowing researchers to directly observe the effects of cyber threats on the vehicle's operational integrity and safety. Secondly, the paper demonstrates the capability of generating regular CAN messages alongside malicious CAN ones in a Simulink model for a BEVs vehicle based on the Tesla Model 3. These models undergo testing across two driving scenarios to assess their performance when exposed to CAN bus attacks, i.e., Extra Urban Driving Cycle and Cruise Mode. The final generated dataset contains regular CAN traffic and malicious torque CAN messages.

This study was carried out within the SERICS - Security and Rights in the CyberSpace and received funding from the European Union NextGenerationEU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.3 – D.D. 1556 11/10/2022, PE00000014), and within the COLTRANE-V project – funded by the Ministero dell'Università e della Ricerca – within the PRIN 2022 program (D.D.104 - 02/02/2022). This manuscript reflects only the authors' views and opinions. Neither the European Union, nor the European Commission, nor the Ministry can be considered responsible for them.
(*) Correspondence: alessandro.savino@polito.it

The remaining paper includes Section II, which analyzes the state of the art in modeling and dataset generation of CAN attacks. In contrast, Section III describes the modeling approach. Eventually, Section IV presents the simulated attacks results, and Section V sets the conclusion and future work perspectives.

## II. Related Works

The rapid expansion of research in CAN IDSs often encounters impediments due to challenges in replicating and comparing methodologies, primarily caused by difficulties obtaining valuable data [20]. As a result, many detection techniques proposed still need to be validated due to the constrained availability of appropriate datasets [17]. Over time, numerous datasets have emerged to alleviate data scarcity concerns. These datasets encompass real-world scenarios involving vehicles subjected to network attacks. For instance, Hacking and Countermeasure Research Lab (HCRL) has released three datasets, each varying in data types, vehicles, and attack scenarios [10], [15], [16]. Additionally, the Laboratory of Cryptography and System Security (CrySyS Lab) team at Budapest University of Technology and Economics has published the CrySyS dataset comprising CAN traffic logs [21], while Eindhoven University of Technology has contributed datasets that combine real CAN data with simulated attack injections [22].

However, these real-world datasets show significant challenges during the data collection procedure, which is time-consuming and expensive, thus limiting the number of vehicles and the range of attacks used. A potential remedy for these limitations is the creation of simulated environments to generate artificial CAN signals and simulate attacks within. For instance, Hanselmann et al. from Bosch GmbH, in [23], developed a synthetic CAN dataset to train and test their Long Short-Term Memory (LSTM)-based anomaly detector, "CANet". Despite its utility, the main drawback with this and other synthetic datasets (like the synthetic data from TU Eindhoven [22]) is the lack of verification of their effects on actual vehicles [17], posing doubts about the consequences of these attacks on vehicle systems.

The framework presented in this paper is designed to generate CAN vehicle traffic under normal operating conditions by emulating a closed-loop model that adjusts based on critical system parameters. This framework facilitates the easy integration of various car models, enhancing its applicability across different automotive technologies. Furthermore, our CAN Injector module is designed to emulate a broad spectrum of external hacking tools used by attackers [24]. This enhancement significantly increases the adaptability and robustness of our testing process, thereby making it more effective in identifying vulnerabilities.

Ultimately, the capabilities of the injector are crucial, particularly in determining the severity of attacks and their correlation with the system's anticipated behavior on various road conditions.

## III. Proposed Approach

The proposed system is composed of three interconnected modules designed using Simulink [19]: (i) a BEV Dynamic model, (ii) a CAN bus model, and (iii) a CAN Attack Injection simulator.

### A. BEV Dynamic model

The BEVs behaviors on driving scenarios are supported by a Simscape [18] model built within the Simulink environment. The BEV model, shown in Figure 1, integrates the essential components of a vehicle system, such as the motor, chassis, battery, and control algorithms. These interconnected components were designed to emulate not only the core dynamics of an electric vehicle but also CAN message transmissions to allow the assessment of these CAN attacks' impacts.

The chassis serves as the vehicle's structural backbone, incorporating vital features like physical attributes and weight distribution, which are crucial for accurately simulating vehicle dynamics. A simplified representation of the battery was employed to power the motor and showcase the dynamics of the battery charge, sidestepping the complexities typical in real-world battery management systems. The motor was modeled with a focus on its performance characteristics and how it interacts with the rest of the vehicle's powertrain. A vital parameter of the motor (used as the target of the simulated attacks in this work) is torque, which represents the amount of rotational force the motor develops. It is responsible for the acceleration and breaking of a vehicle.

This combination of components within the simulation framework supports validating vehicle behaviors under various operational conditions. It enhances the ability to adjust and scale the model for future, more complex scenarios.

### B. CAN bus model integration

The CANdb++ Editor is a software tool from Vector [25] essential in automotive and industrial applications for designing, editing, and managing CAN communication protocol databases. Its capacity to precisely define message frames and signal layouts makes it invaluable for creating controlled synthetic attack scenarios on the CAN network. Using the CANdb++ Editor, users can rigorously set up and modify a database, which is necessary for simulating specific network behaviors, including malicious attacks. This tool enables crafting unique messages and signal configurations that mimic potential security threats, thus providing a realistic environment for testing the robustness of BEV against CAN injections and other related cyber-attacks.

By configuring the specific attributes of the CAN messages and defining how these messages interact with various ECUs in the network, engineers can simulate and monitor how these synthetic attacks would affect the vehicle's operation. This includes observing changes in vehicle parameters such as torque, speed, and state of charge under attack scenarios.

We set the CANdb++ Editor with the parameter needed for our simulation and generated the CAN DataBase (DBC) file. Several Simulink Blocks contribute to integrating the CAN

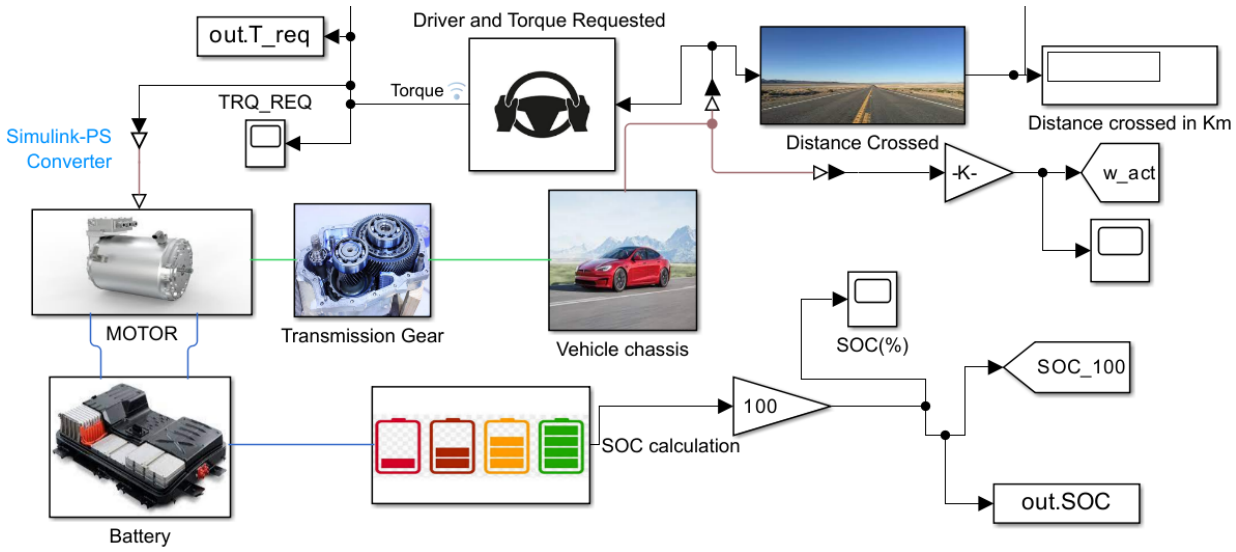


Fig. 1: Full BEV Simulink model.

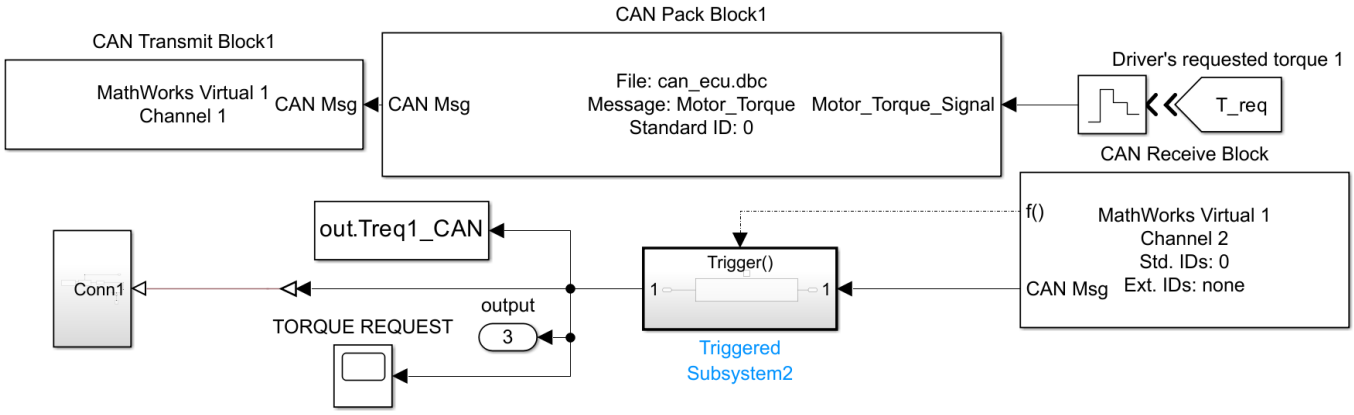


Fig. 2: Torque request schema through CAN.

into the model. The designed system and the interactions among the blocks are graphically represented in Figure 2. The CAN Configuration block allows selecting the device, channel, and bus speed. Once the signals are configured, a Simulink Zero-Order-Hold block converts continuous signals to discrete ones suitable for CAN communication. Following this, the CAN Pack block encapsulates these signals into CAN messages, which are subsequently transmitted over the network using the CAN Transmit block. The CAN Receive block captures these messages and delivers them to the Simulink vehicle model through the Trigger Block. In the latter, a CAN Unpack block converts the CAN data into signals for the Simulink model.

### *C. CAN Attack Injection simulator*

To conduct the simulation of a CAN injection attack, we utilized the Simulink signal builder block (Figure 3). The signal builder enabled us to construct various attack signals to replicate malicious scenarios and observe their impact on the vehicle's functionalities. Eventually, the signal builder provides a fast way to craft the related CAN messages that perform the attack.

Figure 3 shows how the attack, thanks to a summing block, can be easily programmed separately, leaving the original model unaltered. Such modularity also allows for the setting up of different attack models within the same experimental setup. Moreover, the trigger-based activation of the attack simplifies the data collection of benign and malicious messages in the same data generation campaign. As the manipulation is at the signal level, it is easy for the user to manipulate the frequency and amplitude of the signals to vary the attack's modality instead of crafting CAN messages directly.

The *attack scenarios* simulated in this work involve a torque attack on the CAN bus during the Extra Urban Driving Cycle and while in Cruise Mode. This attack injects malicious torque signals into the network, altering the average motor torque and consequently affecting the vehicle's speed.

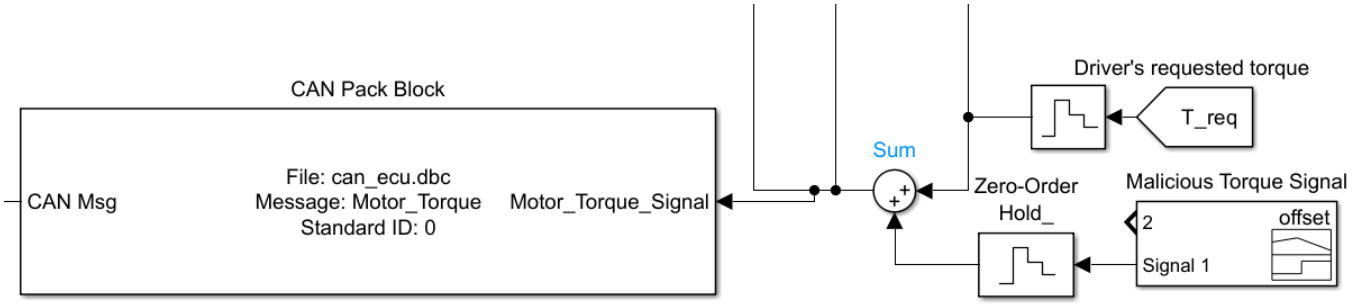


Fig. 3: Torque attack injector integration into the CAN model.

## IV. Results

To assess the potentiality of the modeling approach, the evaluation involved simulating a CAN attack on a Tesla Model 3 during an Extra Urban Driving Cycle (with varying velocity and torque values) and Cruise Mode at constant velocity. In both scenarios, the injection of a braking torque signal of $-15nm$ is strategically timed to occur between $t = 160s$ and $t = 240s$ (Figure 4).

The aim is to simulate a physical attack, then check if the model adheres to the realistic consequences, demonstrating the correct modeling, and if the CAN messages follow, proving that a valid dataset can be generated.

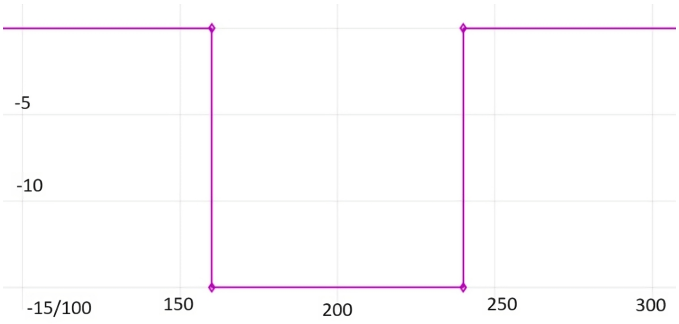


Fig. 4: Step signal injected to simulate a braking torque. Simulation time in seconds [s] on the x-axis and torque in Newton meters [Nm] on the y-axis.

### A. Extra Urban Driving Cycle

Figure 5 highlights the comparative effects of the torque with and without the attack. Before $t = 160s$, both the not attacked (shown in blue) and attacked (shown in red) torque signals displayed identical values, as no attack was implemented during this initial phase. However, at $t = 160s$, a discernible divergence occurs between the two signals. The regular torque signal, i.e., the one which is not attacked, remains stable at approximately $15Nm$. In contrast, when the attack started, the torque signal visibly dropped by an equivalent magnitude of $15Nm$ below the baseline, effectively mirroring the parameters of the engineered attack signal. This shift clearly illustrates the direct impact of the CAN intrusion, manifesting as an altered torque output that deviates from the expected performance.

Similar to the torque scenarios, both velocity signals followed the same pattern until around $t = 160s$, when the attack was executed (Figure 6). Instead of maintaining a steady velocity, the vehicle began to decelerate at this juncture, failing to adhere to the prescribed velocity trajectory. This behavior was a direct result of the imposed braking torque, which co-occurred as the vehicle attempted to accelerate according to the trajectory plan.

### B. Cruise Mode

The second use case simulates an attack while the vehicle was on cruise mode at 100 km/h, with the attack signal initiated at $t = 160s$ and continued until $t = 240s$. This can be seen in the torque graph comparison (Figure 7 ), where till $t = 160s$, no attack was active, and both the non-attacked torque (represented in blue) and the attacked torque (represented in yellow) displayed identical values. At $t = 160s$, the non-attacked torque remained stable at approximately $11Nm$; however, upon application of the attack, the torque dropped by $15Nm$, the same magnitude as the signal created, shifting the graph downwards.

The impact of this modification is further visible in the velocity plot shown in Figure 8. Here, both signals showed the same trend until around $t = 160s$ when the attack occurred. The car that is maintaining a constant velocity of $100Km/h$ at $t = 160s$ suddenly starts to decelerate undesirably until $t = 240s$ and loses its trajectory of the cruise mode. As soon as the attack was stopped, the vehicle began accelerating again, trying to reach the target velocity.

## V. Conclusion

This paper presents a systematic approach to simulate and analyze cybersecurity attacks, mainly focusing on CAN injection attacks targeting BEV. Our research addresses the rising concerns regarding the security and safety of vehicular systems in light of evolving cyber threats.

A novel simulation framework capable of generating synthetic CAN traffic and simulating various attack scenarios enables the emulation of CAN injection attacks on BEV, allowing direct observation of their effects on vehicle operational integrity and safety.

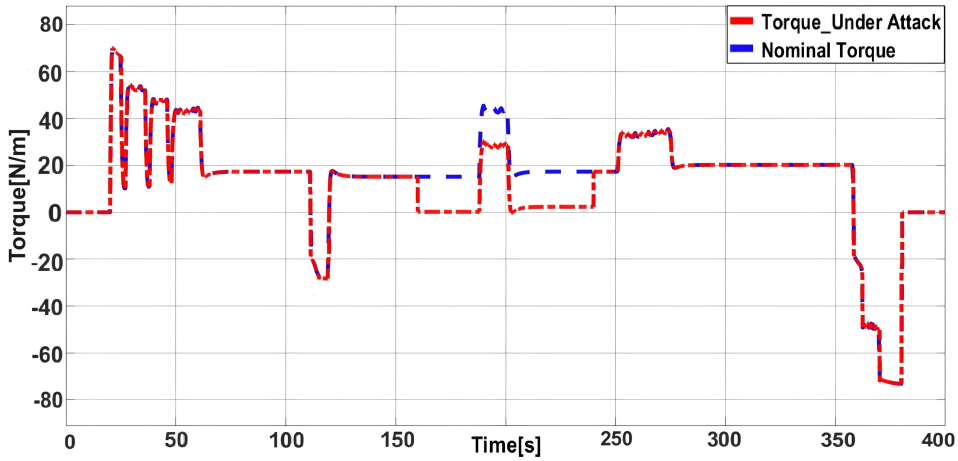


Fig. 5: Torque without any attack vs Torque with the attack while the vehicle is in Extra Urban Driving Cycle.

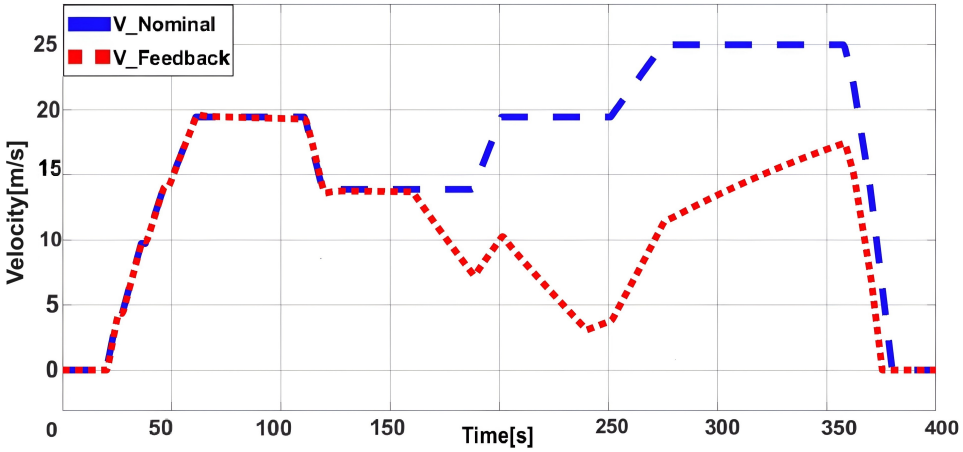


Fig. 6: Velocity trajectory during torque attack and in regular driving condition while the vehicle is in Extra Urban Driving Cycle.

Future endeavors should focus on refining simulation models, exploring additional attack scenarios, and assessing the effectiveness of diverse cybersecurity measures in mitigating CAN attacks. To encourage research in this field, we release the code related to our experiments as open-source: https://github.com/smilies-polito/CARACAS

## References


[1] S. Kim and R. Shrestha, *Automotive Cyber Security: Introduction, Challenges, and Standardization*. Springer Nature Singapore, 2021.

[2] Upstream, "Global automotive cybersecurity report." upstream.auto. https://upstream.auto/reports/global-automotive-cybersecurity-report/ (accessed May 20, 2024).

[3] C. P. Szydlowski, "Can specification 2.0: Protocol and implementations 921603," in *SAE Technical Paper 921603*, 1992.

[4] O. Avatefipour and H. Malik, "State-of-the-art survey on in-vehicle network communication c̈an-bussëcurity and vulnerabilities," *International Journal of Computer Science and Network*, vol. 6, pp. 720–727, 12 2017.

[5] F. Oberti, E. Sanchez, A. Savino, F. Parisi, and S. di Carlo, "Taurum p2t: Advanced secure can-fd architecture for road vehicle," in *2021 IEEE 27th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, pp. 1–7, 2021.

[6] F. Oberti, A. Savino, E. Sanchez, P. Casasso, F. Parisi, and S. D. Carlo, "CAN-MM: Multiplexed Message Authentication Code for Controller Area Network Message Authentication in Road Vehicles," *IEEE Transactions on Vehicular Technology*, pp. 1–13, 2024.

[7] Z. Bi, G. Xu, C. Wang, G. Xu, and S. Zhang, "A method for translating automotive body-related can messages based on labeled bits," *Applied Sciences*, vol. 13, no. 3, 2023.

[8] Stelvio Forum, "Peace of mind car hacking relay attack," 2023.

[9] H. J. Jo and W. Choi, "A survey of attacks on controller area networks and corresponding countermeasures," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6123–6141, 2022.

[10] M. L. Han, B. I. Kwak, and H. K. Kim, "Anomaly intrusion detection method for vehicular networks based on survival analysis," *Vehicular Communications*, vol. 14, pp. 52–63, 2018.

[11] D. Canavese, L. Mannella, L. Regano, and C. Basile, "Security at the Edge for Resource-Limited IoT Devices," *Sensors*, vol. 24, no. 2, 2024.

[12] C. P. Chenet, A. Savino, and S. Di Carlo, "A survey on hardware-based malware detection approaches," *IEEE Access*, vol. 12, pp. 54115–54128, 2024.

[13] H. M. Song, H. R. Kim, and H. K. Kim, "Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network," in *2016 International Conference on Information Networking (ICOIN)*, pp. 63–68, 2016.

[14] S.-F. Lokman, A. T. Othman, and M.-H. Abu-Bakar, "Intrusion detection system for automotive Controller Area Network (CAN) bus system: a re-

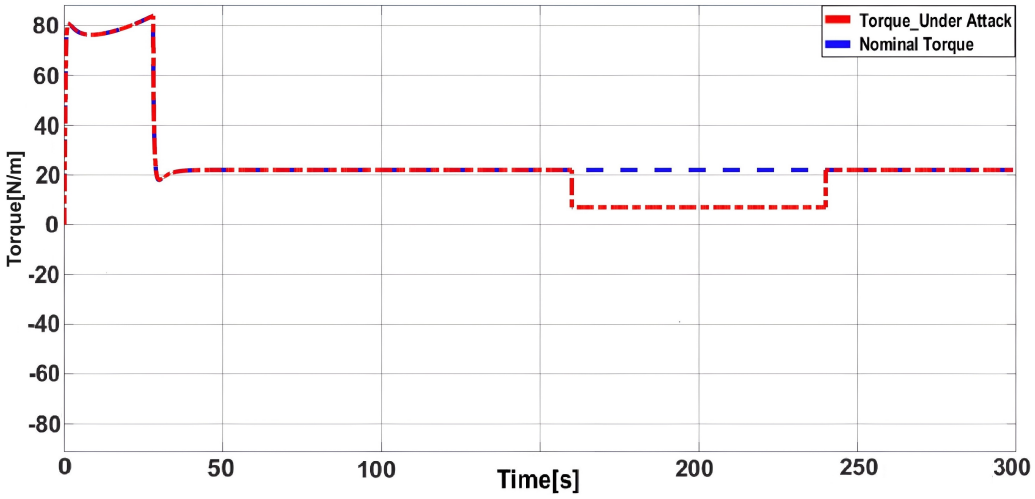


Fig. 7: Torque without and with the attack while the vehicle is in Cruise Mode.

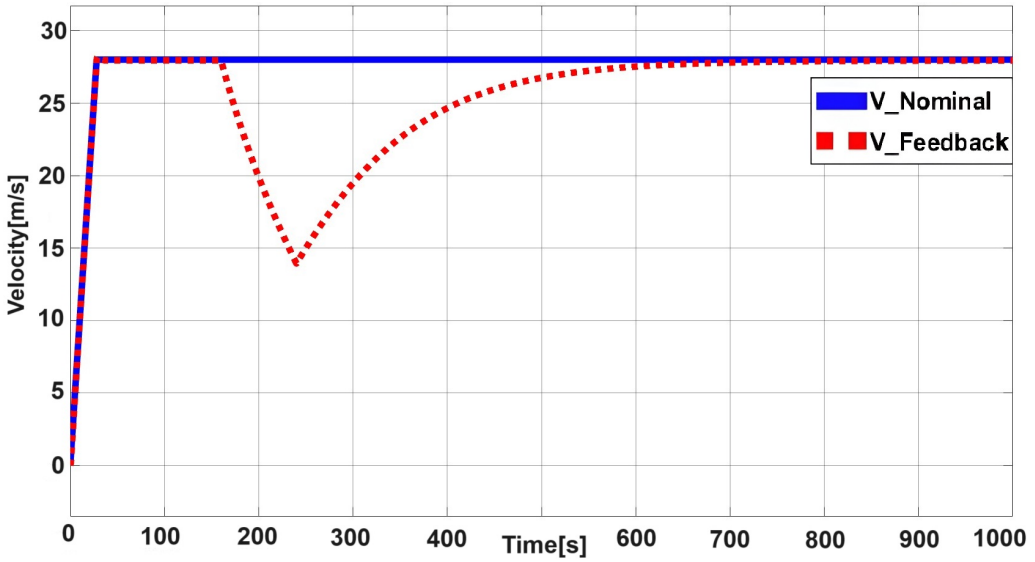


Fig. 8: Velocity trajectory during torque attack and in regular driving condition while the vehicle is in Cruise Mode.

view," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, July 2019.

[15] E. Seo, H. M. Song, and H. K. Kim, "GIDS: GAN based Intrusion Detection System for In-Vehicle Network," *In 2018 16th Annual Conference on Privacy, Security and Trust (PST), pp. 1-6. IEEE, 2018*, 07 2019.

[16] H. Lee, S. H. Jeong, and H. K. Kim, "Otids: A novel intrusion detection system for in-vehicle network by using remote frame," in *2017 15th Annual Conference on Privacy, Security and Trust (PST)*, IEEE, Aug. 2017.

[17] M. E. Verma, R. A. Bridges, M. D. Iannacone, S. C. Hollifield, P. Moriano, S. C. Hespeler, B. Kay, and F. L. Combs, "A Comprehensive Guide to CAN IDS Data & Introduction of the ROAD Dataset," *PLoS one 19, no. 1 (2024): e0296879*, 12 2020.

[18] MATLAB, "Simscape." https://it.mathworks.com/products/matlab.html, 2024. https://mathworks.com/products/simscape.html (accessed May 20, 2024).

[19] S. Documentation, "Simulation and model-based design," 2020.

[20] T. Hoppe, S. Kiltz, and J. Dittmann, "Security threats to automotive can networks - practical examples and selected short-term countermeasures," *Reliability Engineering & System Safety*, vol. 96, no. 1, pp. 11–25, 2011. Special Issue on Safecomp 2008.

[21] A. Gazdag, R. Ferenc, and L. ButtyÃ!'n, "CrySyS dataset of CAN traffic logs containing fabrication and masquerade attacks," *Scientific Data*, vol. 10, Dec. 2023.

[22] G. Dupont, A. Lekidis, J. J. den Hartog, and S. S. Etalle, "Automotive controller area network (can) bus intrusion dataset v2," 2019.

[23] M. Hanselmann, T. Strauss, K. Dormann, and H. Ulmer, "CANet: An Unsupervised Intrusion Detection System for High Dimensional CAN Bus Data," *IEEE Access*, vol. 8, pp. 58194–58205, 2020.

[24] CAN Hacker, "CAN Hacker: Tools and Resources for CAN Bus Hacking," 2023. Accessed: 2023-05-05.

[25] Vector Informatik GmbH, "Candb++ editor," 2024.