A Framework for Demonstrating and Mitigating CAN Injection Attacks in Vector CANoe: a Case Study Using ABS

Uma Kulkarni
Institute for Secure Cyber-Physical Systems
Hamburg University of Technology
Hamburg, Germany
email: uma.kulkarni@tuhh.de

Sibylle Fröschle
Institute for Secure Cyber-Physical Systems
Hamburg University of Technology
Hamburg, Germany
email: sibylle.froeschle@tuhh.de

***Abstract*—Modern vehicles rely heavily on the Controller Area Network (CAN) for communication between electronic control units, yet CAN lacks inherent security features like authentication, making it susceptible to attacks, such as message injection. In this paper, we develop a means of simulating and visualizing the effect of an attack on the communication traffic and the actual system and the effect of a countermeasure. Our tool is based on Vector CANoe simulation and demonstrates a CAN message injection attack and a simple rule-based Intrusion Detection System (IDS). The setup includes a custom Graphical User Interface (GUI) for interactive demonstration. This framework serves as an educational tool and a foundation for future research into more advanced attack and defense scenarios. This work takes a step towards developing simulation platforms or frameworks for security of in-vehicle networks that are flexible and based on industry suitable toolchains.**

Keywords - Controller Area Network (CAN); In-vehicle Network Security; Simulation; Demonstration for Education

I. Introduction and Background

Vehicles today are increasingly dependent on in-vehicle communication networks such as the Controller Area Network (CAN) bus to support functionalities ranging from basic engine control to advanced driver assistance systems. While the CAN protocol was originally designed for reliability and efficiency, it lacks built-in security features such as authentication, encryption and message integrity [1]. This makes it vulnerable to a range of cyber-attacks including spoofing, injection, and Denial-of-Service (DoS) [2][3][4]. The growing concern over automotive cybersecurity has prompted researchers and engineers to explore both, attack vectors and countermeasures.

However, there remains a gap between experimentally established attacks and targeted security concepts and being able to demonstrate and explore them in flexible and configurable environments such as those used by the automotive industry in their development processes. Vector CANoe provides a powerful platform to model, simulate and analyse automotive networks, making it well-suited for such exploratory work [5].

This paper presents a simulation-based demonstration using CANoe that showcases a common CAN bus attack and the impact of an Intrusion Detection System (IDS) designed to mitigate it. The setup simulates normal traffic on the CAN network, introduces a malicious node injecting abnormal messages, and then analyzes the detection capability of a rule-based IDS. The objective is to provide a practical framework for visualizing both vulnerabilities and potential defense mechanisms within the CANoe environment. This demonstration can be useful in teaching as well as serve as a starting point to build complex real-time simulation framework for other attack and countermeasure demonstrations.

The rest of the paper is structured as follows: Section II explains the methodology, Section III describes the demonstration itself followed by conclusion and future work in Section IV.

II. Methodology

This section outlines the experimental setup created in Vector CANoe to simulate a CAN bus environment, demonstrate the attack and the IDS. The demo contains the baseline network configuration, attack implementation, IDS integration and the Graphical User Interface (GUI).

(1) Configuration: The simulation is built on the sample

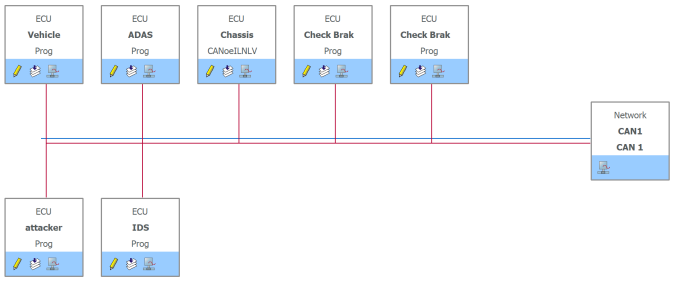


Fig. 1. Simulation set-up showing the CAN network.

configuration *ABS Brake Control* provided with the CANoe installation. The simulation, as seen in Figure 1, mainly consists of an embedded Simulink model for the *Chassis* block that simulates the application of brakes by providing the brake moment depending on two signals, status of ABS and brake activation. Other physical properties, such as mass of the vehicle are input as parameters to the model. The two *Check Brak* nodes are used for testing the Antilock Brake System (ABS) performance. The environment consists of two vehicles: the vehicle under consideration and a trailing vehicle. We added two Communication Access Programming Language (CAPL) programs *Vehicle* and *ADAS* that perform the functions such as setting the car in cruise control, calculating the distance between the vehicle under consideration and the

 ISBN: 978-1-68558-300-2

trailing vehicle among other things.
(2) Attack Simulation: We use the attacker model from [3]. The attack is simulated with the addition of a node called *attacker* which represents a compromised gateway. The behaviour is simulated in CAPL. The attacker can now inject malicious messages on the CAN with the aim to bring down the *Vehicle speed* suddenly causing an accident on the highway. When the attack is active, if the *Vehicle speed* is high (in this case more than `130 kmph`), *Brake Input* messages are injected at intervals of `50ms` for `3s`.
(3) IDS Simulation: We have implemented a simple rule-based IDS here. This is realised as an additional CAPL node. It utilises the periodic property of the *Brake Input* message. Since the time interval between two consecutive legitimate *Brake Input* messages is known, any message occurring on the bus between this is flagged.
(4) GUI: We developed the GUI to facilitate the demonstration.

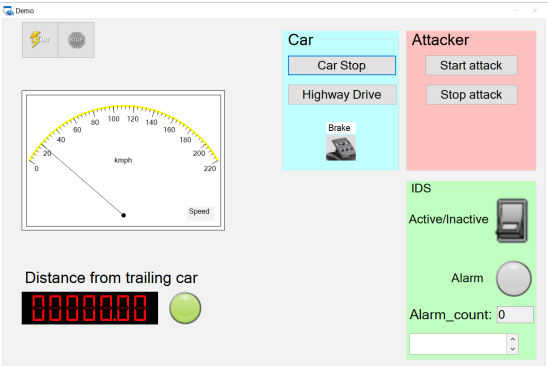


Fig. 2. GUI developed to facilitate the demonstration.

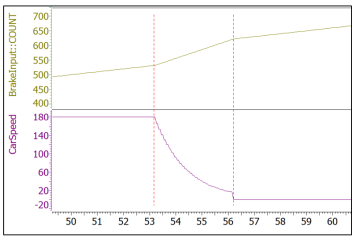
Here, the dial shows the current speed of the vehicle in `kmph`. The *Highway Drive* button provides a quick way to accelerate the vehicle to constant high speed and set the vehicle in cruise control like motion. The *Car Stop* button provides a quick way to decelerate the vehicle until it reaches a speed of `0`, i.e. it stops. The *Brake* button is used to simulate standard brake pedal action. The *Attacker* part of the GUI shown in Figure 2 provides the means to start the attack during the demonstration. The activation and deactivation of IDS can be done via the *IDS* part of the GUI. The distance from the trailing car is also displayed in `m` with an indicator LED showing if the distance is safe (green), is in warning limits (yellow) or dangerous (red).

III. Demonstration and Results

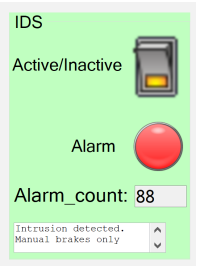
The steps of the demonstration flow are as follows: Without IDS: (1) Start the config from the GUI. (2) Press *Highway Drive* and wait till cruise control. (3) Press *Start attack*. (4) Observe - The warning LED goes from green to yellow to red and the cars in animation collide. See Figure 3. CAN message injection can be seen in the trace and graphics window in Display Desktop as shown in Figure 4(a), where the attack was started at around 53s after simulation start. With IDS: (1) Stop and Start config from the GUI. (2) Press *IDS* - see that IDS indicator is ON. (3) Press *Highway Drive* and wait till cruise control. (4) Press *Start attack*. (5) Observe - As

(a) Safe distance. (b) After attack.

Fig. 3. Demonstration results - cars animation



(a) Graph shows increased Brake input messages between 53s and 56s and corresponding decrease in vehicle speed when IDS is inactive.



(b) IDS section of GUI when IDS is active and attack is attempted.

Fig. 4. Demonstration results

shown in Figure 4(b), *Alarm* LED indicates that intrusion is detected. Warning text is displayed for the driver. Alarm count maintains history as long as Config is running (analogous to car ECUs being reflashed and reset). Cars in animation do not collide. From this point on, there is no effect of Start attack. Brake by wire is deactivated as a preventive measure but manual brake application is possible with the pedal and can be observed as a short dip in speed and also small dip in distance from trailing car. The cars continue to drive safely.

IV. Conclusion and Future Work

We have presented in this paper a CANoe-based simulation framework demonstrating a CAN bus injection attack and a rule-based IDS designed to mitigate it. This demonstration provides an effective means to visualise the said attack and countermeasure and observe the effects. Future work can be pursued in two directions. One is to add other attacks such as those exploiting fault-confinement to the demonstration and other security concepts. Second is to extend the demonstration to include real-time simulations with Hardware-in-the-loop (HIL) for more accurate analysis of attacks and IDS implementations.

References

[1] *CAN Specification, Robert Bosch GmbH, Postfach*, vol. 50, p. 15, 1991.
[2] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham *et al.*, "Experimental security analysis of a modern automobile," in *2010 IEEE symposium on security and privacy*. IEEE, 2010, pp. 447–462.
[3] S. Fröschle and A. Stühring, "Analyzing the capabilities of the CAN attacker," in *Computer Security–ESORICS 2017: 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part I 22*. Springer, 2017, pp. 464–482.
[4] Z. Tang, K. Serag, S. Zonouz, Z. B. Celik, D. Xu, and R. Beyah, "Eracan: Defending against an emerging can threat model," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, 2024, pp. 1894–1908.

[5] J. Hayward, A. Tomlinson, and J. Bryans, "Adding cyberattacks to an industry-leading can simulator," in *2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*. IEEE, 2019, pp. 9–16.