Implementation and Demonstration of Threat Over CAN Bus for Automotive Embedded System Equipped with ADAS

Soufian Zerouali[1](✉), Abdelhafid Messaoudi[1], Abdelhamid Rabhi[2], Mohammed Karrouchi[3], and Ismail Nasri[3]

[1] Energy, Embedded Systems and Information Processing Laboratory, National School of Applied Sciences, Mohammed First University, Oujda, Morocco
mr.zerouali.soufian@gmail.com

[2] Modeling, Information and Systems Laboratory, Picardy Jules Verne University, Amiens, France

[3] Electrical Engineering and Maintenance Laboratory, High School of Technology, Mohammed First University, BP. 473, Oujda, Morocco

Abstract. Embedded Software System has now become an integral part of automotive network architecture. It is based on the interconnection of several microcontrollers called Electronic Control Unit (ECU) by various buses especially the Controller Area Network (CAN). These ECUs are currently connected to the outside world as shown by mobile communication systems, on-board navigation or entertainment. Also, they are the basis of advanced driver assistance systems (ADAS) which represents a reduced driver's stress load or an active information safety system. At this stage, internal and external communications pass through the same hardware support and are isolated by rules defined by the software. This represents significant security limitations that can open the door for hackers to break into the internal components of vehicles.

The objectives of this article are part of an approach aimed at exposing two types of attacks on the CAN bus. These attacks show how an ADAS-equipped vehicle gives attackers the ability to take control of its critical modules. We conducted a number of experiments on a test bench and also on a real car to estimate its vulnerabilities to possible attacks.

Keywords: Vehicle · CAN bus · Vulnerabilities · Attack · ADAS · Embedded Software System · ECU

1 Introduction

The safety of embedded systems in automobiles to ensure and minimize the occurrence and consequences of road collisions is a problem that interest researchers from several fields such as security, artificial intelligence, electronics and embedded systems. The new generation of cars is equipped with several microcontrollers named Electronic Control Unit (ECU) [1] embedding software and interconnected via serial buses based on a standard protocol called Controller Area Network (CAN) [2] In order to coordinate their decision-making process and their actions. Indeed, the CAN bus is vulnerable due to its lack of authentication, access control, encryption or even message verification mechanism.

However, while technologies like embedded diagnostics systems, divertissement systems and advanced driving assistance systems (ADAS) [3] have received a lot of attention because they can make driving easier, they can also introduce a lot of hidden risks and become potentially vulnerable to attacks.

Several attack strategies against the CAN bus have been developed, either to investigate its flaws or carry out other malicious attempts. These attacks have demonstrated significant negative effects on vehicle security as well as significant risks to the lives and property of passengers.

2 Related Work

The security of embedded networks in the automotive environment is a problem that interest researchers [4] to solve the security problems of the CAN bus protocol [5]. In this context, some researchers and engineers have started to think about the computer security problem of vehicles, and in fact, a series of several methods that can be implemented to attack cars have been presented.

Amato and Coppolino provide a deep learning-based approach for finding attacks on the CAN-bus. They validate their approach by analyzing a real-world dataset with the injection of messages from different types of attacks: denial of service, fuzzy pattern attacks, and attacks against specific components [6].

Kurbanov and Grebennikov show how an ADAS-equipped vehicle gives attackers the opportunity to gain control over its critical modules by Acceleration Attack and Wheel Steering Attack [7].

Zhang and Binbin create frame injection attacks via OBD port that allow an attacker to control the car's light, lock, and steering system [8].

Ning and Wang create two attacks Spoofing and Bus-Off on a real vehicle and also they create a way of detection for each attack [9].

Farivar and Haghighi studies the security of the Adaptive Cruise Control System of smart cars by implementing Two covert/stealth attacks on the speed regulator with the implementation of a new method of intrusion detection and compensation that can be used to reveal and defend against such attacks. They conducted through Matlab experiments to gauge the viability of the proposed scheme in a simulated environment [10].

Khatri and Shrestha present a survey on security attacks launched against in-vehicle networks with countermeasures adopted by various researchers. They propose a new approach for intrusion detection which cover some other approaches limitations [11].

However, many details are still waiting to be explored. At this stage, our team is trying to find and examine other useful car attack methods before creating detection patterns through experimentation.

3 Background

The first step of our approach is to know the physical access points to the automotive network architecture. In our case, an access via the diagnostic vehicle OBD2 were carried out on a real Car (the brand of which cannot be announced) and also a physical access directly in the CAN bus of the representative test model of the vehicle that we will present it in the next chapter.

In a second step, know the CAN frames responsible for each functionality that we want to attack. At this point, applying reverse engineering is necessary to analyze the CAN bus frames and identify the functionality of each signal since each car manufacturer and car model has a different communication implementation.

3.1 Vehicle Investigation

Our methodology is to identify the functionality of each signal of CAN frames presented on the figure below (see Fig. 1). The first step of this methodology is performing a physical connection to the CAN bus by Vector CANalyzer Tool and read the Frames. And then, we made a record of a CAN trace (CAN log1).

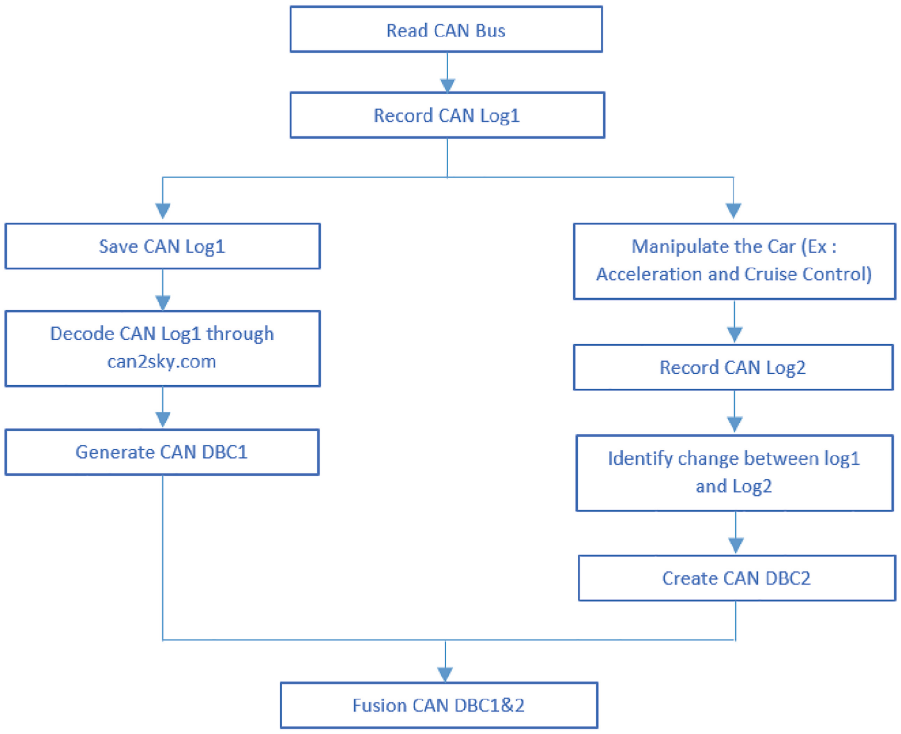

Read CAN Bus
Record CAN Log1
Save CAN Log1
Decode CAN Log1 through can2sky.com
Generate CAN DBC1
Manipulate the Car (Ex : Acceleration and Cruise Control)
Record CAN Log2
Identify change between log1 and Log2
Create CAN DBC2
Fusion CAN DBC1&2

Fig. 1. Diagram of the steps for the definition of the CAN database file.

After that, we define 2 ways to have a good result of the CAN Database file (CAN DBC):

- Once the CAN bus Log1 is recorded and saved, we upload it to the service can2sky.com to decode it and then generate the CAN DBC1 file.
- Perform some manipulation especially in our case the acceleration and setting the Cruise Control and at the same time record the trace CAN bus Log2. And then, identify the byte of the frames where there is changes to know if the signal is related to the accelerator or the cruise control. Therefore, according to this manual investigation we can create the CAN DBC2 file.

The last step is to fusion the 2 DBC files to have a good result to help us to command whatever we want on the Car.

3.2 Test Model and CAN Devices

As shown in the Fig. 2 and Fig. 3 below, the main components that can be considered to meet the needs of the experimental model are the use of an ECU (Electronic Controller Unit), a dashboard represented by a vector panel and a screen 7-inch touchscreen car radio that integrates several technologies that will allow us to carry out all our future tests Short Range Attack (through Bluetooth and Wifi). To interface the various components with each other, we offer a harness grouping the 12V power supply and the CAN bus with the two 120 Ω termination resistors.

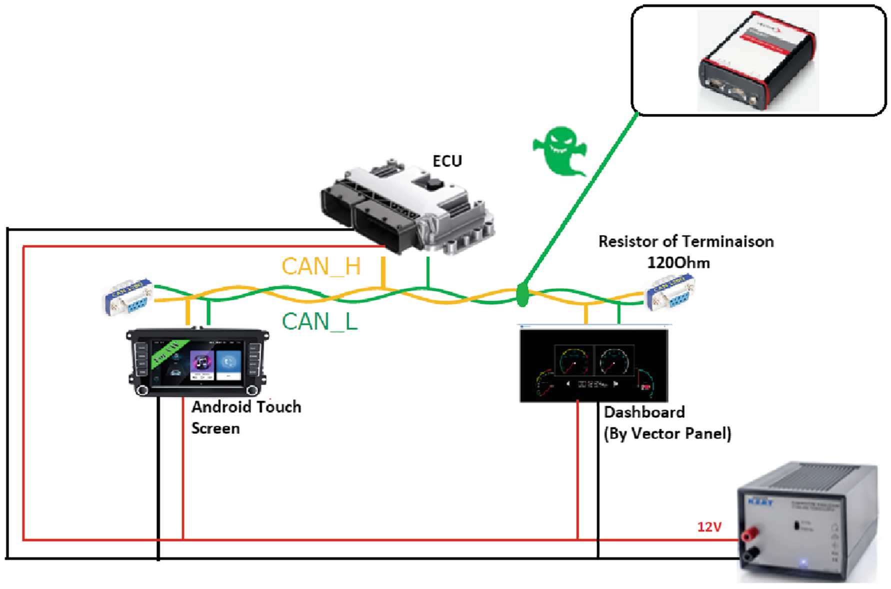


Fig. 2. Block diagram polishing the test model.

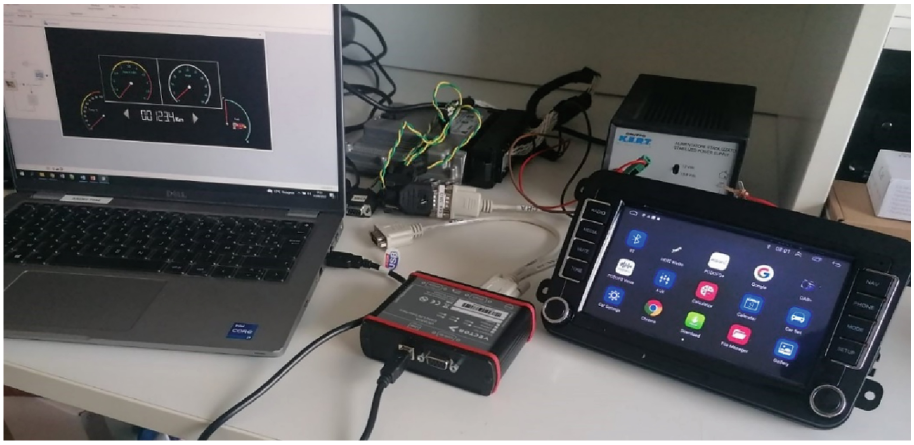


Fig. 3. Real picture of the test model.

4 Experiment and Result

4.1 CAN Bus Load by Injection Attack

The experiments of the CAN bus load by Injection Attack were carried out on a test model (see Fig. 4).

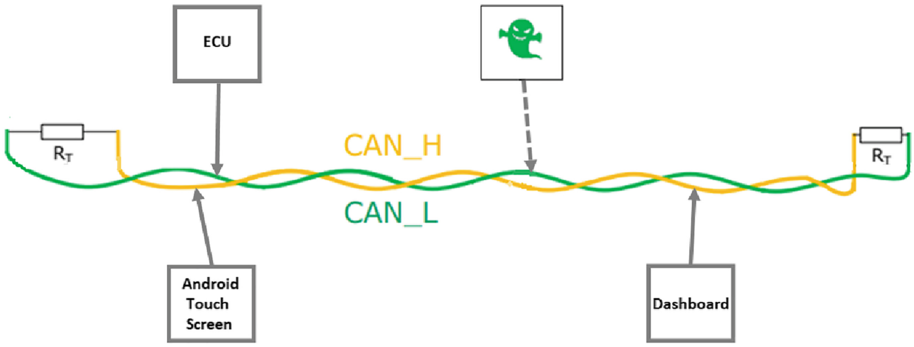
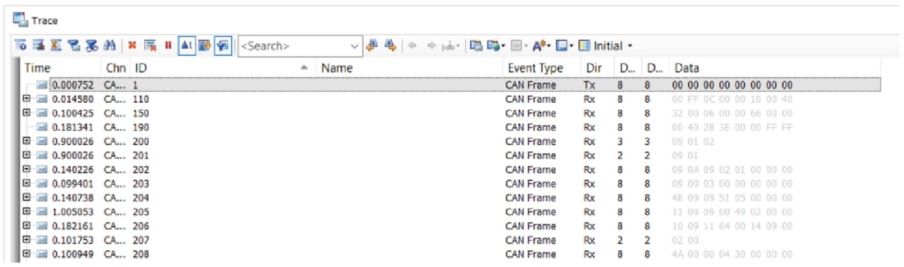


Fig. 4. CAN bus load by Injection Attack.

The principle of this attack is to send a hazard frames into the CAN bus with a very low periodicity in order to load the bus. This will lead to bus saturation and a total stop of the vehicle which can lead to a very dangerous situation especially when a high driving speed.

In our scenario, we send a standard frame with a CAN ID of 0 × 1 and a periodicity of 1 ms in a 250 Kbit/s CAN bus (see Fig. 5).



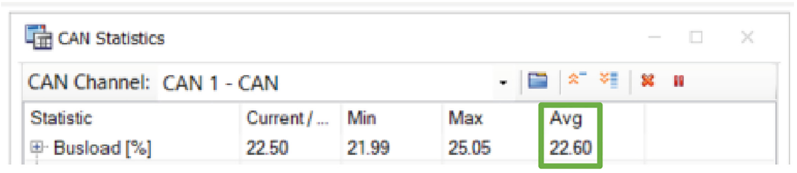
Trace

<Search> Initial

Time	Chn	ID	Name	Event Type	Dir	D...	D...	Data
0.000752	CA...	1		CAN Frame	Tx	8	8	00 00 00 00 00 00 00 00
0.014580	CA...	110		CAN Frame	Rx	8	8	00 FF 0C 00 00 10 00 40
0.100425	CA...	150		CAN Frame	Rx	8	8	32 00 06 00 00 66 00 00
0.181341	CA...	190		CAN Frame	Rx	8	8	00 40 2B 3E 00 00 FF FF
0.900026	CA...	200		CAN Frame	Rx	3	3	09 01 02
0.900026	CA...	201		CAN Frame	Rx	2	2	09 01
0.140226	CA...	202		CAN Frame	Rx	8	8	09 0A 09 02 01 00 00 00
0.099401	CA...	203		CAN Frame	Rx	8	8	09 09 03 00 00 00 00 00
0.140738	CA...	204		CAN Frame	Rx	8	8	4B 09 09 51 05 00 00 00
1.005053	CA...	205		CAN Frame	Rx	8	8	11 09 09 00 49 02 00 00
0.182161	CA...	206		CAN Frame	Rx	8	8	10 09 11 64 00 14 09 00
0.101753	CA...	207		CAN Frame	Rx	2	2	02 00
0.100949	CA...	208		CAN Frame	Rx	8	8	4A 00 00 04 30 00 00 00

Fig. 5. CAN trace of the bus load by Injection Attack.

The result of the CAN bus load by Injection Attack is described on the Fig. 6 and Fig. 7.

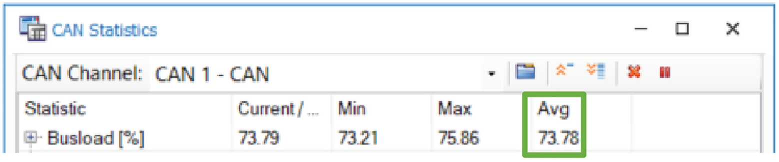


CAN Statistics

CAN Channel: CAN 1 - CAN

Statistic	Current / ...	Min	Max	Avg
Busload [%]	22.50	21.99	25.05	22.60

Fig. 6. CAN load of the bus before sending the frame.



CAN Statistics

CAN Channel: CAN 1 - CAN

Statistic	Current / ...	Min	Max	Avg
Busload [%]	73.79	73.21	75.86	73.78

Fig. 7. CAN load of the bus before sending the frame.

We can conclude that with a simple frame of 1ms on a CAN bus of 250 Kbit/s we can overload the bus with around 51% of the CAN load total. Therefore, in the case of injecting several frames, it will lead to bus saturation and the ECU will automatically restart (due to a safety reaction) which can lead to a very dangerous situation.

4.2 Cruise Control Acceleration Attack

The experiments of the Cruise Control Acceleration Attack were carried out on a real car (the brand of which cannot be announced) (see Fig. 8).

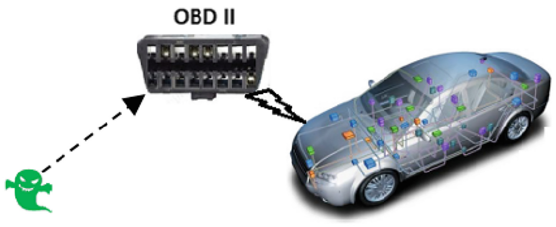


Fig. 8. CAN load of the bus before sending the frame.

The Cruise control is a system for maintaining the speed of the car at a desired level. This system capable of taking over the accelerator once the driver activates the cruise control via a button. Once activated, the driver can set the cruise speeds up and down via 2 buttons. In our case, the experimental car contains the cruise control buttons on the steering wheel.

The principle of the Cruise Control Acceleration attack is to send the increasing speed signal request instead of the Cruise Control Speeds up button when the Cruise control is activated at 5 km/h in our case (see Fig. 9).

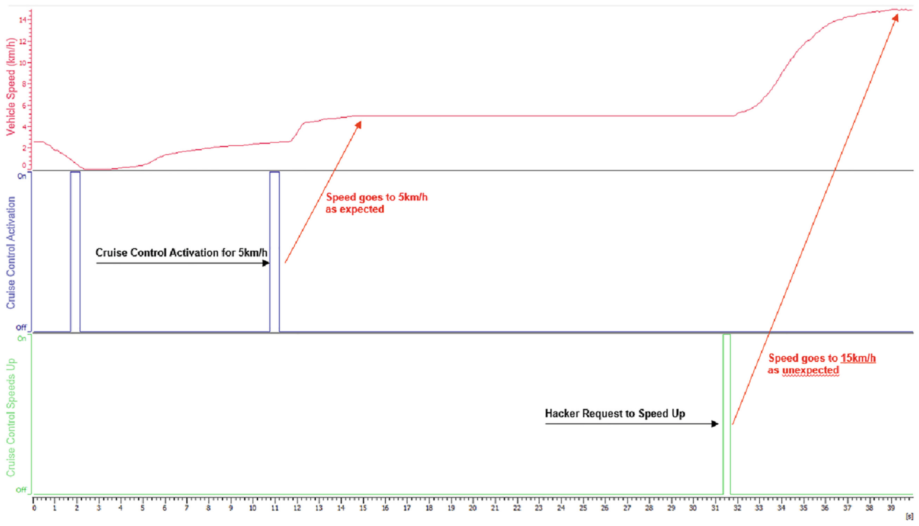


Fig. 9. Cruise Control Acceleration Attack Results.

The result of the Hacker request is increasing immediately the speed at 15 km/h (see Fig. 9). Therefore, we can imagine the very dangerous situation of having another car right in front.

5 Conclusion and Future Work

In this article, we presented our test bench which allows us to simulate the main part of the vehicle and also allows us to verify the feasibility of the attack mechanism proposed by experiments. The attacks provided have shown that an intruder is able to physically access the CAN bus and will be able to take full control of the vehicle. The first vehicle attack scenario is to inject one or more frames to load the CAN bus. This attack usually causes car ECUs to restart which could be very dangerous when driving at high speed. The second, potentially fatal, scenario for attacking a car goes like this: The attacker accelerates the vehicle at high speed during the activation of the cruise control which results in a frontal impact of the vehicle in front.

Theoretically, in addition to physical access to the bus, an attacker may take control of the vehicle via a multimedia system that includes wireless access points like Bluetooth or Wi-Fi.

In the future, our team will keep conducting research to identify more potential threats. We will also work to develop models and solutions for safeguarding cars against illegal attacks.

References

1. Jafarnejad, S., Codeca, L., Bronzi, W., Frank, R., Engel, T.: A car hacking experiment: when connectivity meets vulnerability. IEEE Globecom Workshops (GC Wkshps) **2015**, 1–6 (2015). https://doi.org/10.1109/GLOCOMW.2015.7413993
2. Buscemi, A., Turcanu, I., Castignani, G., Crunelle, R., Engel, T.: CANMatch: a fully automated tool for CAN bus reverse engineering based on frame matching. IEEE Trans. Veh. Technol. **70**(12), 12358–12373 (2021). https://doi.org/10.1109/TVT.2021.3124550
3. "Fail-Operational Safety Architecture for ADAS Systems Considering Domain ECUs," in The Safety of Controllers, Sensors, and Actuators , SAE, 2020, pp. 55–67 ()2020
4. Woo, S., Jo, H.J., Kim, I.S., Lee, D.H.: A practical security architecture for in-vehicle CAN-FD. IEEE Trans. Intell. Transp. Syst. **17**(8), 2248–2261 (2016). https://doi.org/10.1109/TITS.2016.2519464
5. Gmiden, M., Gmiden, M.H., Trabelsi, H.: Cryptographic and Intrusion Detection System for automotive CAN bus: Survey and contributions. In: 2019 16th International Multi-Conference on Systems, Signals & Devices (SSD), pp. 158–163 (2019). https://doi.org/10.1109/SSD.2019.8893165
6. Amato, F., Coppolino, L., Mercaldo, F., Moscato, F., Nardone, R., Santone, A.: CAN-bus attack detection with deep learning. IEEE Trans. Intell. Transp. Syst. **22**(8), 5081–5090 (2021). https://doi.org/10.1109/TITS.2020.3046974
7. Kurbanov, A., Grebennikov, S., Gafurov, S., Klimchik, A.: Vulnerabilities in the vehicle's electronic network equipped with ADAS system. In: 2019 3rd School on Dynamics of Complex Networks and their Application in Intellectual Robotics (DCNAIR), pp. 100–102 (2019). https://doi.org/10.1109/DCNAIR.2019.8875529
8. Zhang, Y., Ge, B., Li, X., Shi, B., Li, B.: Controlling a car through OBD injection. In: 2016 IEEE 3rd International Conference on Cyber Security and Cloud Computing (CSCloud), pp. 26–29 (2016). https://doi.org/10.1109/CSCloud.2016.42
9. Ning, J., Wang, J., Liu, J., Kato, N.: Attacker identification and intrusion detection for in-vehicle networks. IEEE Commun. Lett. **23**(11), 1927–1930 (2019). https://doi.org/10.1109/LCOMM.2019.2937097

10. Farivar, F., Sayad Haghighi, M., Jolfaei, A., Wen, S.: On the security of networked control systems in smart vehicle and its adaptive cruise control. IEEE Trans. Intell. Transp. Syst. **22**(6), 3824–3831 (2021). https://doi.org/10.1109/TITS.2021.3053406
11. Khatri, N., Shrestha, R., Nam, S.Y.: Security issues with in-vehicle networks, and enhanced countermeasures based on blockchain. Electronics **10**, 893 (2021). https://doi.org/10.3390/electronics10080893