A Real-time Method for CAN Bus Intrusion Detection by Means of Supervised Machine Learning

Francesco Mercaldo[1,2][a], Rosangela Casolare[1], Giovanni Ciaramella[2], Giacomo Iadarola[2], Fabio Martinelli[2], Francesco Ranieri[1] and Antonella Santone[1]

[1]*University of Molise, Campobasso, Italy*

[2]*Institute for Informatics and Telematics, National Research Council of Italy (CNR), Pisa, Italy*

Keywords: Automotive, CAN, Machine Learning, Classification, Security.

Abstract: Nowadays vehicles are not composed only of mechanical parts, exits a plethora of electronics components in our cars, able to exchange information. The protection devices such as the airbags are activated electronically. This happens because the braking or acceleration signal from the pedal to the actuator arrives through a packet. The latter is an electronic and not a mechanical signal. For packets transmission a bus, i.e., the Controller Area Network, was designed and implemented in vehicles. This bus was not designed to receive access from the outside world, which happened when info-entertainment systems were introduced, opening up the possibility of accessing bus information from devices external to the vehicle. To avoid the possibility of those attacks, in this research article, we propose a method aimed to detect intrusions targeting the CAN bus. In particular, we analyze packets transiting through the CAN bus, and we build a set of models by exploiting supervised machine learning. We experiment with the proposed method on three different attacks (i.e., speedometer attack, arrows attack, and doors attack), obtaining interesting performances.

1 INTRODUCTION

Currently, cars are no longer just mechanical vehicles: they contain a plethora of electronic collaborating components connected to the network, that allows monitoring and controlling of the status of the vehicle. Each electronic component can continuously communicate with other nearby components, producing a continuous flow of data in real-time for instance, to monitor the state of several components, the state of the vehicle in general, and in particular, is used to increase road safety (Martinelli et al., 2017).

CAN is short for *Controller Area Network*: it is an electronic communication bus defined by the ISO 11898 standards designed to permit the packet exchange between vehicle electronic components: each CAN message contains the priority and the content of the transmitted data. Moreover, this standard defines how communication happens, how the wiring is configured, and how messages are constructed. Collectively, this system is referred to as a CAN bus.

The introduction of electronic devices into vehicles turned cars into cyber-attack targets and allowed a plethora of new kinds of cyber-attacks, increasingly complex and arduous to mitigate. Most of them have the purpose of altering the normal functioning of the car itself, and it represents a fatality for anyone on board the vehicle or near it.

Attacks targeting the latest generation cars are increasingly frequent and dangerous. An attacker in the automotive context has a wide choice of attack surfaces (Keuper and Alkemade, 2018). Some cyber-attacks have to be launched in the proximity of the car, while other attacks can be carried out from anywhere in the world through wireless connections that the cars have on board.

For this reason, we propose a method for intrusion detection of malicious packets targeting the CAN bus. The method can identify an attack in progress in real-time. This technique uses supervised machine learning to distinguish between three different forms of attacks (speedometer attack, arrows attack, and doors attack), obtaining interesting performances.

The paper has been organized as follows: in Section 2 we report current state-of-art literature in automotive cybersecurity in the automotive context; in

[a] https://orcid.org/0000-0002-9425-1657

Mercaldo, F., Casolare, R., Ciaramella, G., Iadarola, G., Martinelli, F., Ranieri, F. and Santone, A.
A Real-time Method for CAN Bus Intrusion Detection by Means of Supervised Machine Learning.
DOI: 10.5220/0011267500003283
In *Proceedings of the 19th International Conference on Security and Cryptography (SECRYPT 2022)*, pages 534-539
ISBN: 978-989-758-590-6; ISSN: 2184-7711

Section 3 we provide background notions about the CAN bus; in Section 4 the proposed intrusion detection method is presented; in Section 5 we show the experimental analysis results; in the last section conclusion and future research directions are drawn.

2 RELATED WORK

Road safety is an ever-present topic, researchers are constantly working to improve and increase controls on the traffic of information exchanged by the electronic components that make up the computer systems of modern cars. Several works in the literature focus on this very important issue.

In (Amato et al., 2021) authors propose a method that uses deep learning techniques to detect attacks targeting the CAN-bus. This method is based on the use of *Neural Networks* and *MultiLayer Perceptrons*, and for the analysis is considered a real-world dataset that has the injection of messages belonging to different types of attacks, such as denial of service, attacks against particular components and other. Differently, the proposed method is evaluated on three different kinds of attacks i.e., speedometer, doors, and arrows.

In (Checkoway et al., 2011), authors sought to answer the question "whether automobiles can also be susceptible to remote compromise", and to do so they analyzed the outer attack surface of a modern automobile. They found that remote exploitation is possible through a wide range of attack vectors (i.e., mechanical tools, CD players, Bluetooth, and cellular radios) highlighting that wireless communication channels allow for long-distance vehicle control, allowing for detecting the position of the vehicle and also to carry out a theft. In this paper, the authors propose a mitigation strategy for each aforementioned attack, which is different from the attacks considered by us (i.e., speedometer attack, arrow attack, and doors attack).

By exploiting vulnerabilities in the external interfaces of the car, such as Wi-Fi, Bluetooth and physical connections, it is possible to access the CAN bus of the automobile and send commands to control the car. To mitigate this threat, it is necessary to detect malicious behavior on the CAN bus and in this regard, Taylor et. colleagues (Taylor et al., 2016) propose an anomaly detector based on a long-term memory neural network; we have also used the neural network, but we have not limited ourselves only to the use of this technique. The operation of the detector is based on learning to predict the next words of data sent by the various senders on the bus. The unexpected bits contained in the next word are flagged as anomalous. In this regard, the authors evaluate the detector by synthesizing anomalies (which are designed to mimic attacks reported in literature) using modified CAN bus data.

The latter work is always focused on intrusion detection, but it is a little different from the aforementioned, and in particular from our method, as it is based on a sort of physical intrusion: car theft. Here (Kwak et al., 2016) the authors have thought of detecting car thefts through the behavior of users while driving the vehicle, then analyzing and recognizing their driving style through some measurements of specific values, carried out with the vehicle sensors. Specifically, to detect theft, they worked on a method to which they add mechanical characteristics of automotive parts usually excluded in other works in the literature, which can be useful for identifying the driving behavior of drivers, thanks to the variation they undergo depending on different driving styles. This work involves the analysis of CAN packets and demonstrates that the model adopted is reliable and discriminates between car owners and impostors.

3 BACKGROUND

In this section preliminary notions about CAN bus and the attacks targeting this protocol are provided.

3.1 CAN Bus

CAN bus is a protocol based on packets exchanged between the electronic components of cars. It works in multimaster - multislave mode, i.e., the units connected to the bus (called nodes) work either as a master, sending and receiving information, or as a slave, receiving only information and providing it on request. The CAN-bus communication takes place via sensors or actuators capable of producing data independently and then putting them on the BUS i.e., by generating CAN packets.

The CAN packets are contained in a message and each message is composed of the following values:

- *Timestamp*: recorded time (s);
- *CAN ID*: identifier of CAN message in HEX (i.e., 03B1);
- *DLC*: number of data bytes, from 0 to 8;
- *DATA[0 7]*: data value (byte);

To generate CAN packets, in this paper we resort to the ICSim simulator[1], a tool for learning the main

[1] https://github.com/zombieCraig/ICSim

functions of the CAN bus and simulating CAN traffic. To allow a correct simulation of the activities of the CAN bus, the CAN-utils[2] package was installed: this package enables ICSim to imitate the communications between the CAN network and the vehicle. CAN-utils provide five sub-modules as explained below:

- *cangen*: it is able to generate CAN frames for testing purposes. To take advantage of its functions, it is necessary to specify the interface in which the CAN frame is to be generated (in this case, the interface is vcan0) and then execute the command;
- *candump*: aimed to store CAN frames, which in turn are saved in a folder chosen by the user. In addition to the store, candump also has the CAN packet recording function;
- *canplayer*: it provides the user with the right to reproduce CAN frames. Since it is a utility unable to generate data but only reproduce it, its functions are closely related to the candump utility;
- *cansniffer*: it is a utility used to observe changes in real-time during CAN frame traffic
- *cansend*: it is used to send CAN frames to a specific interface.

3.2 The Attacks

In this paper, we experiment with three types of attacks targeting the CAN network. The attacks simulation is shown in Figure 1, where it is possible to see the simulation of one of the attacks under analysis (i.e., the speedometer attack) carried out thanks to IC Simulator.

To simulate the attacks, after having correctly installed and set up the *ICSim* software, it is possible to generate and read the CAN data traffic through the use of different windows, as shown in Figure 1; to activate the simulated dashboard, it is necessary to start the *CANBus Control Panel* (i.e., the window at the top left in Figure 1), which is the interface that allows control of the simulator and performs the attacks. The second window on the top right is the one relating to *IC Simulator* which simulates part of the dashboard of a common car we can observe the presence of the speedometer and directional arrows. At the bottom left there is the *Terminal* window that allows controlling the joystick in the Control Panel, while on the right there is the window that reports *CAN traffic* in real-time, showing all its changes.

The considered attacks are described below.

[2]https://github.com/linux-can/can-utils

Speedometer Attack: during the simulation phase, we identified the packet containing the information relating to the speedometer in the CAN traffic, with the help of the *cansniffer* utility: this packet, which is identified with the ID 244, allows an attacker to alter the normal operation of the speedometer, by moving the needle indicating the speed, to a level set by the attacker. This type of attack can be implemented through the use of the command:

```
cansend vcan0 244 \# 000000FFFF
```

The packet containing the attack is sent thanks to the *cansend* utility, which is followed by the name of the virtual interface, the packet ID, and finally a series of data that allow the attacker to set the speed (for example, if at the end of the code the attacker writes 50FF, the needle reaches the middle of the speedometer). To repeatedly send this defective packet, making the attack permanent all the time, we use the command:

```
While true;
do cansend vcan0 244 \# 000000FFFF;
done
```

Arrows Attack: this attack involves tampering with the packet containing the data relating to the operation of the position lights. In this case, a possible attacker violates CAN traffic frame 188 by making the arrows of a car continuously lit or not, checking the intermittence and duration. In the simulated environment, this type of attack is implemented with the command:

```
cansend vcan0 188 \# 01
```

The position of the arrows can be checked by alternating the values 01 (left), 02 (right), and 03 (both on) respectively. The following command is used to carry out the attack:

```
While true;
do cansend vcan0 188 \# 01;
done
```

Doors Attack: the latest simulated attack tampered with the doors of a car, controlling opening and closing and thus making them completely unusable. The packet containing the data relating to the operation is designated with the ID 19B and in the simulation, this type of attack was implemented with the command:

```
cansend vcan0 19B \# 00000F
```

Similarly to the previous cases, the connection allows to tamper with some or all the doors modifying the last character of the code with the values: *F* - doors closed; *A* - left side doors open; *5* - right side doors open; *C* - front doors open; *3* - rear doors open. By typing the characters E, B, D, and S it is possible to check the status of every single door.

To make the attack we have to use the following command:

```
While true;
```

```
do cansend vcan0 19B \# 00000F;
done
```

4 A METHOD FOR CAN BUS INTRUSION DETECTION

In this section, we describe the proposed method for the detection of intrusion targeting the CAN bus.

Figure 2 shows the workflow relating to the proposed method. As can be seen in the first step depicted in Figure 2, we start from a dataset composed of (legitimate and malicious CAN packets) stored in *csv* files.

In order to discriminate packets injected by an attacker from the normal ones, we consider these bytes as the feature vector composed in the following way:

- 1st byte: F1 feature;
- 2nd byte: F2 feature;
- 3rd byte: F3 feature;
- 4th byte: F4 feature;
- 5th byte: F5 feature;
- 6th byte: F6 feature;
- 7th byte: F7 feature;
- 8th byte: F8 feature.

We resort to supervised machine learning 2, in particular to enforce the conclusion validity four different algorithms are exploited: *Random Forest*, *Constant*, *Neural Network* and *Logistic Regression*.

The classification analysis consists of building classifiers to evaluate the feature vector accuracy to distinguish between injected and normal messages.

For classifier training, we defined T as a set of labeled messages *(M, l)*, where each M is associated to a label $l \in \{IM, NM\}$. For each M we built a feature vector $F \in R_y$, where y is the number of the features used in training phase ($y = 8$).

For the learning phase, we use a k-fold cross-validation: the dataset is randomly partitioned into k subsets. A single subset is retained as the validation dataset for testing the model, while the remaining $k-1$ subsets of the original dataset are used as training data. We repeated the process for $k = 5$ times; each one of the k subsets has been used once as the validation dataset. To obtain a single estimate, we computed the average of the k results from the folds.

We evaluated the effectiveness of the classification method with the following procedure:

1. build a training set $T \subset D$;
2. build a testing set $T' = D \div T$;
3. run the training phase on T;
4. apply the learned classifier to each element of T'.

Each classification was performed using 80% of the dataset as a training dataset and 20% as a testing dataset employing the full feature set.

5 EVALUATION

For dataset generation we resort to a simulated environment i.e., we installed on a Linux virtual machine the *ICSim* tool, a traffic simulator, and, starting from it, we generated (malicious and legitimate) CAN packets. The generated packets were stored by exploiting the Candump CAN utility.

For each attack, we considered five minutes of CAN traffic.

To build and evaluate the considered supervised machine learning models, we resort to the Orange[3] tool-suite, an open-source data mining software. Orange allows performing data analysis by building a visual scheme of them. It also allows an interactive data exploration for a quick qualitative analysis.

The effectiveness of the four different algorithms for intrusion detection is evaluated through four distinct metrics: Accuracy, Precision, Recall, and F-Measure.

Table 1 shows the results obtained from the experimental analysis.

By analyzing the results reported in Table 1, we observe that among all the models, the Constant is the model that gives lower values if compared to the others.

The best performances are given by the remaining algorithms (i.e., Random Forest, Neural Network, and Logistic Regression), which reach a value of 1, indicating that for each prediction made, it is possible to recognize and distinguish every malicious packet present in the registered CAN traffic. Only for the Logistic Regression algorithm applied to the Doors attack, the values obtained for each metric are equal to 0.99, still representing a satisfying result.

As for the analysis carried out on all the attacks, we have the values relating to Random Forest, Neural Network, and Logistic Regression equal to 1; In the Constant model, values are all greater than 0.9, except for the Precision which is equal to 0.89. In general, we can say that the performances obtained are very satisfactory.

[3]https://orangedatamining.com/

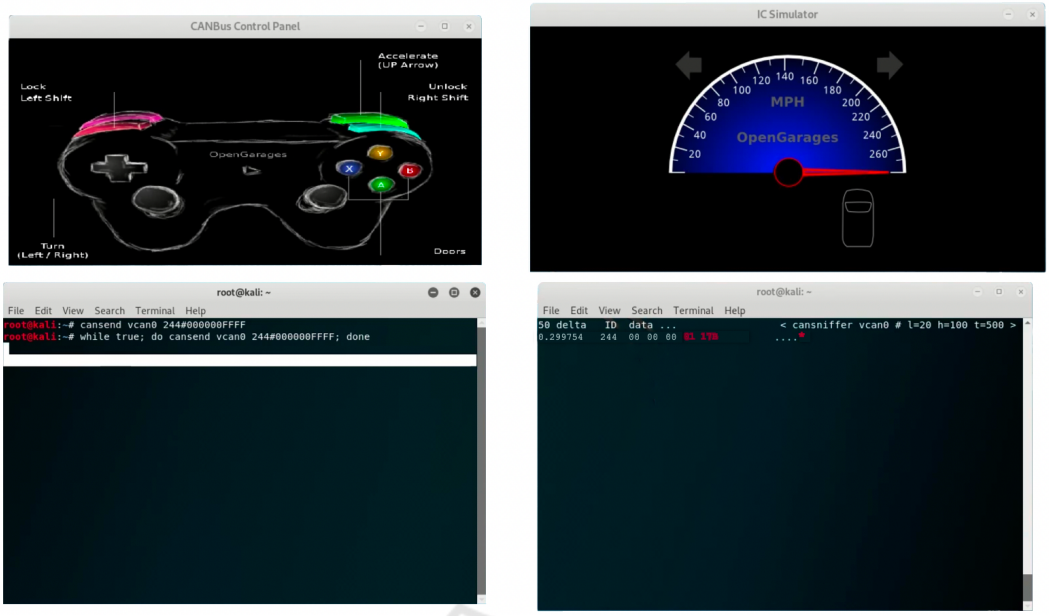

Figure 1: The simulation environment: the speedometer attack traffic is generated in the command prompt window.

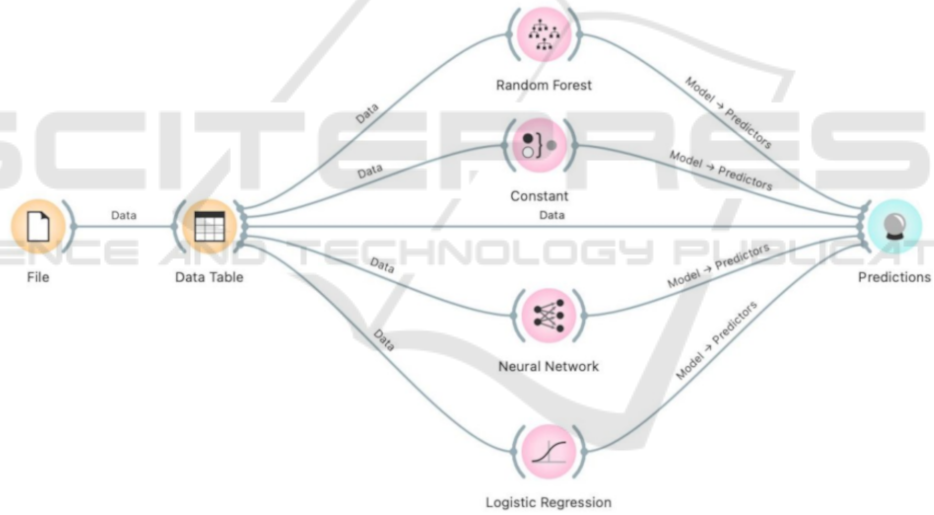

Figure 2: Workflow of the proposed method for intrusion detection.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a method aimed to discriminate between malicious and legitimate CAN packets with supervised machine learning. As evidenced by the experimental analysis results, the approach suggested in this paper achieved good results.

For future works, as a first aim, we would like to introduce new types of attack, for example, we could conduct detailed experimentation on the replay attack. This attack permits the attacker to record CAN traffic related to the driver's actions and make the vehicle reproduce the same actions. Furthermore, the replay attack turns out to be very difficult to identify, as it is based on sequences of legitimate CAN packets reproduced from a previous transmission.

ACKNOWLEDGEMENTS

This work has been partially supported by MIUR - SecureOpenNets, EU SPARTA, CyberSANE and E-CORRIDOR projects.

Table 1: Classification result.

Model applied to Speedometer attack	**Accuracy**	**Precision**	**Recall**	**F-Measure**
Random Forest	1.00	1.00	1.00	1.00
Constant	0.96	0.93	0.96	0.95
Neural Network	1.00	1.00	1.00	1.00
Logistic Regression	1.00	1.00	1.00	1.00
Model applied to Doors attack	**Accuracy**	**Precision**	**Recall**	**F-Measure**
Random Forest	1.00	1.00	1.00	1.00
Constant	0.99	0.98	0.99	0.99
Neural Network	1.00	1.00	1.00	1.00
Logistic Regression	0.99	0.99	0.99	0.99
Model applied to Arrows attack	**Accuracy**	**Precision**	**Recall**	**F-Measure**
Random Forest	1.00	1.00	1.00	1.00
Constant	0.98	0.97	0.98	0.98
Neural Network	1.00	1.00	1.00	1.00
Logistic Regression	1.00	1.00	1.00	1.00
Model applied to all attacks	**Accuracy**	**Precision**	**Recall**	**F-Measure**
Random Forest	1.00	1.00	1.00	1.00
Constant	0.94	0.89	0.94	0.91
Neural Network	1.00	1.00	1.00	1.00
Logistic Regression	1.00	1.00	1.00	1.00

REFERENCES

Amato, F., Coppolino, L., Mercaldo, F., Moscato, F., Nardone, R., and Santone, A. (2021). Can-bus attack detection with deep learning. *IEEE Transactions on Intelligent Transportation Systems*.

Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., Koscher, K., Czeskis, A., Roesner, F., and Kohno, T. (2011). Comprehensive experimental analyses of automotive attack surfaces. In *20th USENIX Security Symposium (USENIX Security 11)*.

Keuper, D. and Alkemade, T. (2018). 'the connected car ways to get unauthorized access and potential implications. *Computest, Zoetermeer, The Netherlands, Tech. Rep*.

Kwak, B. I., Woo, J., and Kim, H. K. (2016). Know your master: Driver profiling-based anti-theft method. In *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, pages 211–218. IEEE.

Martinelli, F., Mercaldo, F., Nardone, V., and Santone, A. (2017). Car hacking identification through fuzzy logic algorithms. In *2017 IEEE international conference on fuzzy systems (FUZZ-IEEE)*, pages 1–7. IEEE.

Taylor, A., Leblanc, S., and Japkowicz, N. (2016). Anomaly detection in automobile control network data with long short-term memory networks. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 130–139. IEEE.