future internet

MDPI

Article

Automotive Cybersecurity Application Based on CARDIAN

Emanuele Santonicola [1], Ennio Andrea Adinolfi [1,2,*], Simone Coppola [1] and Francesco Pascale [1,*]

[1] MinervaS S.P.A., Via Giovanni Paolo II 132, 84084 Fisciano, SA, Italy; santonicolaemanuele@gmail.com (E.S.); simo992simo@gmail.com (S.C.)
[2] Department of Industrial Engineering, University of Salerno, 84084 Fisciano, SA, Italy
* Correspondence: ea.adinolfi@minervas.it (E.A.A.); francescopascale87@gmail.com (F.P.); Tel.: +39-320-750-4440

Abstract: Nowadays, a vehicle can contain from 20 to 100 ECUs, which are responsible for ordering, controlling and monitoring all the components of the vehicle itself. Each of these units can also send and receive information to other units on the network or externally. For most vehicles, the controller area network (CAN) is the main communication protocol and system used to build their internal network. Technological development, the growing integration of devices and the numerous advances in the field of connectivity have allowed the vehicle to become connected, and the flow of information exchanged between the various ECUs (electronic control units) becomes increasingly important and varied. Furthermore, the vehicle itself is capable of exchanging information with other vehicles, with the surrounding environment and with the Internet. As shown by the CARDIAN project, this type of innovation allows the user an increasingly safe and varied driving experience, but at the same time, it introduces a series of vulnerabilities and dangers due to the connection itself. The job of making the vehicle safe therefore becomes critical. In recent years, it has been demonstrated in multiple ways how easy it is to compromise the safety of a vehicle and its passengers by injecting malicious messages into the CAN network present inside the vehicle itself. The purpose of this article is the construction of a system that, integrated within the vehicle network, is able to effectively recognize any type of intrusion and tampering.

Keywords: automotive; cybersecurity; CAN bus; IoT; intrusion detection systems; Bayesian network

Citation: Santonicola, E.; Adinolfi, E.A.; Coppola, S.; Pascale, F. Automotive Cybersecurity Application Based on CARDIAN. *Future Internet* **2024**, *16*, 10. https://doi.org/10.3390/fi16010010

Academic Editors: Mario Di Mauro and Wei Yu

Received: 18 October 2023
Revised: 19 December 2023
Accepted: 22 December 2023
Published: 28 December 2023

1. Introduction

As we can seen in CARDIAN project [1], the CAN protocol was conceived in the 1980s by Robert Bosch Gmbh for connecting ECUs and is still the main communication protocol used in the automotive environment today [2]. The standard that regulates the protocol defines the physical level and the data-link level; the other levels of the ISO/OSI model are therefore established by the network designer. The type of communication is serial and asynchronous; furthermore, it allows multi-master type communication. The communication system is of a differential type, and based on the configuration, the dominant or recessive bit level can be chosen; in fact, we speak of wired and coding when the dominant bit is 0 and wired or decoding when the dominant bit is 1. This type of coding occurs by connecting the various nodes of the network in parallel through ports.

The communication protocol is based on sending messages, which will always be equipped with an ID field and which can be equipped with a data field. The ID field intrinsically contains the concept of priority, as it is the first part of the message to be transmitted, depending on the coding installed on the bus; a higher (wired or) or lower (wired and) ID may have a higher priority than another. Each ECU can participate in the communication both in transmission and in reception. In the event that two messages are produced simultaneously, the message with the highest priority will be sent, while the "conflicting" message will be blocked and sent as soon as the bus is free again.

The CAN architecture was designed to be lightweight and robust, unencrypted, unsegmented and authentication-free, so that CAN messages can flow freely to and from each ECU.

In each vehicle, there are 2 CAN buses: one at high speed, with a baud rate ranging from 40 kbit/s up to 1 Mbit/s, used for communication between the nodes critical for the safety of the vehicle and the driver, and for command transmission, and one at low speed, with a baud rate ranging from 40 kbit/s to 125 kbit/s, used for all other ECUs.

Cars and light vehicles must comply with the SAE J1979 standard [3], which defines the transmission and decoding of the parameters necessary for vehicle diagnostic operations. Heavy vehicles (buses, trucks, etc.) must, however, comply with the SAE J1939 protocol [4]. These standards define the communication protocol, the connector to use, the data format and their decoding using dedicated dbc files.

The integration of services and functions inside a car has introduced the presence of numerous attack surfaces [5]. These interfaces can be divided according to their access point and according to their range of action:

- Physical access point: access occurs by connecting a device directly to the CAN network; this can occur via a USB port connected to the infotainment system, by accessing the OBD port, etc.
- Short-range access points: access occurs via a connection with a device that can be located up to a maximum of 300 m away.
- Long-range access points: access occurs via connection to the Internet; the device connected to the network can be attacked, and controlled, by devices very distant from it.

A connected vehicle can be connected to the Internet to provide services linked to the manufacturer's proprietary platform or third-party platforms. Based on what was previously stated, the CAN bus presents numerous vulnerabilities [6], in particular:

- Due to the lack of authentication, every device connected to the bus can transmit and receive all data on the CAN bus. Given its nature, it is not able to prevent unauthorized devices from connecting to the bus and sending harmful messages to all control units. By accessing the bus, hackers can send malicious messages to any ECU on the network. Security in this context is only guaranteed by the lack of documentation: hackers must dedicate time and resources to decode the CAN protocol before they can launch malicious attacks on a particular vehicle.
- All control units are connected to the same network. This feature is one of the main reasons why this type of protocol has been successful for automotive networks, as it allows to considerably reduce the wiring necessary for point-to-point connections between the various subsystems. However, this implies that a component dealing with, for example, infotainment is able to communicate with subsystems critical to the safety of the vehicle and the driver. Some manufacturers are starting to segment the network to separate safety-critical systems. As far as design is concerned, however, cross-communication between safety-critical and non-safety-critical systems is still widely used.
- CAN was designed in the 1980s to be light and robust, when car hacking was not a reality and the computing capabilities of the various control units were not particularly developed; therefore, adding some type of encryption would only slow down the sending and decoding of messages, potentially leading to the clogging of the network. However, because CAN traffic is not encrypted, it can easily be intercepted, altered, modified and reproduced. In Table 1, it is possible to see a summary of the presented scenario.

Table 1. Automotive vulnerabilities.

Vulnerability	Consequences
Lack of authentication	Any device can access the network
Lack of segmentation	Access to critical ECUs
Lack of encryption	Intercepted and altered messages

2. Background on Cybersecurity in Automotives: Methods and Related Works

The problem of safety in the automotive sector has emerged in recent years, and several research institutes have developed various techniques aimed at providing adequate protection for the CAN bus present on board vehicles. The main techniques analyzed are as follows:

2.1. Network Segmentation

The network segmentation mechanism [7] is the most direct protection system. Using this technique, it is possible to establish a priori which components of the system must participate in the communication of a given subnetwork; any attack is thus limited to a particular area. The connection between the subnetworks is managed through particular ECUs that act as gateways. This protection model is currently used on commercial vehicles as it is very simple to implement. This system is not sufficient to guarantee the protection of the vehicle when the compromised node actually turns out to be the gateway. Furthermore, this type of network configuration makes the maintenance of the network itself more expensive and difficult.

2.2. Encryption

Various encryption systems have been developed [8] which are not particularly computationally expensive. Numerous companies, both automakers and otherwise, have developed their own encryption techniques. Unfortunately, it has been demonstrated that these can be easily bypassed by potential attackers. The main problem with using this technique arises from the fact that, in order to be effectively implemented, the data field of the message must have a fixed width. This problem can be solved by separating the messages of larger width into various messages, but this solution can only be used if the data traffic on the bus is very low, in complete contrast to recent technological developments. The use of this technique is also linked to the use of ECUs equipped with adequate calculation capacity for the generation and management of dynamic decryption keys; otherwise, if we consider the life time of a commercial vehicle, it is highly possible that the pre-installed static decryption key will be exposed.

2.3. Authentication

Currently, there are authentication systems [9] that allow response times for authentication of around 50 us. These are based on the use of "trust groups", which are in turn equipped with secret keys. This method is particularly effective when these groups are few compared to the number of ECUs. The protection mechanism consists in sending an authentication message after each transmitted frame, the data traffic is therefore always doubled. The main problem, similarly to what happens with the segmentation method, arises when one of the ECUs belonging to one of the trust groups is compromised.

2.4. Intrusion Detection Systems

To overcome the problems of the protection systems described previously, intrusion detection systems (IDSs) can be used [10]. The installation of an IDS does not compromise data traffic on the bus nor does it require the modification of the various CAN controllers, necessary for the use of encryption and authentication techniques as well as the structure of the network itself. The main intrusion detection methods can be divided into two categories:

- Signature-based: the system identifies the presence of attacks through the use of databases in which different types of attacks are present. It is clear that this method is not particularly effective if the type of attack has not been previously described.
- Anomaly-based: the system is responsible for analyzing the behavior of the network and is able to recognize any deviations from normal behavior. Unlike signature-based approaches, it can easily identify attacks that are not yet known.

In the next section the main anomaly-based methods will be described.

3. Intrusion Detection Systems in Automotives

The design and implementation of an intrusion detection system must take into account the following critical issues [10]:

- Limited resources: the ECUs inside a vehicle are typically equipped with small memories, little computational power and limited bandwidth.
- Real-time operation: CAN messages are generated and transmitted in real time; delaying a message and generating queues can therefore become critical during communication. The messages must be processed by the other ECUs as soon as they are received, in order to guarantee the correct functioning of every part of the vehicle.
- CAN traffic management: the CAN traffic management protocol is different from typical internet communication protocols; for example, CAN messages always have a broadcast type operation.
- Unstable connections: since these are moving systems, they could move in areas with limited or even no connection to the network. The use of IDS systems connected to the network must take into account the fact that such areas may exist, guaranteeing smooth operation offline, even for long periods of time.
- Weight, dimensions and cost: the installation of an IDS can affect the topology of the network to which it is connected; it must therefore be chosen appropriately so as not to require an excessively expensive modification of the network.

3.1. Main Cyberattack Techniques

Starting from the analysis of the CAN bus, the possible attack types can be divided into two categories [11]:

- Attacks based on transmission frequency;
- Attacks based on message content.

Frequency-based attacks generally aim to compromise the operation of the entire bus, effectively causing it to stop functioning. The most obvious type of frequency-based attack is DoS (denial of service) [12]. This technique consists in sending messages with the highest priority and zero content in rapid succession. In this way, all other messages will be delayed, causing malfunctions in the ECUs, which can even go offline and stop actively participating in the communication. A very similar type of attack is the fuzzy attack, which consists in sending numerous messages with random IDs and contents. This type of attack still leads to the compromise of the bus, but is more difficult to identify.

Attacks based on the content of the message do not aim to destroy the bus, but to compromise the nodes. This type of attack can be used to disable some peripherals (such as the brakes), suspend the sending of data, or report non-existent faults. Furthermore, on some types of vehicles, it can allow an attacker, connected remotely, to take control of the vehicle itself. This type of attack typically occurs by sending messages structured in an identical manner to those typically present on the bus; these can be generated by external devices connected to the bus, or by nodes of which it is possible to take control.

This category also includes man-in-the-middle attacks, which request a great deal of information from the vehicle via remote frames, and then send that information to some externally connected device.

This type of attack can be recognized directly by interpreting the messages circulating on the bus (Table 2).

Table 2. Main cyberattack categories.

Type of Attack	Consequences	Detection
DoS	Network saturation	Analysis of the data present on the bus
Fuzzy	Processing of altered data	Behavioral data analysis
Man-in-the-middle	Theft of sensitive data	Data interpretation

3.2. Intrusion Detection Techniques

As explained in the previous section, the main attack identification techniques can be divided into signature-based and anomaly-based. Signature-based techniques are ineffective against unknown attacks; therefore, we proceed to describe different approaches for anomaly detection [13–15].

An anomaly-based system observes the behavior of the system in real time, and when it deviates from the expected normal function, it triggers an alarm signal. It is clear that in order to create an adequately effective system, the training phase of the system becomes critical (Table 3). In order to create normal operating profiles, the following approaches are typically used [16,17].

Table 3. Approaches for IDS development.

Approach	Advantages	Disadvantages
Data frequency-based approach	Simplicity of implementation	Cannot identify all types of attacks
Machine learning approach	Effectiveness in identifying anomalous situations	Computational complexity
Statistical approach	High accuracy	Does not recognize replay-type attacks
Approach based on the electrical analysis of the CAN bus	Direct analysis of the operating state of the bus	High levels of False Positives

3.3. Bayesian Network Technique

A further method for identifying attacks involves the use of probabilistic approaches [18,19]. This method is based on the use of machine learning techniques aimed at training a system that is able to classify the attack state on a system by providing the probability that it being under investigation is the result of an attack. The technique considered involves the use of Bayesian networks.

A Bayesian network is a graphical model that represents the dependency relationship between a given number of variables [20].

Its structure can be represented through the use of a directed graph: a graph equipped with directed nodes and arcs. This is a feedforward structure: starting from any node and following the direction of the arcs, it is not possible to return to the node itself or to nodes whose hierarchical level is higher. Each node represents a variable whose possible state, unique and mutually exclusive with respect to the others, is associated with a certain probability value, while the arcs between nodes indicate a dependence relationship between the variables represented by them: if two nodes are not directly connected, they are conditionally independent [21].

TPCs are associated with nodes that have parents, i.e., that are connected to at least one edge that points to them: tables containing the probabilities of the values of the node conditioned by the possible combinations of values of the parent nodes. In those nodes that do not have parents, there will instead be a prior probability table which will simply express the probabilities for each value of the node variable.

Inference on Bayesian networks is a statistical inference process in which Bayes' theorem is exploited to estimate and update the probability of a hypothesis as soon as new evidence is collected. The theorem allows us to express a conditional probability in terms of the opposite conditional probability, weighing it appropriately [22].

4. Case Study

In the previous sections, the main vulnerabilities in the automotive sector have been shown, as well as the main related countermeasures; the main techniques for intrusion detection have also been analyzed. Furthermore, it has recently been demonstrated how

easy and possible it is to break in and inject malicious messages into the CAN bus of a vehicle, accessing it remotely through the infotainment system.

This paper will describe the creation of an intrusion detection system based on the use of a Bayesian network, which, connected directly to the CAN bus, controls the flow of messages in order to detect anomalies and intrusions. The Bayesian network was structured starting from the analysis of the automotive domain ontology.

The data used for training and validation of the network are real data, acquired from the CAN bus of a heavy vehicle, in SAE J1939 format. This dataset was modified by inserting attacks at regular time intervals, as described in [23].

4.1. Automotive Domain Ontology

For the representation of a domain of interest, ontologies are increasingly used. This type of representation allows us to describe all the relevant entities of a particular domain of interest, and the relationships between them, through the use of a data structure.

For the automotive domain, one of the most used ontologies is the one built by the Automotive Ontology Working Group [24] (Figure 1), a consortium of companies and individuals whose aim is to develop this ontology with the aim of guaranteeing an increasingly better interoperability of data within the automotive industry and to create a place where researchers and professionals can collaborate to advance developments in this sector.

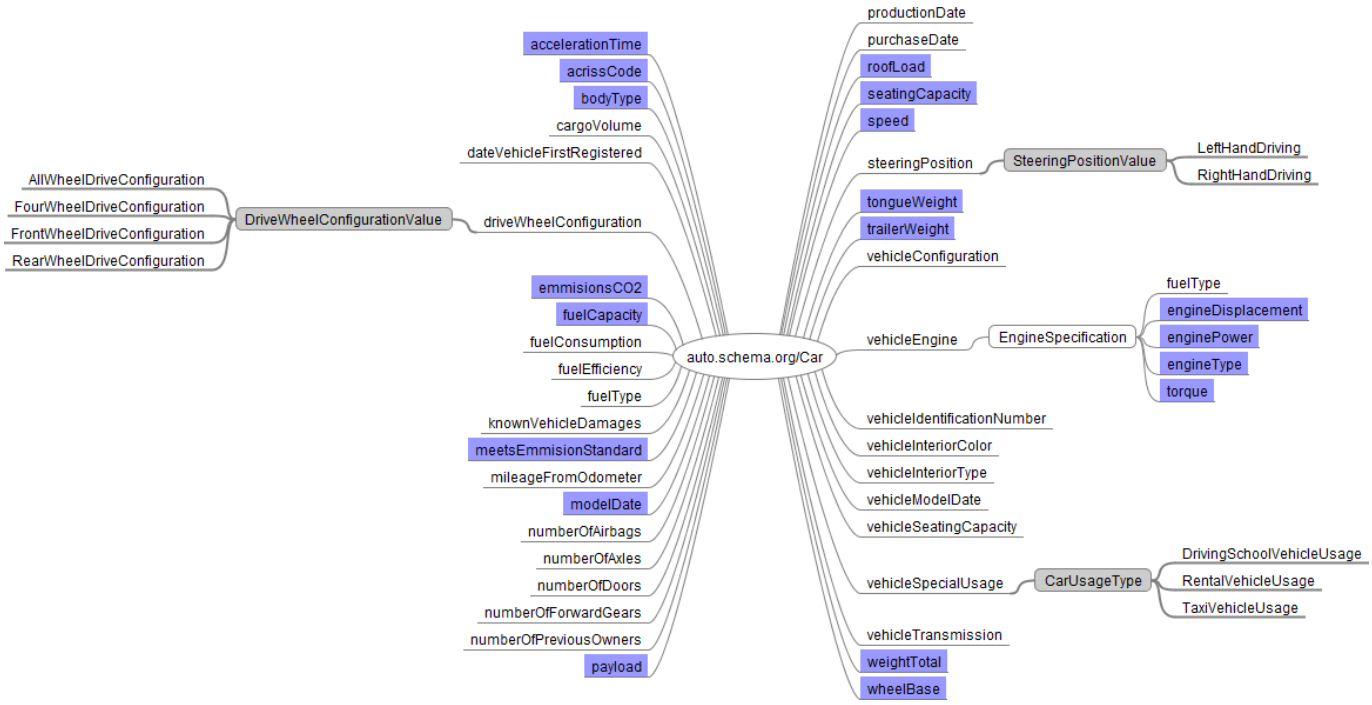

Figure 1. Automotive domain ontology.

The ontology used is represented in the following figure. From it, it is clear that the car object is characterized by numerous properties, listed in the table below. The data which, in turn, have properties are called types.

4.2. Three-Step Algorithm

The intrusion detection methodology consists of three phases and is described in patents No. 102021000009548 and No. EP 22168635.5 (see Figure 2).

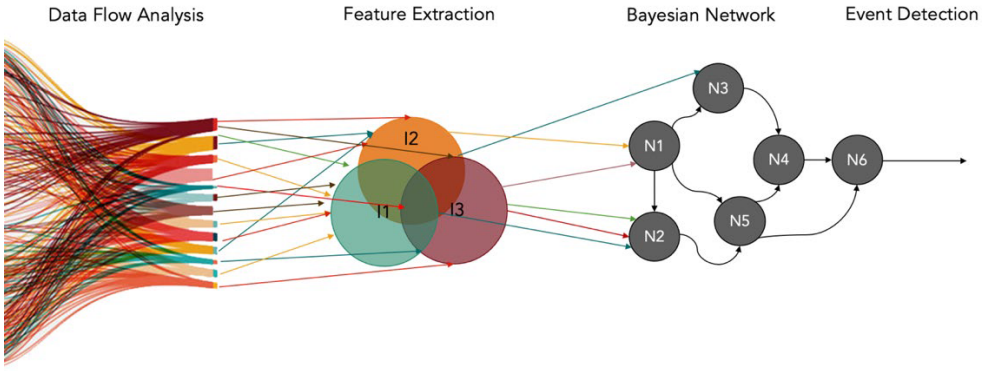

Figure 2. Three-step algorithm.

The first phase includes the acquisition of data within time windows of standard duration. Within these windows, it may happen that multiple messages containing the same information may be repeated; in this case, the data that will be forwarded to the second step will be the result of the averaging operation between these. On the contrary, if the data are not present within the time window, they will be replaced by the value 0.

The second phase concerns the identification of the driving scenario (e.g., urban driving, motorway, etc.). The sequence of data produced by the first phase is analyzed and compared with the predefined scenarios using the Jaccard index. At that point, the scenario that will be used as a reference will be the one whose Jaccard index has the highest score.

The Jaccard index, also known as the Jaccard coefficient, is defined, for two sets called A and B, by the following equation:

$$J = \frac{|A \cap B|}{|A \cup B|} \tag{1}$$

The identification of the scenario allows the selection of the Bayesian network to be used, and more precisely, the values contained in the a priori and conditional probability tables, which describe each node of the network.

The third phase involves the use of the Bayesian network. The data produced by the first phase are fed to the Bayesian network, which will provide as output the probability that this data set was determined by an attack.

4.3. System Architecture

The system has been developed in such a way as to have direct access to the vehicle's CAN bus like a normal ECU (see Figure 3), which will be able to signal the presence of an attack by sending a file in json format via MQTT protocol to the node to which the device is connected.

From a hardware point of view, the device is made up of the Minized board produced by Avnet. The board's SoC consists of an ARM-Cortex-A9 single core processor and an Artix 7 FPGA. A CAN controller is also directly connected to this SoC, making it necessary to install only the CAN transceiver. In order to connect it, it is necessary to configure, using the Vivado software, version 2020.1, the enabling of the CAN controller peripheral integrated into the System on Chip, as well as its connection to the pins of the Pmod2 port by modifying the xdc file associated with the project. The CAN controller used in this phase is the Transceiver SN65HVD230. This device interfaces directly to the CAN bus via the CAN-H and CAN-L lines, and takes care of the conversion between the differential signal coming from the bus and the serial one used by the CAN controller.

The AXI bus is also connected to the CPU of the device, which is used for the connection and transfer of data between the CPU and the FPGA. The following figure highlights the AXI peripheral, which is connected directly to the CPU unit of the Zynq SoC (Advanced Micro Devices, Inc., Santa Clara, CA, USA).

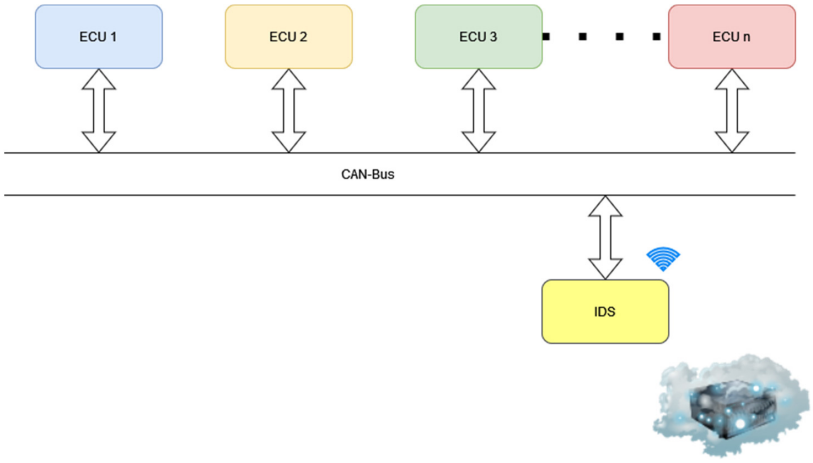

Figure 3. System architecture.

The Bayesian network will be connected downstream of the AXI peripheral, using the AXI GPIO component (see Figure 4). For each of these components, the manufacturer Xilinx makes IP available already integrated into Vivado.

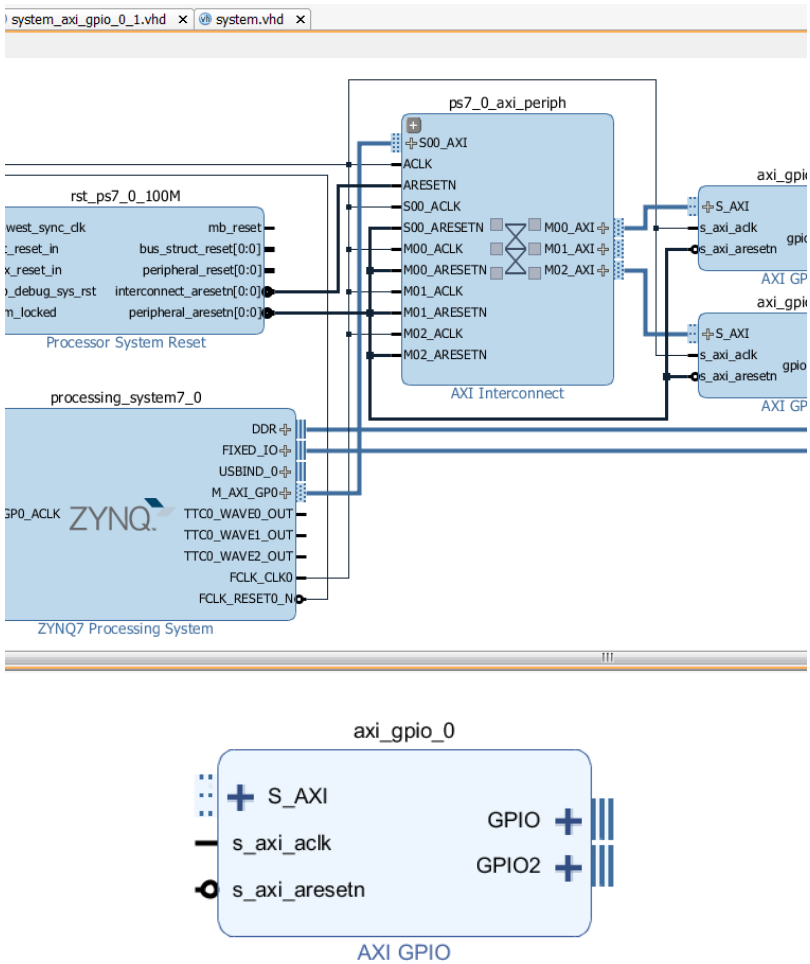

Figure 4. AXI bus connection.

The system created in hardware, including a Bayesian network connected to the AXI bus and including the enabling of the CAN controller, is used as the basis for compiling the Petalinux operating system. Petalinux is a particular Linux distribution optimized for embedded systems and systems based on System on Chip containing FPGAs.

Using the Vivado software, the hardware file containing the part relating to the device peripherals and the bitstream relating to the Bayesian network are exported.

Subsequently, the operating system is compiled and configured, which is finally installed on the board. The programs used for the acquisition of CAN messages, for communication with the AXI interface and for sending messages via MQTT, are written in C, with the integration of the Mosquitto and cantools libraries.

There are two scripts in use, and they are executed in parallel; the management of active threads is delegated to the operating system.

4.4. Deployment of Bayesian Network

The Bayesian network was built after a careful analysis of approximately 300 h of acquisition, distributed across four different vehicles, operating in the same conditions and compatible with the SAE J1939 protocol. From the analysis of the data produced, it emerged that only a small number of parameters are suitable for analysis, as they are sent on the bus at intervals of less than 10 ms. Furthermore, only data presenting at least one variation during the driving and, therefore, acquisition period were used.

The nodes of the network, and the definition of the classes to which the data belong, have been described in the previous section (Figure 5).

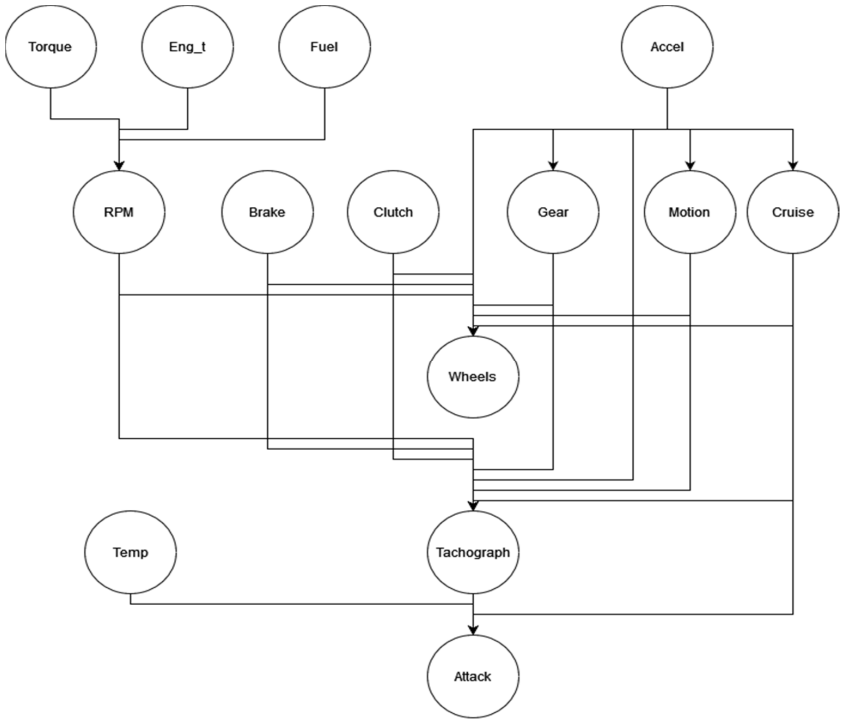

Figure 5. Structure of the Bayesian network.

The following table highlights, explicitly and for each of the nodes, their parent nodes, i.e., those nodes whose state inevitably conditions the state of the node itself. The Bayesian network was then implemented using Vivado software in VHDL language.

The hardware structure of a Bayesian network [25] is very simple; each node is composed of the following:

- a memory unit containing the a priori and conditional probability distributions;
- a multiplier, already described by the IPs present within Vivado, for calculating probabilistic inference.

Nodes that do not have parents implement a ROM memory, so it is possible to treat the input data as an address. However, the nodes which do have parents carry out the calculation of the output probability by applying the previously mentioned Bayes' theorem.

As can be seen from Figure 6, each node receives as input a memory address of variable width, depending on the node, and a probability represented with a 16-bit fixed point coding. Following each multiplication, in order to guarantee the uniformity of the data, a truncation is carried out, taking only the 16 most significant bits. The final attack detection node contains a comparator, so as to provide an affirmed output only if it exceeds the threshold of 0.5.

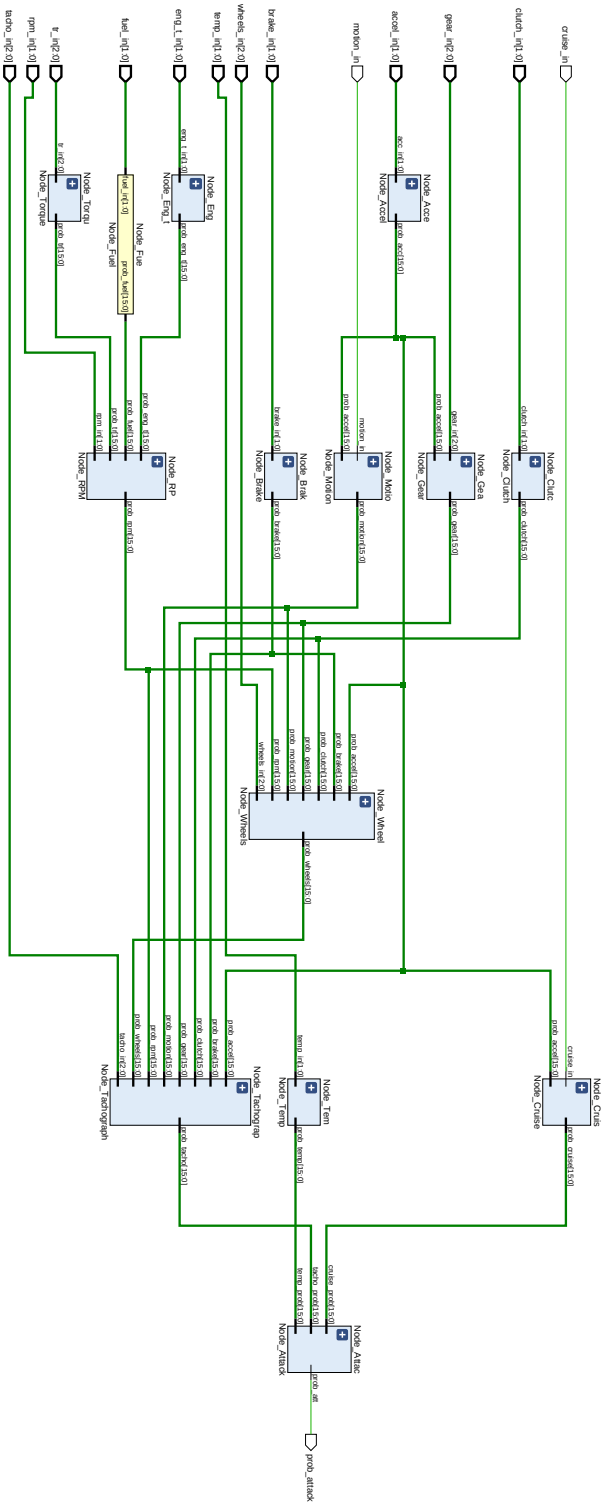

Figure 6. Hardware deployment of the Bayesian network.

5. Experimental Results

In this section, the testing phase will be described and the results obtained will be highlighted.

In the previous section, it was shown how the entire system was built in order to evaluate the performance of a Bayesian network in recognizing intrusions starting from a sequence of CAN messages present on the bus. These messages, in the situation examined, contain all the information relating to the status of the vehicle.

The calculation of the prior and conditional probabilities of each node was carried out in Python, using the Pomegranate library.

The Pomegranate module deals with implementing probabilistic models from the simplest, such as probability distributions, to the most complex, such as Bayesian networks. This module was developed on the idea that all probabilistic models can be represented as probability distributions. This vision allows the various models to be developed in a light and flexible way. Furthermore, each model has numerous integrated features, including various types of learning. It allows us to implement Bayesian networks that use different types of distributions on each feature. Finally, with a few commands, starting from a known number of pieces of evidence, it is able to determine the posterior probability associated with all the nodes of the network.

The probability distributions were determined from the acquired and pre-processed data. The training dataset was constructed as described in [24]. Four types of attacks have been defined:

- DoS attack: messages with CAN ID composed of only zeros and a random data field are injected every 0.3 milliseconds;
- Fuzzy attack: messages with CAN ID composed of random ID and random data field are injected every 0.5 milliseconds;
- Gear attack: the message with ID and SPN relating to the current gear is injected, with a random data field every millisecond;
- RPM attack: the message with ID and SPN relating to the current RPMs is injected, with a random data field every millisecond.

For each interval containing one or more malicious messages, the attack value is set to 1; otherwise, it is set to 0. The number of messages acquired before data entry is 10,900,724; from these data, another four datasets were generated, one for each type of attack. Each data set is composed of the first 50% (5,450,362 messages in 150 h) of normal operation; the remaining part presents the injection of attacks. The network was then trained with a total of 20,058,781 message frames, of which 19,781,678 were malicious. Once the network was trained, it was tested first by using a real dataset lasting 1 h, and then with datasets containing the previously described attacks, of the same duration. The tests were carried out by sending messages directly to the CAN bus of the device using a USB-CAN converter.

The first important parameter to consider is the system response time. The timestamps of the instant in which the last message of the time window is sent and the timestamp in which the processing produces the json file were analyzed. To classify the results obtained, the following cases were distinguished:

- True Positives (TP): presence of an attack and correct reporting of this;
- True Negatives (TN): absence of attacks and recognition of this absence;
- False Positives (FP): absence of attacks but reports of attacks;
- False Negatives (FN): presence of an attack but no recognition of it.

Using the confusion matrix, the reference parameters such as precision, recall and f1-score were calculated.

The first case analyzed, a dataset without malicious messages, produced the following results: (Tables 4–12)

Table 4. Lack of attacks.

Attacks/Detected	*Yes*	*Not*
Yes	0	63
No	0	3678

The second case, a dataset hacked by DoS attack, produced the following results:

Table 5. DoS attack.

Attacks/Detected	*Yes*	*Not*
Yes	17,501	230
No	22	1831

The third case, a dataset hacked by fuzzy attack, produced the following results:

Table 6. Fuzzy attack.

Attacks/Detected	*Yes*	*Not*
Yes	27,311	132
No	42	1736

The fourth case, a dataset hacked by Gear attack, produced the following results:

Table 7. Gear attack.

Attacks/Detected	*Yes*	*Not*
Yes	50,121	111
No	34	1735

The fifth case, a dataset hacked by RPM attack, produced the following results:

Table 8. RPM attack.

Attacks/Detected	*Yes*	*Not*
Yes	50,117	110
No	35	1740

The results obtained showed the following performances in terms of precision, recall and f1-score:

Table 9. Precision.

Type of Attack	Value (Percentage)
No attack	ND
DoS	0.998
Fuzzy	0.998
Gear	0.999
RPM	0.999

Table 10. Recall.

Type of Attack	Value (Percentage)
No attack	ND
DoS	0.987
Fuzzy	0.995
Gear	0.997
RPM	0.997

Table 11. F1-Score.

Type of Attack	Value (Percentage)
No attack	ND
DoS	0.992
Fuzzy	0.996
Gear	0.998
RPM	0.998

Table 12. Response time.

Type of Attack	Response Time (Milliseconds)
No attack	91 ms
DoS	92 ms
Fuzzy	89 ms
Gear	85 ms
RPM	93 ms

6. Conclusions

The developed system shows how it is possible to use probabilistic approaches based on Bayesian networks for the recognition of cyber attacks present on the CAN bus; in fact, approximately 99% of attacks were correctly classified. This was possible thanks to the high number of samples, which allowed the correct training of the network. The averaging operation carried out during data collection made it possible to create an effective index of the status of the vehicle in a given frame, and the high presence of attacks has guaranteed the entire system to be able to respond correctly to numerous types of attacks. The application of domain ontologies allowed an optimal description of the vehicle characteristics, avoiding the presence of nodes that could have led the system to never converge to an optimal solution. The response times of the system are very low thanks to the use of a HW accelerator built on FPGA, allowing real-time operation. The structure of the network obtained, implemented on FPGA, shows the limited consumption of resources, thus also managing to reduce energy consumption. Future developments include, in addition to testing in a controlled real environment, also the implementation of contextual datasets that can better react to the conditions surrounding the vehicle. Once it has been verified that the system is also effective in this environment, it will be possible to develop strategies for the exclusion of malicious nodes or for the avoidance of malicious data flows.

7. Patents

This research work is based on Italian patent No. 102021000009548, registered on 14 April 2021, and International Patent pending No. EP 22168635.5, registered on 14 April 2022.

Author Contributions: Conceptualization, F.P., E.A.A., S.C. and E.S.; methodology, F.P., E.A.A., S.C. and E.S.; formal analysis, E.A.A.; investigation, F.P.; resources, E.S.; data curation, F.P.; writing—original draft preparation, E.A.A. and E.S.; writing—review and editing, F.P. and S.C.; visualization, E.A.A., S.C. and E.S.; supervision, F.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data are not publicly available due to patent protection.

Acknowledgments: The authors want to thank MinervaS and the University of Salerno for the support.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Pascale, F.; Adinolfi, E.A.; Avagliano, M.; Bellacosa, E.; Coppola, S.; Santonicola, E. CARDIAN: A Context Aware Cybersecurity System for Real Time Diagnostic Intrusion Detection Using a Probabilistic Approach with Bayesian Network. In Proceedings of the 2022 6th International Conference on System Reliability and Safety (ICSRS), Venice, Italy, 23–25 November 2022; pp. 424–429. [CrossRef]
2. *ISO 11898-1:2015*; Road Vehicles—Controller Area Network (CAN). ISO: Geneva, Switzerland, 2015.
3. *SAE J1979*; E/E Diagnostic Test Modes. SAE: Warrendale, PA, USA, 1979.
4. *SAE J1939*; Recommended Practice for a Serial Control & Communications Vehicle Network. SAE: Warrendale, PA, USA, 1939.
5. Chhawri, S.; Lane, G.R.; Tarnutzer, S.; Tasky, T. Smart Vehicles, Automotive Cyber Security & Software Safety Applied To Leader-Follower (Lf) and Autonomous Convoy Operations (Aco). In Proceedings of the 2017 Ndia Ground Vehicle Systems Engineering and Technology Symposium, Novi, MI, USA, 8–10 August 2017.
6. Di Natale, M. *Understanding and Using the Controller Area Network Communication Protocol*; Springer: Berlin/Heidelberg, Germany, 2008; p. 223.
7. Török, Á.; Szalay, Z.; Sághi, B. New Aspects of Integrity Levels in Automotive Industry-Cybersecurity of Automated Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 383–391. [CrossRef]
8. Macher, G.; Armengaud, E.; Brenner, E.; Kreiner, C. Threat and Risk Assessment Methodologies in the Automotive Domain. *Procedia Comput. Sci.* **2016**, *83*, 1288–1294. [CrossRef]
9. Liem, C.; Murdock, D.; Williams, A.; Soukup, M. Highly Available, Self-Defending, and Malicious Fault-Tolerant Systems for Automotive Cybersecurity. In Proceedings of the 2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C), Sofia, Bulgaria, 22–26 July 2019; pp. 24–27. [CrossRef]
10. Wei, P.; Wang, B.; Dai, X.; Li, L.; He, F. A novel intrusion detection model for the CAN bus packet of in-vehicle network based on attention mechanism and autoencoder. *Digit. Commun. Netw.* **2022**, *9*, 14–21. [CrossRef]
11. Fakhfakh, F.; Tounsi, M.; Mosbah, M. Cybersecurity attacks on CAN bus based vehicles: A review and open challenges. *Library Hi Tech* **2021**, *40*, 1179–1203. [CrossRef]
12. Martínez-Cruz, A.; Ramírez-Gutiérrez, K.A.; Feregrino-Uribe, C.; Morales-Reyes, A. Security on in-vehicle communication protocols: Issues, challenges, and future research directions. *Comput. Commun.* **2021**, *180*, 1–20. [CrossRef]
13. Karopoulos, G.; Kambourakis, G.; Chatzoglou, E.; Her-nández-Ramos, J.L.; Kouliaridis, V. Demystifying In-Vehicle Intrusion Detection Systems: A Survey of Surveys and a Meta-Taxonomy. *Electronics* **2022**, *11*, 1072. [CrossRef]
14. Young, C.; Olufowobi, H.; Bloom, G.; Zambreno, J. Automotive Intrusion Detection Based on Constant CAN Message Frequencies Across Vehicle Driving Modes. In Proceedings of the ACM Workshop on Automotive Cyber-Security (AutoSec '19). Association for Computing Machinery, New York, NY, USA, 27 March 2019; pp. 9–14. [CrossRef]
15. Lokman, S.-F.; Othman, A.T.; Abu-Bakar, M.-H. Intrusion detection system for automotive Controller Area Network (CAN) bus system: A review. *EURASIP J. Wirel. Commun. Netw.* **2019**, *2019*, 184. [CrossRef]
16. Bozdal, M.; Samie, M.; Aslam, S.; Jennions, I. Evaluation of CAN Bus Security Challenges. *Sensors* **2020**, *20*, 2364. [CrossRef] [PubMed]
17. Choi, W.; Joo, K.; Jo, H.J.; Park, M.C.; Lee, D.H. VoltageIDS: Low-Level Communication Characteristics for Automotive Intrusion Detection System. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 2114–2129. [CrossRef]
18. Rahim, A.; Rahman, A.; Rahman, M.; Asyhari, A.T.; Alam Bhuiyan, Z.; Ramasamy, D. Evolution of IoT-enabled connectivity and applications in automotive industry: A review. *Veh. Commun.* **2021**, *27*, 100285. [CrossRef]
19. Pascale, F.; Adinolfi, E.A.; Coppola, S.; Santonicola, E. Cybersecurity in Automotive: An Intrusion Detection System in Connected Vehicles. *Electronics* **2021**, *10*, 1765. [CrossRef]
20. Lombardi, M.; Pascale, F.; Santaniello, D. Two-Step Algorithm to Detect Cyber-Attack Over the Can-Bus: A Preliminary Case Study in Connected Vehicles. *ASCE-ASME J. Risk Uncertain. Part B* **2022**, *8*, 031105. [CrossRef]
21. Buczacki, A.; Piątek, P. Proposal for an Integrated Framework for Electronic Control Unit Design in the Automotive Industry. *Energies* **2021**, *14*, 3816. [CrossRef]
22. Thantharate, P.; Thantharate, A.; Kulkarni, A. GREENSKY: A Fair Energy-Aware Optimization Model for UAVs in Next-Generation Wireless Networks. *Green Energy Intell. Transp.* **2023**, 100130. [CrossRef]

23. Automotive Ontology Group. (n.d.). Automotive Ontology Domain. Available online: https://Schema.Org/Docs/Automotive.Html (accessed on 12 December 2023).
24. Seo, E.; Song, H.M.; Kim, H.K. GIDS: GAN based Intrusion Detection System for In-Vehicle Network. In Proceedings of the 2018 16th Annual Conference on Privacy, Security and Trust (PST), Belfast, Ireland, 28–30 August 2018; IEEE: Piscataway, NJ, USA, 2018.
25. Zermani, S.; Dezan, C.; Chenini, H.; Diguet, J.-P.; Euler, R. FPGA implementa-tion of Bayesian network inference for an embedded diagnosis. In Proceedings of the 2015 IEEE Conference on Prognostics and Health Management (PHM), Austin, TX, USA, 22–25 June 2015; pp. 1–10. [CrossRef]