Security of In-Vehicle Networks: A Survey

Group-13:-

Mahir Thumar, Prince Viradiya, Aditya Rajpara, Pooja Sheth, Om Sarvaiya, Dhairyashil Dhingani

Abstract— **This comprehensive guide covers every safety measure related to in-car networks. This research focuses on weaknesses in the widely used Controller Area Network (CAN) protocol. Additionally, it examines alternatives such as LIN and FlexRay and proposes Vehicular Ad Hoc Networks (VANETs) as a potential solution. In addition, it looks at security holes in the CAN protocol, discusses defenses against different types of attacks, and recommends conducting hardware inspections to reduce the likelihood of hardware Trojan (HT) assaults. By integrating a Security Monitoring System for the CAN protocol, it also reduces computational complexity and simplifies the authentication process. Implementing ID-Anonymization for Secure CAN (IA-CAN) is an additional suggested defense against Denial-of-Service assaults that enhances the security of communication within automobiles. An integrated approach that fortifies cybersecurity in automobile networks holistically combines firewalls and secure gateways.**

Index Terms— **CAN- Controller Area Network, IDS- Intrusion Detection System DOS- Denial of Service, ECU- Electronic Control Units, AES- Advanced Encryption Standard ,HIDS- Host-based Intrusion Detection System ,NIDS- Network-based Intrusion Detection System , LIN- Local Interconnect Network , HT – Hardware Trojan ,MAC- Message Authentication Code ,CRC- Cyclic Redundancy Check.**

I. Introduction

To improve driving comfort and safety, modern vehicles are outfitted with Electronic Control Units (ECU) to control the electrical systems [1]. ECUs control most of the car's functions, such as engine control, airbag deployment, and anti-lock braking. ECUs must have a reliable communication network in order to drive safely. The primary mode of in-vehicle communication Controller Area Network (CAN) is the protocol. Its well-known benefits include high electrical resistance.

The ability to self-diagnose and repair errors, as well as interference, makes the CAN bus suitable for the automotive industry. CAN is vulnerable to attacks despite its resistance to electrical noise and some security features. As system security becomes a major concern, extensive research on CAN vulnerabilities and potential solutions is being conducted. Some of these studies were successful in their experimental attacks on commercial vehicles. Although most attacks are carried out through physical access to the bus, wireless attacks are on the rise. With the new wireless technology Wireless attack surface will expand due to interfaces such as vehicle-to-vehicle and vehicle-to-infrastructure.

II. Background

A. Electronic Control Unit

Engine electronic control unit (ECU) receives the signals from the sensors and gets the current engine status. Then depending on the signals, the ECU calculates fuel injection rate, injection timing and quantity. Accordingly, the actuators which are driven by ECU carry out better fuel control.

B. Controller Area Network

The CAN protocol is a multi-master communication interface that was developed for in-vehicle communication. It is a broadcast network with the capability of

up to one megabit per second (bps). The CAN bus has a high immunity to electrical interference, is simple to wire, and can self-diagnose and repair errors. The network's distributed architecture simplifies maintenance and lowers overall system costs [2] CAN and ECU are backbone of in-car network communication model that can be explained with following points:

- Communication Backbone: CAN is the communication backbone that allows different ECUs to exchange data in real-time, allowing for coordinated control and decision-making.
- Data Exchange: ECUs use the CAN protocol to transmit and receive data, ensuring synchronized functioning across diverse vehicle systems.
- Efficient Collaboration: By allowing ECUs to collaborate for optimal vehicle performance, CAN integration with ECUs enhances the efficiency of automotive systems.

When it comes to in-vehicle communication protocols, Controller Area Network (CAN) has been the most popular option for guaranteeing security and reliability. Real-Time Communication Efficiency is one of the many variables that enable this technique. CAN is widely recognized for its deterministic and real-time capabilities, which are essential for time-sensitive applications in automobiles that demand prompt reaction. Fault Tolerance and Resilience: CAN has fault-tolerant features that enable uninterrupted operation in the case of malfunctions or disruptions, hence boosting the overall resilience of in-car communication. Vehicle makers can use CAN because it is a cost-effective option without compromising functionality since it strikes a cost-performance balance. A well-established ecosystem consisting of hardware and specific protocols supports CAN, facilitating deployment and compatibility.

C. Security shortcomings of CAN protocol in perspective of in-vehicle security

When the CAN bus was first designed security was not a top priority. It was only used to connect a few ECUs and was not visible to the end user. However, the automobile industry has evolved significantly, and there are now dozens of ECUs connected, and it is mandated by law that the bus should be accessible for diagnostic purposes. [3]

Even though CAN has many security features, it is still vulnerable to attacks. The primary issue with the CAN protocol is the lack of encryption and authentication. Because there is no authentication, any unauthorised node can join the network and participate in communication. Because CAN is a broadcast network, there are no source or destination addresses, and any node can listen to any message. As a result, because the data is not encrypted, an adversary can listen in and understand it. This may cause privacy issues because modern cars collect data about drivers such as location and address book. It also allows an adversary to inject erroneous data into the system.

The CAN protocol is also susceptible to denial of service (DoS) attacks. The CAN arbitration mechanism allows higher priority nodes to speak first. If a malicious node has the highest priority and is always active, the other nodes cannot communicate due to CAN bus prioritisation.

D. Vehicular Ad Hoc Networks (VANET)

Vehicular Ad hoc Networks (VANETs) have recently garnered significant attention as a subset of mobile ad hoc networks. The central concept of VANET is to facilitate communication among vehicles and between vehicles and fixed infrastructure along the road. Numerous research initiatives have delved into this compelling and practical domain, aiming to optimize its implementation.

The primary objective of VANETs is to elevate the safety and comfort of vehicle passengers by seamlessly sharing information about traffic, road conditions, and weather among nearby vehicles. Notable applications include collision warnings, alerts for road signs, and automated toll/parking payment systems. [4] Given its ad hoc nature, VANETs operate without centralized servers, with vehicles autonomously managing the network. Communication within VANETs is categorized into two distinct types: Vehicle-to-Vehicle (V2V) Communication: In this mode, nearby vehicles engage in communication to share vital information about traffic patterns and road conditions among themselves. Vehicle-to-Infrastructure (V2I) Communication: This mode involves vehicles transmitting their gathered information to the nearest RSU. The goal is to expedite the distribution of information in a more efficient and timely manner.

III. CAN BUS OVERVIEW

In the development of a standardized CAN bus system for cars, a comprehensive understanding of the CAN bus protocol is essential. This summary outlines the foundational elements of the CAN bus protocol, with a specific emphasis on its bus architecture and the mechanisms governing data transmission within the bus.

CAN Bus was created as a multi-master message broadcast system that communicates at the maximum signaling rate of 1 Mbps. It is a serial communication bus created for the vehicle industry by the International Standardization Organization (ISO). The CAN bus is made up of a controller and a transceiver that oversee transmitting and receiving data between subsystems. It includes message filtering and status handling object and transfer layers.

Communication on the CAN bus takes place via different pair signals: CAN high and CAN low. The configuration's standard data rates are 125 kbps, 500 kbps, and 1 Mbps.[7] The CAN frame in the most recent CAN bus standard allows transfers of up to 64 B at higher speeds. The diagram below depicts a typical physical configuration of a CAN network with an arbitrary number of nodes. These nodes communicate using the two-wire CAN bus protocol, with CAN high and CAN low lines.

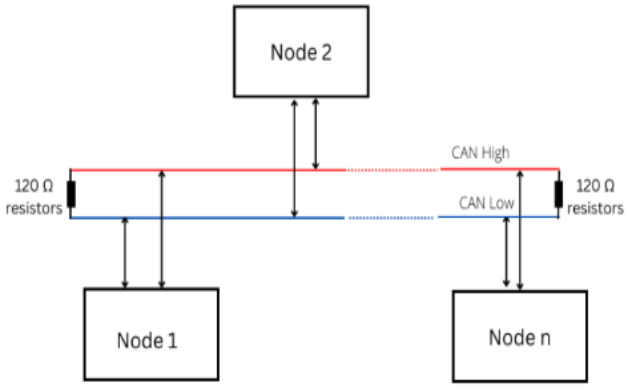


Figure -1 The CAN network [1]: This illustrates a CAN-enabled network with n-number of nodes sending messages via the bus for the vehicle's overall functionality. The nodes can be the engine, radio, brake, steering wheel, etc.

Because the CAN protocol is a broadcast network, all ECUs connected to the bus can receive signals/messages sent via the bus. Frames are the signals that are sent on the bus. These frames contain messages informing the vehicle (or system) of the upcoming operations. The length of CAN messages can be specified in two ways: standard and extended. Because of the additional 18 bits in the arbitration field, the extended CAN message format differs slightly from the standard CAN format. The image below depicts the frame format of an extended CAN message.

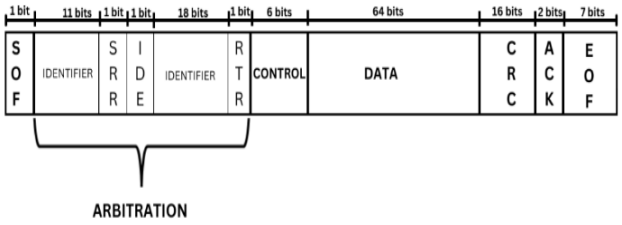


Figure 2. The frame format for an extended CAN message: This shows an example of a message packet sent via the bus. When the message is decoded, we can retrieve all of its components based on the size of each field.

A. **Features of CAN**

The following are the main features of CAN:

(1) Topology of Buses: CAN is widely used in bus topologies with two or more buses. More ECUs are linked to a communication system line.

(2) Master of Many Masters: Because each node can instantly transmit messages, it is simple to add messages to a CAN bus as needed. Messages and nodes in CAN.

(3) Transmission Right Arbitration: When two or more nodes exchange messages on a network at the same time, the gearbox right is on the CAN bus.

CAN-ID is used to arbitrate. Following the arbitration, the CAN message that contains the message with the highest priority is sent first. As a result of this, lower-priority messages are delayed in transmission until more important messages are sent.

B. **Drawbacks of CAN**

Limited Bandwidth: The limited bandwidth of CAN may hinder the implementation of robust security measures, impacting real-time performance.

No Message Authentication:CAN lacks built-in mechanisms for authenticating messages, leaving it vulnerable to unauthorized access and data tampering.

Absence of Encryption: CAN messages are transmitted in plaintext, lacking encryption, which exposes data to eavesdropping and tampering.

C. **Physical Layer Vulnerabilities:**

CAN networks, often physically accessible, are prone to tampering or manipulation by individuals with physical access to the vehicle.

Insufficient Network Segmentation: The lack of network segmentation in CAN makes it challenging to contain security breaches and limits isolation of compromised nodes.

Limited Monitoring Capabilities: Traditional CAN networks lack robust monitoring and intrusion detection capabilities, hindering real-time detection of security incidents.

Difficulty in DoS Detection: CAN is susceptible to Denial-of-Service (DoS) attacks and detecting such attacks can be challenging due to limited monitoring features.

IV. THREATS TO CAN NETWORKS

Apart from the mentioned design flaws, there are several considerations in terms of confidentiality, integrity, authenticity, availability, and non-repudiation in in-vehicle networks. These issues must be carefully considered to secure in-vehicle networks.

i. Confidentiality

CAN bus messages are sent over the common bus. Therefore, all ECUs connected to that bus will receive all messages and decide whether to use the message or not based on the type of ID of the message. Hence, privacy will be an issue when ECUs want to communicate confidential information. Therefore, an attacker can read all the data sent on the bus and is also able to even send data on the bus from a remote location using the external gateway. In FlexRay, the attacker can learn secret keys, proprietary or private data which are sent on the bus as well [5].

ii. Integrity

CAN uses CRC checksums to verify the message's integrity. Since CRC checksums are non-secure, CAN messages are open to alterations. In [6], using cryptographic hash functions are suggested which may seem convincing, but on the other hand, increased processing time, network overheads and delays must also be considered.

iii. Authenticity

A major security flaw in the in-vehicle networks is the lack of sender and receiver address in the frame. Using this design flaw, a message can easily be spoofed and sent on the bus for the victim ECU, e.g., an attacker can create and inject diagnostics messages and force ECUs to perform arbitrary actions [5]. ECUs cannot verify the authenticity of a message and will rely on its contents which may result in performing unauthorized and abnormal actions [6].

iv. Availability

Denial-of-Service attacks are hard to protect against. Since there is no control over a malfunctioning ECU in CAN networks, an attacker can repeatedly spoof error or high-priority messages and send them over the bus. This will culminate in a Denial-of-Service attack on the communication bus and other nodes cannot use the bus to send their messages. According to [6], "FlexRay considers the option of disconnecting malfunctioning devices or branches from the network by node-local or central bus guardians".

v. Non-Repudiation

Since the mentioned features are not fully addressed in CAN networks, it is almost impossible to identify a faulty ECU or a spoofed message after an attack or malicious behavior has occurred.

V. ATTACKS ON CAN NETWORKS

Originally, the Controller Area Network did not include security schemes that prevent attacks and abnormal conditions during communication. There are works focused on the analysis of characteristics, vulnerabilities of the CAN bus. In these works, authors describe and analyze the vulnerabilities and IDSs for the CAN bus, finding the following as some of them:

• Lack of Message Authentication of the nodes. Due to each ECU delivers and receives all data on the same bus

Unsegmented Network. Commonly, the CAN network is not segmented, due to this, components focused on safety-critical systems can communicate with infotainment in the same network.

• Lack of encrypted Messages. As the CAN bus was not designed to prevent hacker attacks, information is not encrypted, allowing access and possible modification of the data, or even its replacement.

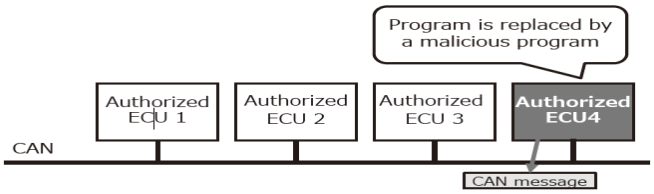
and, therefore, errors in the operation of the system.

• Vulnerable to DoS attacks and replay attacks. On the CAN bus, each device receives the information that is sent from a transmitter. Due to this an attacker could send frames with high priority and interrupt critical functions in the system.

i. Software Attacks on Can Networks:

a. **Spoofing attack:** Spoofing occurs when a compromised node sends CAN data frames with a changed (forged) ID field to masquerade as data or a command from a valid-source ECU node. The spoofing attack is easy to adapt to the CAN bus model. It has adverse effects because it decreases communication performance on the network [7]. Since CAN lacks authentication and the bus is a broadcast network, a compromised ECU might readily deliver CAN frames with any ID, even IDs belonging to other legitimate/critical ECUs. In this research, we achieved this spoofing attack by assuming that, first, an attacker gains physical access to the CAN bus and connects to it, effectively becoming part of the network. Since messages sent via the bus are in plain text, they can easily monitor and understand messages sent and can then skillfully alter bits within the packet frame to manipulate the CAN ID,

Use case 1: Unauthorized alteration of ECU software.
Figure 3 shows an example of spoofed message transmission by an ECU after its authorized program is replaced by a malicious program.



Use case 2: Connection of unauthorized device.
Figure 4. shows an example of spoofed message transmission by an unauthorized device connected on a CAN bus.

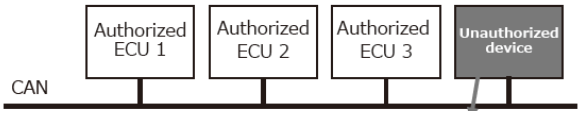


Figure 4. Connection of unauthorized device on CAN bus

Figure 3. Replacement of authorized ECU program

b. **Injection attack:** Generally, attackers use direct or indirect access points to inject messages into the CAN bus, suppress valid communications (i.e., genuine messages with higher-priority IDs than injected ones are ignored), or penetrate an ECU to perform malicious actions. Attacks against direct access points include the OBD-II port, CD player, and USB port [8]. In our case, we could inject futile messages to the bus via the WiFi node we attached to it. In our attack scenario, we implemented an injection attack where the attacker successfully spoofed the CAN ID to a high-priority ID, in particular 0x00. Consequently, the attacker's node flooded the bus with fraudulent or irrelevant messages by either continually injecting arbitrary messages into the CAN bus or injecting unauthenticated messages with the spoofed ID into the vehicle.

c. **Denial of Service (DoS) attack**: The CAN protocol is also subject to DoS attacks. CAN's arbitration system allows higher-priority nodes to talk first. Because of the prioritization on the CAN bus, if a malicious node with the highest priority is always active, the other nodes cannot interact.As shown in this research, an attacker can carry out a DoS attack to render a specific CAN bus system inoperable by conforming to the CAN standard or by breaching it [9]. We achieved this by transmitting as many messages to the CAN bus as physically possible with the smallest feasible ID (0x00). When the bus is idle, if two or more ECUs desire to transmit simultaneously, the one with the lowest ID will have priority (arbitration). As a result we noticed in the logs that, because the zero ID takes precedence over all other message IDs, none of the normal messages will win the arbitration against the injected message, resulting in the prevention of signal transfer from the regular ECUs.

d. **Eavesdropping attack:** Eavesdropping attack: Eavesdropping attacks occur when unauthorized individuals are able to gain access to vehicular messages. CAN's broadcast transmissions allow attackers who gain access to the in-vehicle network to then eavesdrop on CAN transmissions and identify patterns in legitimate CAN frames [10].

e. **Replay attack:** In a replay attack, attackers continually resend valid frames to impede the vehicle's real-time functioning [10].

f. **Bus-off attack:** Bus-off attacks occur when attackers continually send bits both in the identifier field and in other fields, which causes the ECU's transmit error counter (TEC) to then be incremented. When the TEC has a value greater than 255, the corresponding ECU has to shut down [11].

ii. Hardware Trojan attack on CAN Bus and countermeasures.

a. Theory of Hardware Trojan Attack on CAN Bus

The reliability and security of a system are dependent not only on the software but also on the underlying hardware. Embedded electronics are currently facing emerging hardware security threats known as Hardware Trojan (HT) [12] because of economic interest. HT is a stealthy modification of hardware implemented at the hardware level that remains inactive unless a rare event occurs. These characteristics make HTs extremely difficult to detect. Given the vulnerabilities of the CAN bus, HT attacks on CAN systems are simple to execute. The resulting system may be disastrous. Consider the CAN bus system depicted in Figure 2 with a trojanized CAN controller attached to Node 2. Because the CAN protocol lacks authentication and is broadcasting, Node 2 can masquerade as Node 1 or any other node in the network. This gives the malicious node complete control of the system. To test this hypothesis, we launched a DoS attack on the CAN bus raised by HT.

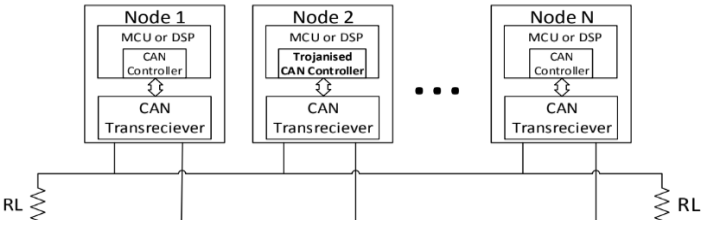


Fig 5. Trojenised CAN controller in a CAN bus communication system

In this attack scenario, the attacker takes advantage of a physical feature of the CAN protocol. When the Trojan is activated, the compromised node begins transmitting six consecutive logic zero bits. Additionally, transmitted logic zero takes precedence over logic one, resulting in six consecutive bits remaining at the same level. The purposeful violation of the Bit-Stuffing rule results in the generation of an error frame.

As a result, other nodes in the network discard the corrupted message, raising the error counters. When either of these error counters exceeds the 256 thresholds, the affected node enters a Bus-Off state. As shown in Figure 4, the node is effectively disabled from further participation in bus communication in this state.

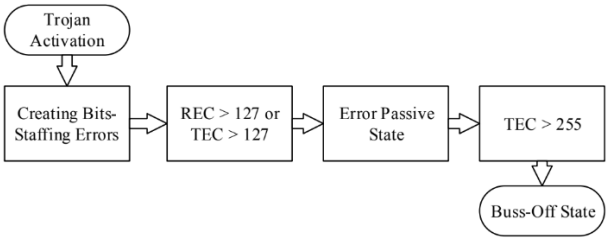


Fig 6. The process of disabling a node

b. Experimental Setup

A configuration is established to demonstrate the impact of a Hardware Trojan (HT) in a CAN system, in which two fully operational nodes are interconnected with a third node that has been compromised by a Trojan. This CAN network is intended to make it easier to observe HT effects. The trojanized node is set to remain inactive until triggered by a specific signal. As shown in Figure 7, this trigger signal is connected to the control pin of a multiplexer. When the trigger signal is activated, the trojanized node deviates from its intended behaviour and transmits manipulated data from the Trojan circuit, which replaces the original data from the CAN controller.

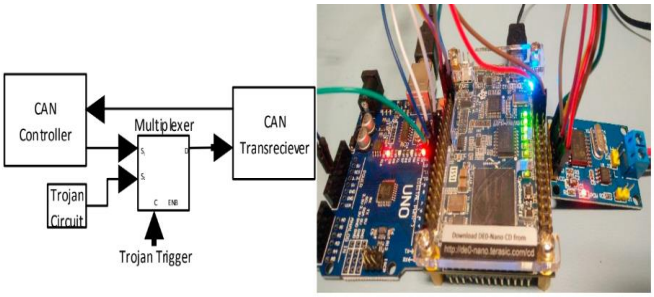


Fig 7. Hardware Trojan implementation in CAN bus controller; (b) attack implementation with Arduino UNO, Altera DE0 Nano, and Microchip MCP2551

The trojanized node, built with the Altera DE0 Nano FPGA kit and connected to the CAN bus via the Microchip MCP2551 CAN transceiver, is compared to other nodes built with Arduino Uno microcontroller boards and also linked to a Microchip MCP2551. The Hardware Trojan (HT) is activated by a time-activated trigger signal. This trigger signal is generated five minutes after the system is activated, prompting the HT to be activated. When activated, the Trojan circuit embedded within the Altera DE0 Nano FPGA kit transmits a sequence of six consecutive dominant logic '0' bits within each CAN frame. As a result, the remaining nodes in the network reject the manipulated message, resulting in the generation of an error flag. The trojanized node's Transmit Error Counter (TEC) rises by eight, while the other nodes' Receive Error Counters (REC) rise by one. When a threshold of 255 is reached, the TEC causes the trojanized node to enter a Bus Off state, effectively removing it from the system. The successful Denial of Service (DoS) attack renders the trojanized node disabled, rendering it incapable of servicing requests from other nodes, disrupting normal network operations.

c. Countermeasures

Although network segmentation can prevent most CAN attacks, HT is resilient to network segmentation. One possible solution is to incorporate a management control system into the system to monitor network traffic and notify the user of any anomalies. In this case, the supervisor controller should be tested for HT thoroughly. Car manufacturers should inspect and test every component they use during the manufacturing process. The end user should only use genuine parts that have been approved by the manufacturer.

iii. Proposed Attack Scenarios

a. Conquest attack:

The conquest attack appears to be a highly sophisticated and stealthy intrusion method into in-vehicle networks. The adversary successfully bypasses both security protection mechanisms and cutting-edge intrusion detection systems in this advanced form of attack.

The adversary directly takes control of the target Electronic Control Unit (ECU) in the conquest attack, achieving a level of compromise not seen in previous scenarios. The adversary can reprogram the ECU after fully compromising it, allowing for subtle manipulation of the payload of a sensitive message (B0). Unlike other scenarios, this attack has no effect on the normal behavior of ECUs in terms of message frequency, clock offset, or clock skew. This level of complexity makes the conquest attack particularly effective against Advanced Driver-Assistance Systems (ADAS) that rely heavily on the accuracy of sensor values. For example, the adversary might alter sensor data bytes within the normal range, causing critical safety systems such as forward collision warning and lane departure warning to behave incorrectly. A practical example of a conquest attack would be reprogramming a parking assistant module to send manipulated steering wheel angles directly to the steering wheel ECU, resulting in erratic behavior. In another hypothetical scenario, the adversary could tamper with engine control module data, subtly altering engine speed messages to provide misleading information to systems that rely on accurate engine speed readings. Such attacks highlight the potentially far-reaching consequences for the In-Vehicle Network (IVN) when adversaries can secretly manipulate critical sensor values.

b. Realtime Perfect bit Modification attack on In-Vehicle CAN

The In-Vehicle Perfect Bit Modification Attack in Real Time CAN is a new type of cyber-attack that targets the Controller Area Network (CAN) vulnerabilities in vehicles. Frame injection and dominant bit injection techniques have been used in traditional CAN cyber-attacks. These methods, however, cannot modify data frames sent from an Electronic Control Unit (ECU) in real time, limiting their ability to perfectly control the target system. Because the dominant bit injection can only perform denial of service (DoS) attacks, it cannot fully control the ECU. The Perfect Bit Modification (PBM) technique is a new attack method that allows the attacker to modify either dominant or recessive bits represented in a CAN bus. The Bus Possession Attack (BPA) and the Target ID Attack (TIA) are introduced in this technique. The BPA can launch an attack without disrupting the CAN bus's communication pattern, whereas the TIA can completely seize control of a specific ECU. The PBM technique is implemented using the SN65HVD230 (CAN transceiver) and the FDS8949 (MOSFET), both of which are commonly used in the construction of ECUs. Because these components are inexpensive, these attack models could be used for supply chain attacks. The BPA model disables all ECUs' transmission functions and generates only the data frames required by the attacker for a set period. Only data frames with a specific ID are modified by the TIA model. During the TIA's operation, ECUs that use the target ID generate error frames or retransmit data frames that were not transmitted, but the TIA also modifies all these frames.

Attack experiments were carried out on two real vehicles to demonstrate the feasibility of BPA and TIA. The research also suggests countermeasures for building a secure CAN environment.

VI. Comprehensive Overview, Security Challenges, and Innovative Solutions for LIN Bus

A. Overview of LIN Bus:

Introduced collaboratively in 1998 by prominent automotive manufacturers, including BMW, Audi, Daimler Chrysler,

Mercedes-Benz, Volcano Automotive, and Volkswagen, in partnership with Motorola, the Local Interconnected Network (LIN) represents a standardized and cost-effective bus system designed for diverse in-car functionalities. Serving as a sub-bus system, LIN employs a serial communication protocol, supporting single master/multiple slave functions and incorporating precise time synchronization for accurate message transmission. An advanced error detection mechanism, utilizing data checksums and parity checks, enhances data integrity. The LIN frame comprises a master-transmitted header with essential elements, including sync break, delimiter bit, sync field, and a protected identifier (PID), along with a slave-transmitted data block accommodating up to 8 bytes of data and a checksum. Key features include cost-effectiveness, widespread use in car production, application in body and comfort systems, robust error detection, 20 Kbits/sec communication speed, support for 64 identifiers, and control over various functions like seating position motors and climate systems [13].

B. Security Issues and Solutions for LIN Bus:

While serving a crucial role in automotive communication, the LIN bus faces security challenges that demand careful consideration and robust solutions.

a. **Low-Security Detection Mechanism:** The LIN bus relies on checksums and parity bits for error detection. To fortify against evolving cyber threats, researchers advocate for augmenting error detection mechanisms, proposing a multi-layered approach to enhance security. A more comprehensive error detection mechanism, integrating anomaly detection algorithms and redundant checks, could provide an added layer of protection against potential cyber threats, ensuring the integrity of data transmissions within the LIN network.

b. **Unencrypted Messages:** Concerns arise with the transmission of unencrypted messages, posing threats to data integrity and confidentiality [14]. Security analysis recommends the incorporation of encryption protocols, such as Advanced Encryption Standard (AES), to shield messages from unauthorized interception and manipulation. Expanding on this solution, the implementation of dynamic encryption keys that change at regular intervals can further enhance the security of transmitted data, thwarting potential attackers attempting to decipher messages over an extended period.

c. **Architectural Limitations:** The LIN bus architecture, reliant on a single master, introduces vulnerabilities exploitable in specific attack scenarios. Diversifying the architecture with redundant masters or fail over mechanisms, as proposed in research, mitigates risks associated with a compromised master. An innovative solution involves the implementation of a distributed master architecture, where multiple master nodes collaboratively manage communication. This distributed approach not only enhances security by reducing the impact of a compromised master but also contributes to the overall resilience and reliability of the LIN network.

d. **Broadcast Transmission:** The broadcast nature of LIN communication poses challenges in ensuring secure communication. Implementation of secure communication protocols and access controls is suggested to counter unauthorized access and potential message injection attacks. Building on this, the integration of secure network segmentation can provide an additional layer of protection. By categorizing LIN network segments based on sensitivity levels, access controls can be tailored, minimizing the risk of unauthorized access to critical segments and preventing potential malicious activities.

e. **Dependency on Slaves and Single Master:** Dependency on slaves and a single master introduces a potential point of failure. Proposals include the introduction of redundant master nodes or failover mechanisms to ensure continuous communication even if the primary master is compromised. Extending this solution involves the implementation of an adaptive network topology, where the LIN network dynamically adjusts its structure based on real-time conditions. This adaptability ensures optimal performance and security, especially in scenarios where nodes or masters may become compromised or unavailable.

f. **Restriction to Non-Critical Functions:** LIN's common use for non-critical functions requires a comprehensive security strategy [15]. Development of intrusion detection systems tailored for LIN networks is proposed to monitor network behaviour and mitigate potential security breaches promptly. Further enhancement involves the integration of machine learning algorithms within intrusion detection systems. These algorithms can continuously learn and adapt to emerging threats, providing a proactive defence mechanism against evolving cyber-attacks, and ensuring the long-term security of LIN bus communication in critical automotive applications.

VII. FLEXRAY BUS: AN EXTENSIVE EXAMINATION OF PROTOCOL, SECURITY CHALLENGES, AND ADVANCED SOLUTIONS

In the intricate realm of automotive electronics, the Electronic Control Unit (ECU) takes center stage, orchestrating the monitoring and control of diverse functions within vehicles. The automotive landscape, spurred by innovations like telematics and Advanced Driver Assistance Systems (ADAS), necessitates sophisticated real-time communication networks. FlexRay, conceived in 2000 through collaborative efforts by industry giants Daimler Chrysler, BMW, Freescale, and Phillips, emerges as a high-bandwidth communication protocol designed to cater to the escalating demands of high-speed technology in automotive vehicles. This protocol, characterized by support for single or dual-channel configurations and leveraging differential signaling for noise reduction, positions itself as a crucial solution to the evolving challenges in the automotive sector [16].

FlexRay Frame Structure:

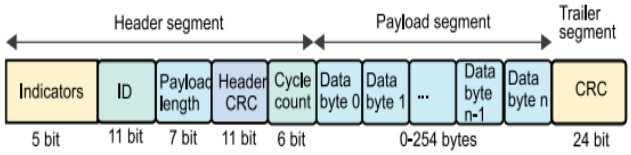


Fig 8

Examining the architecture, Fig. 8 vividly illustrates the FlexRay frame's intricacies, featuring three core elements – header, payload, and forward. The initial five header bits play a foundational role in setting essential properties within the data frame. The Frame ID assumes significance by defining the slot position in a static segment (housing critical information) and conveying frame priority for a dynamic segment handling lower-priority data. Payload length is intricately tied to data length, and the header's cyclic redundancy check (CRC) involves a synchronization frame indicator, a start frame indicator, frame identification, and the useful load length. The cycle count serves as a serial number for the locally defined frame at a node, and the trailer segment finalizes the CRC by calculating it on payload and header segments [17].

A. In-Depth Features of FlexRay Protocol:

FlexRay, distinguished by a range of features, establishes itself as a versatile and efficient protocol for automotive applications:

- **Time Triggering Protocol:** Ensures precise timing for communication processes.
- **Deterministic Behavior:** Guarantees consistent and predictable performance.
- **Fault Tolerance:** Equipped to handle faults, contributing to system robustness.
- **Diverse Network Configurations:** Supports configurations like bus, star, and multi/sta networks, expandable up to 64 nodes.
- **Utilization in Hard Real-Time Systems:** Deployed in critical applications such as chassis, brake direction, and driver assistance.
- **Cyclic Data Transmission Control:** Leverages Time Division Multiple Access (TDMA) and flexible TDMA (FTDMA) for efficient control of data transmission.

B. Integration into Automotive Systems and System Levels:

FlexRay seamlessly integrates into the automotive landscape, forming a critical part of the data backbone alongside other buses like CAN, LIN, and MOST. This integration extends to various levels within the FlexRay system architecture:

- **Network Level:** Defines single/dual-channel configurations and different topologies like bus-type, star-type, and hybrid-type.
- **Interface Level:** Introduces components like a guardian bus, physical interface, and error detection mechanisms, ensuring containment in the time domain.
- **Protocol Engine Level:** Involves the setup of the control host interface (CHI) and protocol engine [18]. The interconnectedness is visually represented in Fig. 14, illustrating the brake pedal control and brake control systems linked through a FlexRay network.

c. FlexRay Security Challenges and Advanced Solutions:

Aligning with the established taxonomy applied to the CAN-bus, the security landscape of the FlexRay protocol is scrutinized, with a focus on algorithmic techniques and comprehensive solutions. In parallel with the CAN bus, primary algorithmic methodologies include Neural Networks (NN) and cryptographic techniques, supplemented by diverse implementation strategies [19, 20].

Innovative Network Monitoring Control:

- Introduction of a FlexRay network monitoring control emphasizing security and reliability.
- Neural Networks (NN) operate as predictive models, dynamically adapting to varying loads to ensure stability and performance.

Rigorous FlexRay Scheduling Analysis:

- In-depth scrutiny of FlexRay scheduling dynamics and characteristics, encompassing both static and dynamic segments, coupled with thorough implementation verification [21].

Addressing Vulnerabilities and Hidden Attacks:

- Identification and consideration of vulnerabilities and concealed attacks on FlexRay.
- Implementation of Advanced Encryption Standard (AES)-128 and SHA1 algorithms, emphasizing confidentiality and authentication [18].

Enhanced Authentication Mechanism:

- Introduction of a lightweight yet robust authentication mechanism to fortify FlexRay's security.
- Successful achievement of encrypted and authenticated message transmission, enhancing system flexibility while maintaining robustness [22].

Adaptive Authentication Protocol:

- Proposal of a transmission's adaptive authentication protocol, capitalizing on time-triggered communication in FlexRay.
- Deliberate consideration of non-deterministic transmissions due to the dynamic segment in FlexRay [23].

Distributed FlexRay Approach:

- Introduction of a distributed FlexRay-based approach, strategically deploying computationally intensive authentication algorithms.
- Optimization problem-solving to determine the minimal number of co-processing units required for achieving security and deadlines [24].

Evolution of FlexRay Architectures:

- Presentation of advanced and improved FlexRay architectures, aiming to address existing challenges.
- Proposing a CAN/FlexRay gateway through hardware/software (HW/SW) co-design, effectively reducing execution time for in-vehicle networks [25] [26] [27].

Communication Controllers Tailored for FlexRay:

- In-depth focus on communication controllers designed specifically for the FlexRay protocol.
- Implementation on Field Programmable Gate Arrays (FPGAs) ensuring adaptability to FlexRay networks with configurable feature extensions [28].

Industry Perspectives and Adoption Trends:

The prevailing sentiment within FlexRay research underscores its capacity to cover characteristics lacking in the CAN bus. Consequently, the automotive industry is gradually steering towards embracing FlexRay either as a standalone solution or in conjunction with CAN bus and other protocols. This strategic shift is motivated by the desire to fortify security measures within vehicular communication systems [29].

VIII. Security Solutions

Security Measures for CAN in In-vehicle Networks

When numerous reported instances of attacks on in-vehicle control systems, significant research has been conducted to develop robust measures aimed at protecting these systems from potential threats (references 6–10). Nonetheless, a common issue has emerged in the implementation of these safeguards. The identified issue is focused on the requirement that all network nodes adopt and integrate these measures to ensure their effectiveness.

In layman's terms, the issue is one of universal application—every component connected to the network must adhere to and carry out the protective measures for them to serve their purpose. This requirement presents a logistical challenge, as achieving comprehensive compliance across all nodes in a network can be a difficult and time-consuming task. The effectiveness of the protective measures is dependent on the collective commitment of all network entities, emphasizing the need for a coordinated and widespread adoption of security measures to reduce the risk of attacks on in-vehicle control systems.

A. Method proposed: Security Monitoring System

The security monitoring system outlined in this paper is visually represented in Figure 9. The vulnerability of the CAN (Controller Area Network) protocol to spoofing attacks, owing to its lack of an authentication system, is a primary concern, as discussed earlier. To address this vulnerability, the proposed monitoring system employs message authentication codes (MACs), which are generally effective in thwarting spoofing attempts. The primary goal of this security monitoring system is to guard against spoofing attacks.

The system's monitoring node is in charge of verifying the message authentication codes connected to the CAN messages and authenticating every Electronic Control Unit (ECU). It is essential to incorporate a specialized CAN controller into the monitoring node in order to streamline this process. By authenticating each ECU and confirming the accuracy of the message authentication codes attached to CAN messages, this specialized controller plays a crucial role in maintaining the integrity and authenticity of communications. Basically, the idea behind the proposed system is to use a dedicated monitoring node to implement strong message authentication measures, thereby strengthening the CAN protocol against the possibility of spoofing attacks.

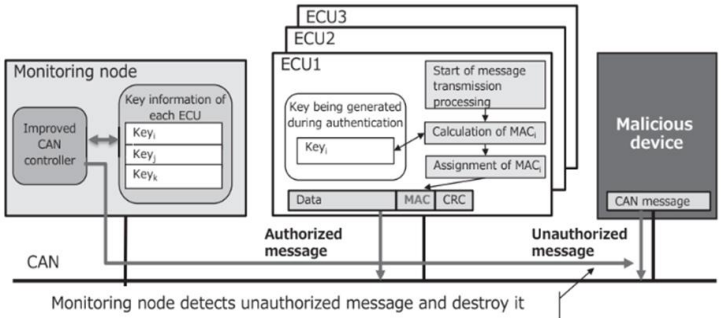


Fig 9. Proposed Security Monitoring System.

The special purpose CAN controller replaces faked messages in real time using an error frame base. The benefit of the suggested security monitoring system is that it just needs new hardware in order to the CAN bus's monitoring node. The only software that needs to be changed for other ECUs is the node authentication and key exchange software. Using an encryption key, a transmission node (approved ECU) determines the MAC and provides a portion of the data frame is transmitted after applying the computed MAC to the payload. The monitoring node uses the same encryption key as each transmission node to confirm the MAC after consulting the CRC. When a spoof message is detected, the new CAN controller uses an error frame to replace the message. The two stages of the suggested security monitoring system are as follows:

Phase of node authentication and key delivery

Phase of monitoring spoofed message and overwriting the message with error frame on a real time basis.

The approaches used for delivering cryptographic keys in traditional systems are overly complex and result in significant communication overhead on the Controller Area Network (CAN) buses. This complexity presents a challenge, particularly for in-vehicle control systems that require real-time data processing. Recognizing the unique requirements of vehicle applications necessitates the use of simple authentication and key delivery protocols. The goal is to simplify the process so that it is efficient and does not consume too many computational resources, allowing for timely and responsive data processing within in-vehicle control systems.

B. Mutual authentication of node and key exchange

Since the payload of the data frame is not encrypted by our security monitoring system, there is no need for every node to have the same encryption key. Therefore, in order to authenticate communication between the monitoring node and each gearbox node (Electronic Control Unit, or ECU), we have chosen to use a challenge-response system. In Figure 4, we use a mutual authentication sequence to help make things more understandable**.** Mutual authentication is essentially a set of procedures wherein the transmission nodes (ECUs) and the monitoring node confirm each other's identities. This challenge-response system improves authentication by using a dynamic and more secure method rather than depending on a shared encryption key. Figure 4 provides a visual representation of this sequence, showing the steps that must be taken to guarantee mutual authentication between the monitoring node and every ECU in the system.

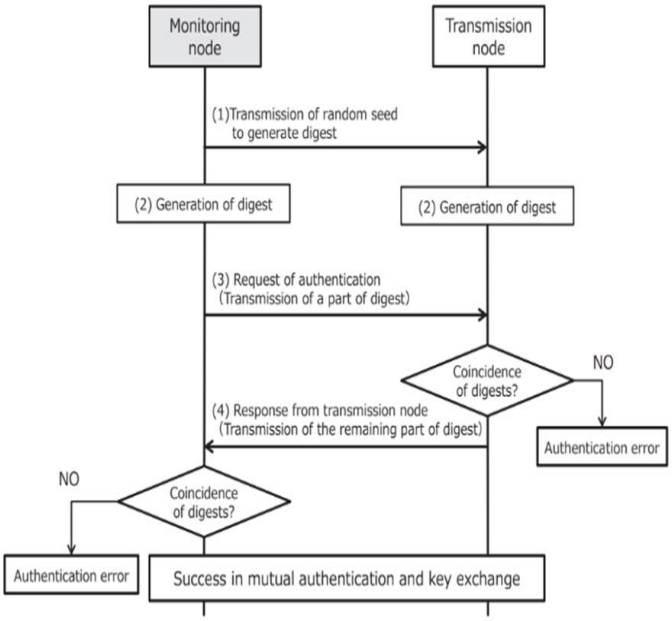


Fig. 10 Protocol for mutual authentication and key exchange

A random seed is sent to the transmission node by the monitoring node.

Both nodes compute the digest after the transmission node gets the random seed from the monitoring node. Typically, a hash function is employed in the digest computation.

A portion of the digest is sent from the monitoring node to the transmission node.

After computing the digest, the transmission node compares it to the portion of the digest that was sent from the monitoring node. The transmission node sends the following digests to the monitoring node when the two digests are equal.

By calculating the digest and comparing it with the portion of the digest it received from the transmission node, the monitoring node performs authentication.

C. Pre-shared information and encryption key

The security monitoring system that is suggested here is predicated on the idea that the transmission nodes lack tamper-resistant memory. In light of this presumption, the system depends on ROM data and programme code that include a distinct ID that acts as a pre-shared key. In essence, a unique code kept in each transmission node's read-only memory (ROM) serves as its identification. For the monitoring node to function properly, it is necessary for it to have the programme codes of every transmission node.

In practical terms, by calculating and storing different authentication codes ahead of time, the monitoring node can maximise memory usage. Although the SHA-256 hash function is used in this demonstration, it's important to remember that the suggested security monitoring system can use a variety of techniques. The details of the encryption key—including how it is generated and used—are explained in more detail in the description that follows. However, many other methods can be used for our system. The details of the encryption key are described below.

Pre-shared information

To make the demonstration easier to understand, we used a programme code as pre-shared information. In this instance, programme size affected how long it took to generate the authentication code. The consequences of programme code leaks must be kept to a minimum because they represent a weakness in our system. It is better for each ECU to have its own key as shared data in an actual use environment. Generation of an authentication code is expressed by the following formula:

AUTHKEYI = SHA256 (MSG || NONCE)

AUTHKEYI represents the authentication code of transmission model and the SHA256 function represents a function for SHA-256 hash calculation. MSG represents the program code of the transmission node and NONCE represents a random seed.

MAC generation/verification key

The remaining portion of the previously mentioned AUTHKEYI is used as the MAC generation/verification key in the security monitoring system that is suggested in this paper. A 128-bit MAC generation key was mounted as a result.

D. Authentication message

This proposal proposes a unique security monitoring system that separates the frame used for message transmission from the frame used for authentication information transmission. The differences between a CAN message used in our system and a more traditional CAN message are shown in Figure 11. A section of the Message Authentication Code (MAC) is assigned to a payload component in the CAN message that is unique to our system. It's significant to note that encryption is not applied to the payload itself.

A portion of the generated digest is assigned to the entire payload in the mutual authentication scenario shown in Figure 10. This suggests that a particular portion of the authentication code is linked to the payload's content during the authentication process, enabling the integrity and authenticity of the entire message to be confirmed. This tactic makes it easier to strike a compromise between keeping the payload clear—which in this specific security monitoring system is left unencrypted—and securing the transmission through authentication.

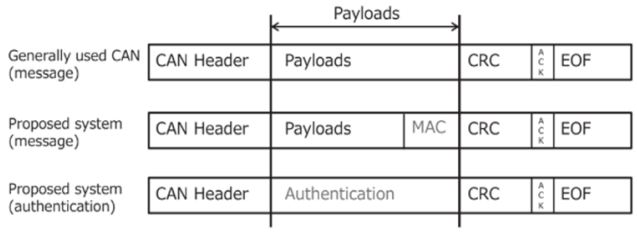


Fig 11 CAN protocol of proposed security monitoring system

E. Operation of monitoring node

All receiving nodes in a commonly used CAN protocol use a CAN controller to carry out a CRC check and, as a result, identify transmission errors. In our system, a portion of the payload is assigned a MAC to ensure the CAN message's integrity, as depicted in **Figure 12.**

In this case, the Message Authentication Code's (MAC) main purpose is to protect data on the in-car network from spoofing attempts. It is important to note that the proposed security system does not replace the Cyclic Redundancy Check (CRC) with the MAC in order to ensure compatibility with current CAN controllers.

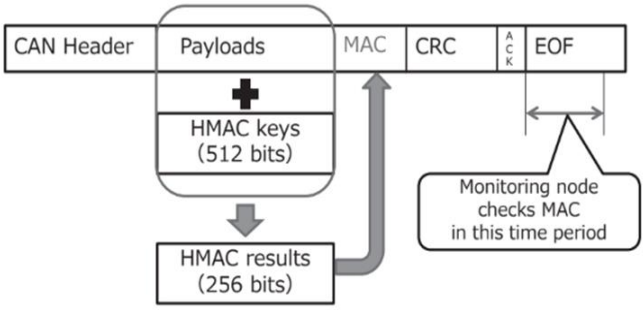


Fig 12 HMAC calculations timing

Upon receiving a message, the monitoring node examines the CAN-ID to determine if the message's MAC has been assigned to it. The monitoring node instantly computes the HMAC to confirm the MAC after receiving the message's MAC. As of right now, none of the nodes have sent the CAN bus an ACK signal. On the other hand, the monitoring node replaces the spoof message with the error frame up to the end of frame (EOF) if it detects the MAC error. Therefore, our system can prevent the transmission of spoof messages without requiring MAC verification by every reception node thanks to substitutional verification of the MAC by the monitoring node. We may additionally prevent replay attacks by providing a portion of the monotonic counter to the payload.

F. Real Time survey

Several symmetric key encryption and authentication schemes have been put forth especially for in-car networks in recent years. Notably, CanAuth [30] and its successor LiBrA-CAN [31] are two robust schemes introduced by Verbauwhede et al. Since these schemes only use shared-key authentication, it is not supported to include passive listeners from outside parties. It's crucial to emphasise that single-source node authentication is not made easier by CanAuth. Rather, it confirms that the message comes from a reliable source. Moving on to LiBrA-CAN, it's important to keep in mind that, despite its strength, this scheme's design may require a significant number of keys to be implemented in practise. Although the schemes are strong, their application to in-vehicle networks require careful consideration due to their unique authentication and key management features.

TESLA, an alternative broadcast network described in reference [32], employs a strong symmetric approach with delayed key disclosure. This protocol, however, has received significant criticism due to its reliance on storing unauthenticated messages until their verifying key is disclosed. Because of this feature, TESLA is vulnerable to Denial of Service (DOS) attacks. Subsequent versions of TESLA attempted to address this vulnerability, but the proposed solutions resulted in a significant increase in message latency.

E-safety Vehicle Intrusion protected Applications (EVITA), a European-funded project, developed a hardware security module (HSM) for On-Board network security [33].

Schweppe et al. proposed an EVITA- HSMbased communication security architecture for vehicles [34]. Due to the limited properties of an in-vehicle network (CAN bus load and bandwidth), they used a truncated 32-bit MAC and explained that a 32-bit MAC is secure from collision attacks for 35 weeks. Schweppe et al.'s security architecture, on the other hand, is very abstract. It does not give a detailed explanation regarding the generation and transmission of a 32-bit MAC.

The authentication scheme presented in this paper uses a simplified version of the International Data Encryption Algorithm (IDEA) to generate a Message Authentication Code (MAC), avoiding the use of S-boxes and lookup tables in its implementation. A truncated MAC, specifically 32 bits in length, is proposed to optimise bandwidth usage. This truncation effectively cuts the payload size in half, resulting in increased bandwidth efficiency. In addition, the proposed scheme includes the use of a message counter among Electronic Control Units (ECUs) to prevent replay attacks. To stop adversaries from replaying previously intercepted messages in an attempt to trick the system, this counter records the order in which messages are exchanged between ECUs. To sum up, the authentication scheme integrates message counters to strengthen security against replay attacks, a truncated MAC to reduce payload size, and the simplified use of IDEA for MAC generation.

G. GID-Anonymization for Secure CAN: In real-time systems, there are two major drawbacks: firstly, receivers accept all incoming frames regardless of their validity, risking the processing of potentially harmful data; secondly, the need for cryptographic verification introduces delays, making the system susceptible to both significant delays and Denial of Service attacks.

ID-Anonymization for Secure CAN (IA-CAN) is a two-step authentication process that is designed to protect in-vehicle communication against unauthorized access and attacks.

The first step involves generating an anonymous ID (A-ID) on a per-frame basis, which enables the authentication of the sender. Only an authorized sender or receiver can generate or identify a valid A-ID using a shared secret key and a random nonce. The receiver ECUs update their filters by pre-computing the A-ID and, upon receiving a frame, filter it. The ID is altered or anonymized on a per-frame basis and invalid frames are filtered without requiring any additional run-time computation. Since each A-ID is used only once, the attacker does not gain anything from reusing the captured A-ID (i.e., replay attacks are not possible).

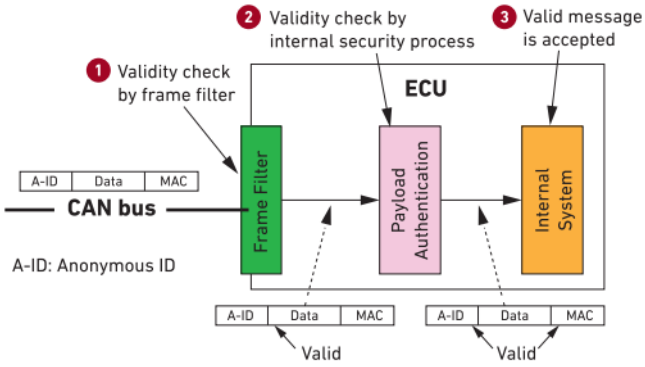


Fig 13 Two step-authentication processes of IA-CAN.

The second step involves message authentication to check the validity of data. The current A-ID is generated from the previously used A-ID (initially from original ID assigned to the frame type), and shared secrets are established by using a nonce per session. The shared secrets are composed of a pre-shared key and a shared secret from a previous transmission between authorized entities. Overall, IA-CAN randomizes the CAN ID by using cryptographic operations, which makes the frame ID anonymous to unauthorized entities but identifiable by the authorized entities.

IA CAN provides robustness against Denial of Service (DoS) attacks. Within the realm of CAN there are two types of DoS attacks: flooding attacks and starvation attacks. A flooding attack involves inundating a target ECU with a few frames while a starvation attack disrupts transmission over the CAN bus preventing ECUs from receiving frames. In the presence of a flooding attack IA CAN maintains a low and stable latency compared to existing security protocols that experience linear increases, in latency until the ECU becomes blocked. Additionally in the face of a starvation attack ECUs can maintain listening frames by transitioning into a mode. This capability is advantageous when it comes to recovery planning.

H. Secure Connectivity Between Vehicles and External Devices:

Secure Connectivity Between Vehicles and External Devices is a three-step authentication protocol that allows external devices to be securely integrated with the vehicle's electronics. The protocol is intended to prevent unauthorized access and attacks, as well as to ensure that only authorized devices can access the vehicle's systems.

In order to start the protocol, the first thing we do is authenticate the device. This involves the device sending a request to connect with the vehicle's electronics. The vehicles [2] electronics then respond by giving a challenge that the external device needs to solve using a shared key. If the challenge is solved correctly, it means that the device is authenticated, and the connection can be established.

The protocol's second step is session key generation, in which the vehicle's electronics and the external device generate a session key that is used to encrypt and decrypt data exchanged between them. The session key is generated by combining the pre-shared key and a random nonce, ensuring that it is unique for each session [35].

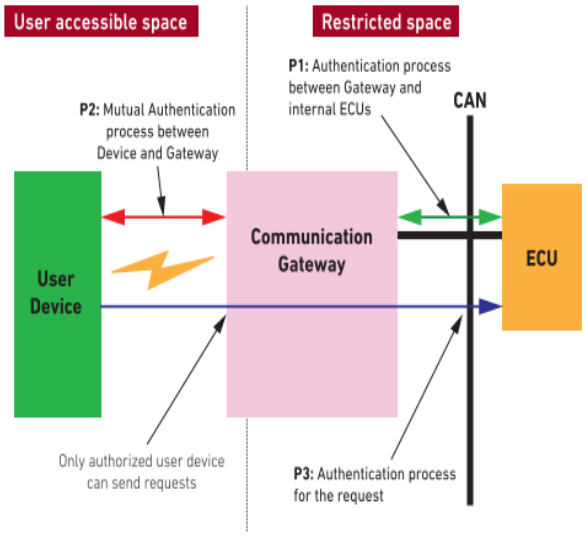


Fig 14 Three-step authentication for secure connection between external entities (user device) and ECUs (CAN).

The third step of the protocol involves data encryption and decryption, in which the vehicle's electronics and the external device encrypt and decrypt data exchanged between them using the session key. The data is encrypted using a symmetric encryption algorithm, such as AES, to protect it from unauthorized access and attacks.

Overall, the Secure Connectivity Between Vehicles and External Devices protocol provides a secure and reliable way for external devices to connect to the vehicle's electronics. By using a combination of device authentication, session key generation, and data encryption and decryption, the protocol ensures that only authorized devices can access the vehicle's systems, and that the data exchanged between them is protected against unauthorized access and attacks.

I. Countermeasures against the hardware trojanized attacks on CAN bus:

Countermeasures against hardware trojanized attacks on the Controller Area Network (CAN) bus involve implementing security measures to detect and prevent unauthorized modifications or intrusions into the hardware components of the CAN [36] bus system. Here are some countermeasures:

1. Cryptographic Techniques:

Message Authentication Code (MAC): This article highlights the importance of MAC to guard against masquerading, eavesdropping, injection, and replay attacks. Existing challenges include the limitation of standard CAN data fields to 8 bytes, making it difficult to fit MAC.

2. Digital Signature:

Digital signatures are used to verify the authenticity and integrity of messages transmitted over the CAN bus. The Elliptic Curve Digital Signature Algorithm (ECDSA) [37]is a commonly used hashing technique that can prevent bogus information attacks and provide more authentication to the data.

3. IDS (intrusion detection systems):

Install intrusion detection systems that will monitor the CAN bus for any unusual or unexpected behavior. An intrusion detection system (IDS) can detect anomalies in message patterns or unexpected changes in the behavior of connected devices, indicating a possible hardware trojanized attack.

4. Firewalls

A firewall is a software or hardware device that monitors and controls network traffic in order to prevent unauthorized access. A firewall can be used to protect the (CAN) bus from unauthorized access and to prevent malicious messages from being transmitted. The firewall can be set to allow only authorized messages to pass through while blocking any messages that do not meet the specified criteria. This can aid in the prevention of CAN bus masquerading, injection, and other types of attacks. It should be noted, however, that a firewall alone may not be sufficient to secure the CAN bus, and other countermeasures such as access control, encryption, and intrusion detection and prevention systems (IDPS) may also be required.

5. Access Control:

Access control involves limiting access to the CAN bus to authorized users only through the use of passwords, biometrics, or other authentication methods.

6. Offset Ratio and Time Interval based Intrusion Detection System (OTIDS):

OTIDS is a CAN bus security system that checks for attacks by requesting data frames on a regular basis and monitoring response times and offset ratios. It can detect denial of service, masquerading, and injection attacks, making it an effective countermeasure for improving CAN bus security and lowering the risk of cyber-attacks.

J. Firewall and secure gateway for in-vehicle networks:

The challenges and security concerns associated with the evolving field of automotive technology, driven by emerging trends such as connected vehicles and Vehicle-to-Everything (V2X) connectivity. This shift calls into question the traditional view of automobiles as closed systems, necessitating a paradigm shift in cybersecurity approaches. The growing integration of external interfaces necessitates rigorous monitoring and analysis of incoming traffic to identify potentially malicious data that could jeopardize the safety-critical functions of Distributed Automotive Cyber-Physical Systems (DACPS).

Furthermore, in the future E/E architecture of DACPS, DACPS is divided into several subsystems with varying security requirements, and they are interconnected by gateways or domain controllers. Thus, internal communication between different ECUs and subsystems generates a significant amount of traffic and adds another attack surface. As a result, firewall technologies such as stateless packet filtering, stateful packet inspection, proxy servers, and application layer firewall with deep packet inspection must be implemented in switching ECUs such as gateways, domain controllers, in-vehicle infotainment systems, and T-Boxes.

Specific proposals made by researchers to address these issues. Kurachi[3] et al., for example, propose a secure gateway with features such as a whitelist-based firewall, detection and response to DoS attacks, centralized authentication, and malware detection. Other researchers, such as Luo and Hu[4], concentrate on defining gateway security requirements based on threat analysis, using message-filter-based firewalls, key management functions, HMAC-based authentication, and AES encryption for secure in-vehicle communication. These proposals highlight a wide range of strategies, ranging from hardware and software codesign methods to distributed implementations, reflecting the complexities of securing in-vehicle networks. The distilled insights provide a comprehensive overview of the proposed firewall technologies, encompassing their features and potential applications in the evolving landscape of automotive cybersecurity.

Integrating firewalls and secure gateways into CAN networks involves a combination of access control, encryption, authentication, and monitoring mechanisms. The goal is to create a robust and secure communication environment within the automotive system, mitigating the risks associated with potential cyber threats. It requires the addition of a monitor node to extract characteristics from measured signals, and complex analysis methods such as support vector machine (SVM), neural network (NN), and bagged decision tree are used for classification and ECU identification, resulting in significant timing and memory overheads.

K. Intrusion detection systems:

IDS is typically used to protect the security of information systems, but many studies have attempted to incorporate it into the availability protection of in-vehicle networks.

Implementing Intrusion Detection Systems (IDS) based on the characteristics of Electronic Control Units (ECUs), in a Controller Area Network (CAN) presents both challenges and advantages. One of the challenges is that CAN messages do not naturally carry identity information making it difficult to establish the authenticity of the sender. The traditional approach of message authentication code (MAC) and freshness value (FV) has been described as a solution, but it incurs significant bandwidth and time requirements, and remains difficult to achieve global synchronization of FV.

Physical invariants of ECUs, such as clock offset, voltage distribution, and signal characteristics, which can serve as fingerprint information for ECU identification. Each ECU has a unique quartz crystal clock, there is a small difference in clock frequency of any two clocks, and this is defined as the clock skew. Clock skew can be used to implement an IDS to identify attacks launched by malicious ECUs.

Using timestamps [5]from received messages to determine the clock skew of each ECU and posit messages sent from the same ECU. The clock skew-based IDS has the advantage of requiring no hardware support or software modification, but the disadvantage is that it can only be used for periodically received messages. Message scheduling, queuing, and arbitration delays, on the other hand, will cause a deviation in the message's period, making the clock skew-based IDS unstable. It is recommended to use unique signal characteristics for ECU fingerprinting[6]; this type of IDS can be used for both periodically and infrequently transmitted messages.

Intrusion Detection Systems (IDS) are vital components in cybersecurity, employing real-time or batch processing to monitor and respond to security threats. They use both anomaly and signature-based detection methods, generating alerts upon detecting suspicious activity. IDS contributes to forensic analysis by logging information and exhibiting diverse response mechanisms. Categorized as network-based (NIDS) or host-based (HIDS), they ensure scalability to accommodate network complexity. Customization options enable tailoring to specific security needs, and regular updates to attack databases are crucial for staying ahead of emerging threats, maintaining their effectiveness in proactive cybersecurity measures.

We summarize and categorize the most recent works on IDS of CAN into categories.

Intrusion Detection Systems (IDS) encompass various types, each designed to address specific aspects of cybersecurity threats. Network-based Intrusion Detection Systems (NIDS) operate at the network level, monitoring traffic and analyzing data packets to identify potential security threats. They recognize malicious network activity using signatures and anomaly detection methods, providing a comprehensive view

of network-wide threats. NIDS are strategically deployed at key network points like gateways or routers, allowing them to scrutinize all incoming and outgoing traffic.

In contrast, Host-based Intrusion Detection Systems (HIDS) focus on individual hosts or devices within a network. They analyze host-specific data, such as log files and system calls, to identify abnormal behaviors or signs of intrusions. Deployed directly on individual hosts, HIDS offer detailed insights into activities at the system level. While NIDS and HIDS cater to different scopes, their combined deployment enhances overall security posture.

Signature-based IDS relies on a database of known attack patterns, offering effective detection of recognized threats. Anomaly-based IDS, on the other hand, establish a baseline of normal behavior and flag deviations as potential threats, excelling in identifying novel attacks. Combining both approaches, Hybrid IDS aims to maximize accuracy by leveraging the strengths of signature-based and anomaly-based methods. This synergy addresses the limitations of individual methods, offering a more robust defense against a wide range of threats.

The advantages of intrusion detection systems (IDS) stem from their ability to continuously monitor system and network activities and generate alerts when suspicious or malicious activity is detected. They help with forensic analysis by logging information for post-incident investigation. IDS can automate responses to detected threats or make manual intervention recommendations. Network scalability, customization options, and continuous updates to attack databases are critical features that ensure adaptability and effectiveness in the ever-changing cybersecurity landscape. In summary, the various types of IDS, each with their own unique functionality, contribute to a layered defense strategy, enhancing system resilience against a wide range of cyber threats.

IX. Future Challenges

Securing in-vehicle control networks in modern automotive systems is critical, and the challenges are evolving. The growing number of connected devices and the increasing complexity of in-vehicle networks pose a challenge because they introduce more potential entry points for cyber threats, thereby broadening the attack surface. Integrating autonomous driving features necessitates extensive communication and coordination among ECUs, sensors, and actuators, making autonomous vehicle systems vulnerable to sophisticated cyber-attacks and necessitating advanced security measures. The evolution of communication protocols, including the transition to advanced standards, presents difficulties in maintaining backward compatibility and securing diverse communication methods.

Privacy concerns arise from the extensive data collection in in-vehicle systems, necessitating a delicate balance between data-driven functionalities and user privacy expectations. Evolving and diverse regulatory frameworks pose challenges, requiring continuous adjustments to security measures for compliance and addressing emerging threats while fostering innovation. Legacy system security, characterised by outdated features, poses challenges in retrofitting security measures into older vehicles without compromising functionality. Addressing these challenges requires a multidisciplinary approach involving collaboration among automotive manufacturers, cybersecurity experts, regulators, and the technology community. Proactive security measures, continuous monitoring, and industry-wide information sharing are essential for navigating the evolving landscape of in-vehicle control network security.

The extensive data collection in in-vehicle systems raises privacy concerns, necessitating a delicate balance between data-driven functionalities and user privacy expectations. Evolving and diverse regulatory frameworks present challenges, necessitating ongoing adjustments to security measures for compliance and addressing emerging threats while encouraging innovation. Legacy system security, defined by out-of-date features, presents difficulties in retrofitting security measures into older vehicles without compromising functionality.

To address these issues, a multidisciplinary approach involving collaboration among automotive manufacturers, cybersecurity experts, regulators, and the technology community is required. For navigating the ever-changing landscape of in-vehicle control network security, proactive security measures, continuous monitoring, and industry-wide information sharing are critical.

X. Conclusion

In conclusion, various security protocols and their effectiveness in addressing the challenges faced by in-car networks. It emphasizes IA CAN's resistance to Denial of Service (DoS) attacks such as flooding and starvation. In the presence of flooding attacks, IA CAN maintains a low and stable latency and allows ECUs to maintain listening frames during starvation attacks, which is beneficial for recovery planning.

Secure Connectivity Between Vehicles and External Devices, a three-step authentication protocol that enables secure integration of external devices with a vehicle's electronics, is also discussed in the paper. This protocol is designed to prevent unauthorized access and attacks while also ensuring that only authorized devices have access to the vehicle's systems. Authenticating the device, generating a session key from a pre-shared key and a random nonce, and using this session key to encrypt and decrypt data exchanged between the vehicle's electronics and the external device are all part of the process.

Overall, the conclusion emphasizes the significance of implementing effective security protocols to protect in-car

networks from various threats and vulnerabilities while preserving system functionality and performance.

XI. References

[1] P. Mundhenk, " Security for Automotive Electrical / Electronic (E /E) Architectures," *Cuvillier Verlag,* 2017.

[2] P. P. a. U. E. Dennis K. Nilsson, "Larson.Vehicle ECU Classification Based on safety-Security Characteristics," 2008.

[3] M. F. a. C. M. R. Buttigieg, "Security Issues in Controller Area Networks in Automobiles," in *international conference on Sciences and Techniques of Automatic Control & Computer Engineering*, 2017.

[4] V. G. a. J.-P. H. P. Papadimitratos, "Securing Vehicular Communications - Assumptions, Requirements, and Principles," in *in Workshop on Embedded Security in Cars (ESCAR)*, 2006.

[5] U. E. L. F. P. a. E. J. D. K. Nilsson, "A First Simulation of Attacks in the," in *First International Workshop on Computational Intelligence in Security for Information Systems*, 2008.

[6] A. R. P. K. S. H. Srivaths Ravi, "Security in embedded systems: Design challenges," in *ACM Transactions on Embedded Computing Systems (TECS)*, 2004.

[7] Y. Yang, Z. Duan and M. Teharipoor, "Identify a Spoofing attackon an in-vehicle CAN bus based on the deep features of an ECU figerprint signal," *smart cities,* vol. 3, no. 1, pp. 17-30, 2020.

[8] S. Hartzell, C. Stubel and T. Bonaci, "Security analysis of an automobile controller area network bus.," *IEEE Potentials ,* vol. 39, no. 2, pp. 19-24, 2020.

[9] A. Gazdag, C. Ferenczi and L. Buttyán, "Development of a man-in-the-middle attack device for the can bus.," *In Proceedings of the 1st Conference on Information Technology and Data Science, Debrecen, Hungary,,* no. 3, pp. 6-8, 2020.

[10] W. S. Y. S. Liu, "In-vehicle network attacks and countermeasures: challenges and future directions," *IEEE NETw,* vol. 4, pp. 50-58, 2017.

[11] K. J. H. J. M. P. D. L. V. l.-l. c.-t. W. Choi, "low-level communica-tion characteristics for automotive intrusion detection system," *IEEE Trans. Inf. Forensics Secur,* vol. 13, 2018.

[12] M. Wolf, "Security Engineering for Vehicular IT Systems," *Vieweg+Teubner,* vol. 4.

[13] A. W. C. P. M. Wolf, "Security in automotive bus systems," in *Proceedings of the Workshop on Embedded Security in Cars (ESCAR)'04*.

[14] A. M. J.M. Ernst, "LIN bus security analysi," in *44th Annual Conference of the IEEE Industrial Electronics Society*, 2018.

[15] Y. A. T. M. H. F. H. Y. K. H. S. U. H. H. J. Takahashi, "Automotive attacks and countermeasures on LIN-bus," in *, J. Inf. Process*, 2017.

[16] "NI, FlexRay Automotive communication bus overview,," vol. 1, 2019.

[17] "FlexRay Consortium, FlexRay communications system, protocol specification," in *FlexRay TM* , 2010.

[18] Y.-N. X. Y.-J.W. Meng-Zhuo Liu, "Research of authenticated encryption security protocol for FlexRay in-vehicle network," in *Int. J. Comput. Theory Eng*, 2018.

[19] Y. W. Z. chao Liu, "LM algorithm neural network predictive control of FlexRay bus system," in *J. Phys. Conf. Ser. 1267*, 2019.

[20] L. C. L. Huan, "FlexRay Vehicle network predictive control based on neural network," in *MATEC Web Conf.*, 2018.

[21] Q. W. Z. Z. X. He, "A survey of study of FlexRay systems for automotive net," in *Proceedings of 2011 International Conference on Electronic Mechanical Engineering and Information Technology*, 2011.

[22] P. N. M. A. M. A. A.R. Mousa, "Lightweight authentication protocol deployment over FlexRay," in *Proceedings of the 10th International Conference on Informatics and Systems*, NY USA , 2016.

[23] L. P. B. G. P.-S. Murvay, "Accommodating time-triggered authentication to FlexRay demands," in *Proceedings of the Third Central European Cybersecurity Conference*, 2019.

[24] G. H. H. Z. Q. Z. Z. Gu, "Security-aware mapping and scheduling with hardware co-processors for FlexRay-based distributed embedded systems," in *IEEE Trans. Parallel Distrib. Syst. 27* , 2016.

[25] J. P. R. Radhiga, "Design of FlexRay communication controller protocol for an automotive application," in *IEEE 9th International Conference on Intelligent Systems and Control* , 2015.

[26] S. F. S. Shreejith, "Extensible FlexRay communication controller for FPGA-based automotive systems," *IEEE Trans,* vol. 64, 2015.

[27] C. K. I. L. T. Lee, "High performance CAN/FlexRay gateway design for in-vehicle network," 2017.

[28] S. F. S. Shreejith, "Extensible FlexRay communication controller for FPGA-based automotive systems," *IEEE Trans,* vol. 64, 2015.

[29] N. A. T. A. S. K. D. Püllen, "Securing FlexRaybased in-vehicle networks,," in *Microprocess. Microsyst*, 2020.

[30] D. S. a. I. V. Anthony Van Herrewege, "Compatible Broadcast Authentication Protocol for CAN bus," 2011.

[31] L.-C. Bogdan Groza et al., "Lightweight Broadcast Authentication for Controller Area Networks,," *ACM Transactions on Embedded Computing Systems,* vol. 16, 2017.

[32] A. P. e. al, "Efficient authentication and signing of multicast streams over lossy channels,," in *Symposium on Security and Privacy*, 2000.

[33] Hendrik Schweppe et al., "" Car2X Communication: Securing the Last Meter - A cost-effective approach for ensuring trust in Car2X applications using in-vehicle symmetric cryptography,," in *IEEE Vehicular Technology Conference,*, 2011.

[34] H. S. e. al., "Car2X Communication: Securing the Last Meter - A cost-effective approach," 2002.

[35] G. A. a. C. M. Limbasiya T, " Secure Automotive Data Transmission Scheme for In-Vehicle Networks," in *Proceedings of the 23rd International Conference on Distributed Computing and Networking*, 2008.

[36] S. K. a. J. D. T. Hoppe, "Security threats to automotive CAN networksPractical examples and selected short-term countermeasures," *Reliability Engineering and System Safety*, vol. 96, 2010.

[37] H. Q. Luo F, "Security mechanisms design for in-vehicle network gateway," in *SAE technical papers*, 2018.