Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2023.0322000

Fortifying Security of LIN : Insights from Ultrasonic Sensor Data and its Influence on CAN Protocol in Practical Environments

INSU OH[1], YOONJI KIM[2],MD REZANUR ISLAM[3], MAHDI SAHLABADI, (SENIOR MEMBER, IEEE)[1], KANGBIN YIM*[1]

[1]Department of Information Security Engineering, Soonchunhyang University, Asan-si, Korea.
[2]Department of Mobility Convergence Security, Soonchunhyang University, Asan-si, Korea.
[3]Department of Software Convergence, Soonchunhyang University, Asan-si, Korea.

Corresponding author: Kangbin Yim (e-mail: yim@sch.ac.kr).

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the Convergence security core talent training business support program(IITP-2024-2710008611) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation) & This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT)(RS-2024-00346749) & This work was supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea Government (MIST) (No. 2022-0-01022, Development of Collection and Integrated Analysis Methods of Automotive Inter/Intra System Artifacts through Construction of Event-Based Experimental System) & This work was also supported by the Soonchunhyang University Research Fund.

ABSTRACT

Autonomous driving technologies are dependent on external sensors that transmit data through the Local Interconnect Network (LIN), an area of research that has not yet been thoroughly investigated. This study examines the LIN vulnerabilities within a vehicle, emphasizing the security threats posed by unauthorized data injections and how they affect the Controller Area Network (CAN). The study uses ultrasonic sensor data as a case study to explore the intricate relationship between LIN and CAN in a real-world automotive environment. To counter these security threats, a lightweight, reliable, deep learning-based Intrusion Detection System (IDS) is developed to detect and mitigate the risks. The feasibility of the research was examined on a testbed, and data was collected from a real-world environment based on practical scenarios. As a result, the study observed the effects of a LIN attack on CAN, the backbone of the vehicular network. LIN vulnerabilities can cause disruptions to the entire In-Vehicle Network (IVN) ecosystem. Therefore, a dedicated security mechanism is required to secure IVN regarding LIN data. This study introduces an IDS that considers these consequential effects. The study highlights the critical need for enhanced security in-vehicle systems to effectively mitigate significant vulnerabilities within the LIN protocol. This would safeguard the entire ecosystems of the In-Vehicle Network (IVN) and the Controller Area Network (CAN) against possible disruptions brought on by unauthorized data injections.

INDEX TERMS Ultrasonic Sensor, Automotive vehicle, Security, LIN, CAN, IDS.

I. INTRODUCTION

With the increase in Electronic Control Unit (ECUs) within vehicles, the weight and cost increased due to the need for numerous connecting wires [1]. To solve this problem, the CAN protocol emerged as a solution that offers high-speed data transmission [2]. However, using the CAN protocol for modules without high-speed capabilities may result in waste of resources and power. As a result, LIN protocols were developed to be used in various low-power communication modules such as doors, seats, steering wheels, air conditioning, window opening and closing, and ultrasonic sensors inside the vehicle [3], [4]. Although the LIN communication speed is slower than the CAN protocol, it consists of a single line, making it simple to configure at low cost [5], and communication between the ECU and the ultrasonic sensor operates via LIN [6]. However, similar to CAN protocols, LIN protocols are susceptible to malicious attacks. Attackers can approach the LIN bus and maliciously cause the ultrasonic sensor to not recognize obstacles or people when using autonomous parking or autonomous driving functions [7]. Therefore, it is essential to study the security of LIN protocols in IVN to prepare for security threat situations because some cyberattacks

[8] may paralyze the system within the vehicle. Given the limited number of existing analyses, there remains a paucity of comprehensive examinations on vulnerabilities in the LIN protocol, leading to areas that have yet to be explored.

The paper thoroughly investigated the potential impact of attacks on the LIN on the IVN's backbone, called CAN. Based on real vehicle data, we developed a testbed to directly understand the IVN and configure circuits through customized ECUs. The evaluation was performed on the testbed to ensure accurate and reliable results by efficiently collecting and injecting large amounts of data (secure assessment), considering all situational data, and implementing reliable deep learning approaches as a security mechanism. Data collection begins with a hardware communication evaluation to understand the mechanisms that support vehicle communication protocols in real-world scenarios. This understanding is then replicated in a controlled testbed environment to ensure safety and investigate potential vulnerabilities. Security assessments performed using insights from the testbed environment show that the impact of data injection on the testbed is negligible. This is due to the number of ECUs in the testbed setup compared to the real vehicle configuration. However, data from the LIN and CAN communication protocol are collected in the testbed environment based on the insights gained from the real vehicle. Afterward, a concise analysis is performed to identify appropriate features that can inform the development of effective countermeasure mechanisms.

In this paper, **Section II** provides a detailed exploration of the LIN specifications as described in ISO 17987. This section delves into an analysis of vulnerabilities within the LIN system and outlines the corresponding countermeasures. A detailed description of the experimental setup is presented in **Section III**. This includes details on the testbed's environmental configuration, describing the procedures for data collection from a real vehicle. **Section IV** focuses on the evaluation of LIN vulnerabilities. The section discusses the environmental setup aligned with the activation of ultrasonic sensors and the characteristics of the resulting data. Turning attention to **Section V** describes the generation of error data on the CAN on the vehicle's internal LIN bus. In **Section VI**, a countermeasure system for LIN is presented. This section details the application of a deep learning approach to improve security measures. Finally, **Section VII** encapsulates the study with a conclusive summary, summarizing the key findings and contributions of the research.

II. RELATED WORKS

This section provides a comprehensive overview of the LIN protocol by the ISO 17987 standard. Subsequently, it delves into the current phase of the security evaluation of LIN and examines its associated security mechanisms. These security mechanisms are categorized into two main components: authentication-based and intrusion detection-based. The authentication-based security mechanism involves verifying the identity of entities within the LIN communication network. On the other hand, the intrusion detection-based mechanism focuses on identifying and responding to unauthorized access attempts or suspicious activities within the network. Both approaches are discussed in detail, highlighting their respective limitations and advantages.

A. LIN PROTOCOL IN ISO 17987

According to the ISO 17987 [9], the fundamental transmission unit within the LIN protocol is a frame. Among these, the unconditional frame, serving as a standard, is commonly employed in the interaction process on the LIN bus. The frame structure of LIN divides the protocol into a header and a response, as illustrated in Fig. 1. The header comprises a break field indicating the frame's initiation, a SYNC field determining communication speed and frame synchronization by the master node, and a protected identifier (PID) field signifying a unique identifier for each message. On the other hand, the response segment includes substantial data corresponding to a unique ID, featuring a data field capable of holding up to 8 bytes of payload. The checksum field, serving error detection during transmission or validating messages, can be categorized into classical and improved checksums. The header is consistently transmitted by the LIN master, with the LIN slave executing functionalities based on the identifier value. In scenarios where both master and slave tasks are dispatched simultaneously in the header, the command is directed to the slave node. Conversely, when data is requested, the slave node includes the identifier value of the requested data via the PID, transmitting it to the corresponding slave node through the response frame.

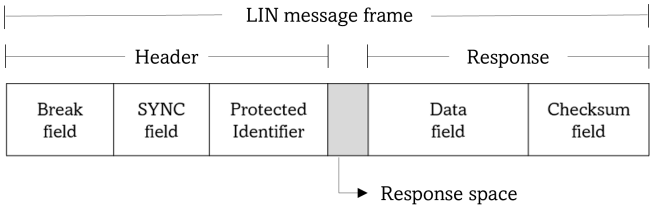


FIGURE 1: LIN Message Frame [9].

Distinguishing itself from the CAN network, the LIN protocol adopts a master-slave structure, where one master governs all communications within the network [10]. Unlike CAN, which communicates with all nodes through broadcasting, LIN accommodates up to 15 slaves under one master [11]. IVN, the primary ECU assumes the role of master, while sensors or actuators function as slaves. Moreover, with help of the LIN Description File (LDF), master node is capable of managing both master and slave tasks concurrently, whereas the slave node is restricted to handling slave tasks exclusively [12]. The LDF serves as the cornerstone component in detailing the entire network by delineating all the properties of the network bus [4]. LIN operates at data rates of up to 20 kbps and is configured over a single wire [13], making it conducive to cost-effective data bus implementation. However, it is important to note that LIN lacks a built-in security mechanism.

B. SECURITY ASSESSMENT ON LIN

In the modern automotive industry, there's a strong focus on creating self-driving technologies. These technologies depend on various external sensors to properly understand the environment and make decisions. However, despite considerable advancements, the efficacy of these systems is hindered by persistent limitations in vulnerability assessment and the deployment of effective countermeasures. Recent discussions [14]–[17] within the automotive community have underscored the pressing need for comprehensive vulnerability assessments pertaining to vehicular sensors.

The ISO 17987 standard provides guidelines for efficient communication in LIN protocol systems as mentioned earlier, yet it lacks specifications for security mechanisms, leaving LIN susceptible to integrity breaches and potential malicious attacks such as response collision attacks and message spoofing [14]. These vulnerabilities pose significant risks, including manipulation of critical vehicle functions like steering, brakes, and throttle control [14]. LIN protocols, unlike CAN, are particularly vulnerable to electrical noise interference, further exacerbating the risk of message spoofing attacks [15]. Studies indicate that compromised LIN bus masters or slave nodes can corrupt messages and generate false but seemingly valid packets, highlighting the ease with which LIN communications can be disrupted compared to CAN systems [15]. The study [16] investigated the vulnerability of ultrasonic sensors in autonomous vehicles to various attack scenarios, such as thin objects acting as obstacles, covering of the transmitter and receiver, use of acoustic foam, and interference from additional sensors. Defense mechanisms proposed include multi-sensor fusion and fast calibration of sensors at vehicle startup. The findings revealed significant inaccuracies in obstacle detection under attack conditions, highlighting the ease of compromising sensor functionality with low-cost methods. Recommendations for robust defense mechanisms, including encryption, anomaly detection, and authentication, are underscored as essential to mitigate these vulnerabilities effectively [10]. Cybersecurity analyses on vehicular sensors also emphasize the urgency of advanced countermeasures against various attack vectors such as jamming, spoofing, relay, and physical tampering [10]. Furthermore, investigations into LIN bus vulnerabilities underscore the absence of security features in the LIN standard, paving the way for attacks like master node attacks and frame injection [17]. These weaknesses underscore the necessity of implementing encryption and authentication mechanisms while balancing the challenge of maintaining security without compromising LIN's cost-effectiveness and simplicity advantages [17].

Most analyses and validations of hypotheses concerning LIN data often rely on simulated scenarios. However, translating these hypotheses into practical applications within real vehicles requires the utilization of authentic LIN data transmitted and received under genuine operating conditions. The study mentioned above conducted its analysis through a testbed where the data generation process and frequency followed a random pattern. Conversely, in a real environment, the data generation process adheres to LDF specifications, where data generation on LIN follows predefined payloads and frequencies. Given that external sensors operate in accordance with surrounding environments and predefined manufacturer rules, the incorporation of real-world scenarios becomes paramount for formulating effective security mechanisms.

C. COUNTERMEASURE MECHANISM FOR LIN

Several studies have addressed securing ultrasonic sensor data transmission through authentication and intrusion detection approaches. Authentication verifies data legitimacy, while intrusion detection identifies anomalies.

Multiple research investigations contribute insights into enhancing cybersecurity for vehicle LIN bus systems by authentication approach. Takahashi et al. (2017) investigate LIN vulnerabilities, emphasizing errors induced by false messages and collisions, proposing countermeasures like assigning significant data to response bytes and implementing MACs [18]. Additionally, Xu et al. (2018) identify vulnerabilities in ultrasonic sensors for autonomous vehicles, devising defense mechanisms such as Physical Shift Authentication (PSA) and Multiple Sensor Consistency Check (MSCC) to enhance sensor security, thereby mitigating risks posed by attacks like spoofing and jamming [19]. Paez et al. (2022) propose a defense mechanism against replay attacks, integrating Speck64 encryption and BLAKE2s hash-based message authentication code (HMAC) to ensure data integrity and authentication [20]. Their approach demonstrates high effectiveness in prototype testing. Oberti et al. (2022) introduce the LIN-MM strategy, utilizing signal modulation to integrate a Message Authentication Code (MAC) with LIN communications, effectively countering spoofing and response collision attacks [11]. Authentication mechanisms on LIN may introduce various challenges including increased computational overhead, system complexity, resource consumption, latency issues and vulnerability to attacks [21]–[24].

As a solution, IDS are better than authentication mechanism at finding and stopping a wide variety of network threats early. They offer strong protection against new types of cyber dangers, particularly when monitoring ongoing traffic. The study by Lee et al. (2019) [25] devises a mathematical model to secure ultrasonic sensors in vehicles against signal injection attacks, addressing jamming and spoofing threats. It determines the valid width range of received signals by considering the sensor's operation time, the distance to an obstacle, and the sensor's beam pattern. This model allows the system to differentiate between genuine and malicious signals by checking if the width of the received signal falls within the estimated valid range. The method's effectiveness is demonstrated through experiments, offering protection without the need for additional devices or substantial changes to existing systems. SoundFence (2021) [26] provides advanced protection against signal injection attacks on vehicle ultrasonic sensors through a two-phase approach, without needing extra equipment. First, it randomizes the sensor's pulsing period to

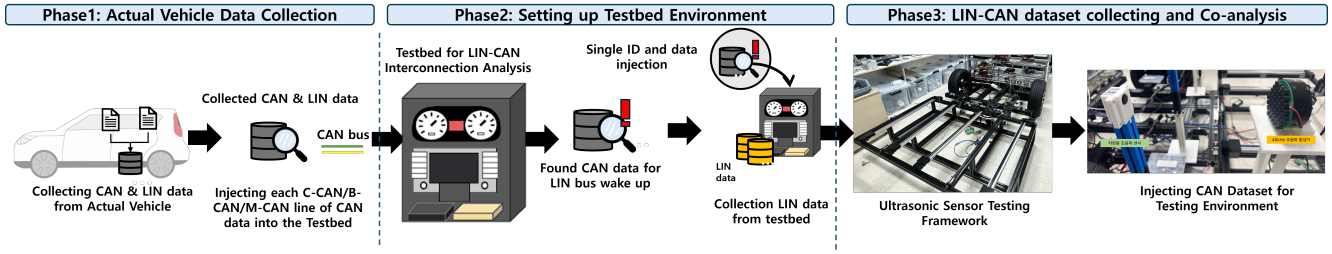


FIGURE 2: Process of Collecting LIN and CAN Dataset from Actual vehicle and Real vehicle environment.

prevent attackers from syncing spoofed signals, making such attacks less effective. This step doesn't affect data transmission frequency but makes it harder for attackers to predict sensor pulsing, leading to more failed attacks. Second, it analyzes side echo signals, using their physical properties to distinguish real signals from fake ones. By examining side echoes' patterns and attenuation, SoundFence identifies and blocks false signals, detecting over 95% of abnormal activities with few false positives. Aung et al. (2023) [27] introduce VNGuard, an intrusion detection system tailored for LIN and Automotive Ethernet in vehicles, transforming network data into images for deep learning analysis. It employs deep convolutional neural networks (DCNN) to detect anomalies and classify attack types, achieving over 96% accuracy for LIN and 99% for AE classification, with detection speeds of 3ms and 4ms respectively.

III. EXPERIMENTAL SETUP

The LIN-CAN dataset collection and analysis experiment consists of three steps: First, data is collected from the real vehicle to analyze the basic CAN and LIN data. Then, the testbed is set up, and finally, the LIN-CAN linkage analysis is performed through the testbed.

First, we collected LIN and CAN datasets based on ultrasonic sensors in an real vehicle. This allows us to identify potential attack vectors for local LIN in a testbed environment, and then testing them in an real vehicle is an important step in this research. This ensures that our findings are appropriate and applicable to real-world situations, contributes to developing effective security measures, and ultimately helps enhance the safety and security of automotive systems. While testbeds provide a controlled environment for the initial discovery and analysis of vulnerabilities, testing on real vehicles is essential to validate these results under real-world operating conditions. This ensures that the theoretical vulnerabilities can be exploited in real-world scenarios that vehicles encounter daily. The entire process of collecting CAN and LIN datasets from real vehicles and performing experiments on a testbed is shown in Fig. 2. Each phase is described in the A, B, and C subsections accordingly.

A. REAL VEHICLE DATA COLLECTION SETUP

IVN consists of LIN and CAN protocols, with the BCM as the intermediary between these two protocols. The vehicle, an SUV, is equipped with four ultrasonic sensors on its rear bumper, as depicted in Fig. 3, which communicate via LIN. Additionally, there are three CAN protocols [28] responsible for various functional operations: C-CAN (Chassis CAN), B-CAN (Body CAN), and M-CAN (Media CAN). Notably, C-CAN is responsible for the LIN association. LIN is responsible for low-power communication with ultrasonic sensors and smart windows, and among them, the ultrasonic sensor module used in the parking assistance system was selected as an analysis module because of its easy access and simple design [29]. In addition, the positions of the BCM and the ultrasonic sensor, which manage the body control field, were checked with reference to the circuit diagram provided by the vehicle manufacturer. The BCM is a IVN ECU responsible for managing and controlling various interior functions and systems, such as lighting, security, and comfort features.

FIGURE 3: Identify the LIN ID on the Target real Vehicle

To activate the sensors require direct current power and a LIN line for data monitoring. The PLIN-USB interface is used to monitor the data. The circuit configuration, as shown in Fig. 6, replicates a real-world scenario where CAN and LIN can be analyzed simultaneously. The Audio, Video, and Navigation (AVN) system in the center of the vehicle and the auxiliary glove box in the passenger seat were removed to gain access to the BCM. These components were then connected to the BCM, and a LIN line was tapped using a T-type wire connector, establishing a link with the PLIN-USB [30] for data monitoring purposes. It is crucial to note that most vehicles operate on a 12V [31] electrical system, the standard voltage for various electronic components, sensors, and lighting systems in modern cars [32]. To ensure compatibility and seamless operation, 12V direct current power was supplied through the 9-pin connector D-sub, using the

positive and negative terminals of the On-Board Diagnostics II (OBD-II) port. This approach ensures that the system operates exclusively when the vehicle is in use. In addition, to enable integrated analysis of CAN and LIN, the CAN line connected to the AVN head unit was tapped using the same T-type wire connector and the CAN line was secured to allow simultaneous monitoring by connecting CAN and LIN.

B. SETTING UP TESTBEDS ENVIRONMENT

To thoroughly analyze the operation and characteristics of a LIN, gathering various data points, including the status of the LIN protocol, the timing of message transmission, errors, collisions, real data values and identifiers, is imperative. However, conducting such experiments directly on a real vehicle poses potential risks. Therefore, initial experimentation is conducted in a controlled environment before implementation in a real vehicle. During the initial testing phase, a simulated setup is utilized to mimic the conditions of a real vehicle. This involves selecting components such as the ultrasonic sensor module and configuring the environment to closely resemble a real vehicle. A minimum of three ECUs are necessary to carry out these experiments. These include the Integrated Control Unit (ICU) [30], responsible for managing the CAN gateway within the vehicle; the BCM, which oversees systems such as the electric motor system, parking assistance, and tire pressure monitoring; and finally, a vehicle ultrasonic sensor. By adopting this approach, potential risks associated with conducting experiments in a live vehicle are mitigated. Moreover, it allows for a comprehensive understanding of the behavior of the LIN protocol and facilitates the identification of any issues or optimizations needed before implementation in real-world scenarios.

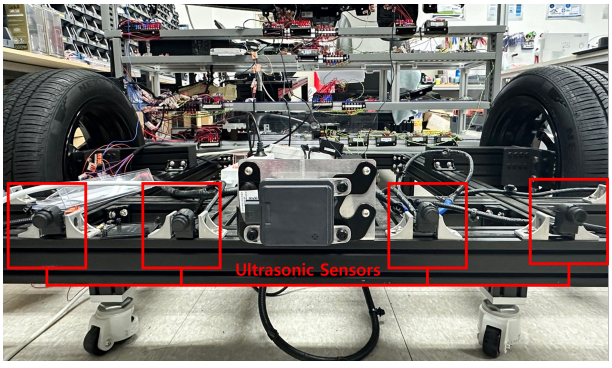


FIGURE 4: Array of Ultrasonic Sensors on a Vehicle testbed.

Fig. 4shows an environment similar to a scaled-down vehicle model with an ultrasonic sensor operating according to specific distance parameters. The circuit diagram shown in Fig. 5 illustrates the structure used to construct the testing environment. This diagram supplies power to each ECU using a 12V-5A DC power source for every ultrasonic sensor. Furthermore, an experimental setup was created by physically connecting the BCM, responsible for communication with the ultrasonic sensor, and the ICU, communicating with the BCM using jumper cables. An environment was established so that the ICU and the BCM could communicate using the P-CAN view to monitor the CAN. In addition, the setup was configured to utilize the P-LIN view program to monitor the LIN communication between the BCM and the ultrasonic sensor. By establishing this integrated environment, simultaneous monitoring and analysis of CAN and LIN communications became possible.

On the other hand, LIN data requires CAN data from the R-end to activate the ultrasonic sensor data. Therefore, we initially collected CAN data from the R-end of a real vehicle and injected it into the testbed. However, since this injection method can also be viewed as CAN replay data, there is a risk of recognizing attack data during data preprocessing and training. Therefore, we chose to find a single ID and data that activate LIN data based on the data collected in the real world environment. Using the Exhausted Searching method, we loaded the CAN data collected from the real car and injected it into the CAN bus line by line, and found the ID: 0x43F, Data: 0x0F 0x47 0x60, 0xFF, 0x55, 0x00, 0x00, 0x00, 0x00 that activates the LIN data. Since these IDs and data do not exist in the testbed, injecting them does not create a conflict. Therefore, we activated the LIN data by injecting these IDs into the CAN bus of the testbed and proceeded to collect data after activation. Also, after injecting these IDs and data, the cluster changed to the R stage, so we could infer that the data are related to the R stage. Furthermore, even if the R stage is not functional, we can confirm that the LIN data will be displayed as soon as the R stage appears in the cluster.

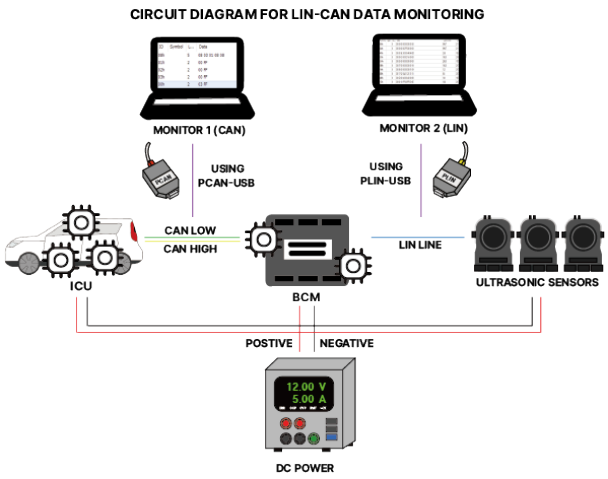


FIGURE 5: CAN-LIN Diagram for Construction in Testbed

C. LIN-CAN DATASET COLLECTING AND CO-ANALYSIS FROM THE TESTBEDS

We collected sample datasets for LIN-CAN correlation analysis based on the developed testbed. The dataset was extracted directly from the testbed, and the detailed dataset information is shown in the table 1. Dataset is collected with obstacles located at primary (30 cm or less), secondary (31-60 cm) and tertiary (61-120 cm) distances per manufacturer-provided

distance, and the collection conditions must include data for primary, secondary and tertiary distance alarms. The dataset scenario also collects 200 seconds of primary distance obstacles, 200 seconds of secondary distance obstacles, and 200 seconds of tertiary distance obstacles to include data from primary, secondary, and tertiary distances for 600 seconds of LIN data. For CAN, we collected datasets based on the scenarios under the same circumstances as for LIN and utilized them for our analysis.

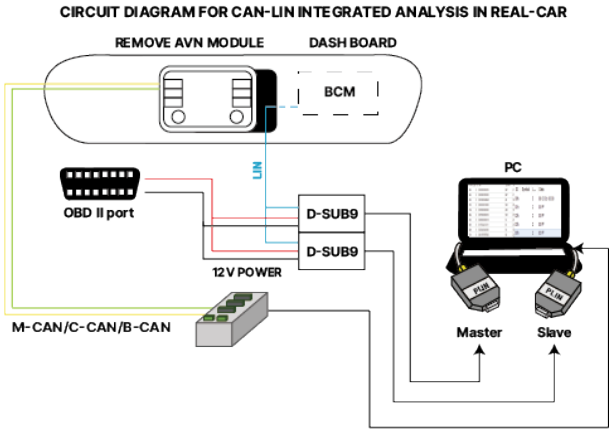


FIGURE 6: CAN-LIN Environment Construction in Real Vehicle Diagram

IV. EVALUATING THE LIN VULNERABILITIES

The experiment was carried out to identify the ID of the target vehicle ultrasonic sensor through data analysis and to determine in what form the LIN data are transmitted and received. This test was conducted to help understand sensor and data transmission and reception, and this section creates a LIN data collection scenario to identify the data and derive the possibility of an ultrasonic sensor attack.

A. SETTING UP THE ENVIRONMENT ON VEHICLE

To analyze the really collected LIN data, experiments were performed in a real vehicle environment. Recently, commercial vehicles have been equipped with ultrasonic sensors in the front, sides, and rear to provide autonomous parking and stopping assistance systems to drivers. However, in the vehicles selected in this paper, the study was conducted in vehicles equipped with ultrasonic sensors only in the rear bumper. The LIN protocol is divided into two modes: wake-up mode and sleep mode. All nodes are in wake-up mode, but to prevent unnecessary power waste, the LIN bus operates in sleep mode when the bus is left in a constant state. Therefore, since there is an ultrasonic sensor in the rear bumper, to activate the LIN node and check the LIN data, the gear must be set to reverse mode (R) and enter the wake-up state. If the driver moves backward, an alarm sounds and the safe distance is indicated based on the detected distance level. A total of 5 IDs and data were collected in real time, and when in gear R, the IDs had values of 0x08, 0x00, 0x01, 0x02 and 0x03, respectively.

A series of experiments were conducted to evaluate the effectiveness of the sensor coverage method, which is utilized as an attack simulation on ultrasonic sensors before identifying their IDs based on location. All four ultrasonic sensors were initially obstructed and obstacle detection was performed to assess its feasibility. Subsequently, three of the four ultrasonic sensors were blocked, while one remained unblocked to facilitate identification of its LIN ID, and the process continued for all sensors. Data were monitored by positioning obstacles of approximately 120 cm height within each detection range in front of the ultrasonic sensor. These detection ranges were classified into three levels: level 1 (61-100cm), level 2 (31-60cm) and level 3 (less than 30cm), providing respective warnings to the driver regarding safe distances. LIN IDs identified for each sensor location are presented in Table 1, with the corresponding sensor coverage indicated. This analysis helps to understand the relationship between sensor location and LIN ID, which is crucial for payload analysis and security enhancement strategies.

TABLE 1: Identify the LIN ID on the target vehicle

LIN ID	Sensor 1	Sensor 2	Sensor 3	Sensor 4
0x00	Open	Blocked	Blocked	Blocked
0x01	Blocked	Open	Blocked	Blocked
0x02	Blocked	Blocked	Open	Blocked
0x03	Blocked	Blocked	Blocked	Open

B. LIN DATA ANALYSIS

The experimental process begins with the implementation of sensor coverage and data monitoring procedures. In this context, all sensors except one are intentionally covered to observe the payload of LIN IDs. The goal is to check each LIN ID and its payload one at a time. Five individual LIN IDs can be used to monitor one master node and four ultrasonic sensors and are presented with the identifiers 0x00, 0x01, 0x02, 0x03 and 0x08. It should be noted that LIN ID 0x08 is explicitly assigned to the BCM, which acts as the master controller for the ultrasonic sensor. This experiment aims to contribute to the effective management and operation of sensor networks by establishing a comprehensive understanding of the connections between LIN IDs and individual sensors.

Table 2 provides insight into the distribution of different LIN IDs across various payload values (0x00FF, 0x07FF, 0x01FF, and 0x03FF) corresponding to different distance categories. The data clearly show that each LIN ID corresponds to specific payload values depending on the distance. Specifically, payload values for 0x07FF are assigned within the range of 0-30 cm, payload values of 0x03FF are assigned within the range of 31-60 cm, and payload values of 0x01FF are assigned at distances exceeding 60 cm (61-120 cm).

Additionally, among the four rear ultrasonic sensors shown in Fig. 3, it was observed that side sensors one and four could not detect objects within the range of the first level 61-100 cm. The analysis confirmed that sensors two and

TABLE 2: Payload Across Distances for Different LIN IDs

LIN ID	0-30cm	31-60cm	61cm-120cm
0x00	0x07FF	0x03FF	0x00FF
0x01	0x07FF	0x03FF	0x01FF
0x02	0x07FF	0x03FF	0x01FF
0x03	0x07FF	0x03FF	0x00FF

three function together to measure objects within this range, with a consistent payload of 0x01FF. In addition, all sensors were activated during the 0-30 cm and 31-60 cm ranges, with fixed payloads of 0x07FF and 0x03FF, respectively. This understanding of sensor behavior helps optimize sensor deployment and interpret data effectively.

The graph in Fig. 7 illustrates the relationship between LIN ID and the time gap between two packets within a LIN communication network. Each data point on the graph represents a specific LIN ID. The graph considers five distinct data groups to analyze the time gap characteristics. It provides valuable insights into how the LIN ID values are related to variations in time gap differences, which can help us understand the behavior and potential patterns of the network. The graph's Fig. 7 y-axis represents time in microseconds, while the x-axis displays the LIN IDs. Specifically, LIN ID 0x00 appears only at 60000-microsecond intervals, and LIN IDs 0x02 and 0x08 appear at 10000-microsecond intervals. LIN IDs 0x01 and 0x03 are also found at 10000 and 60000 microsecond intervals.

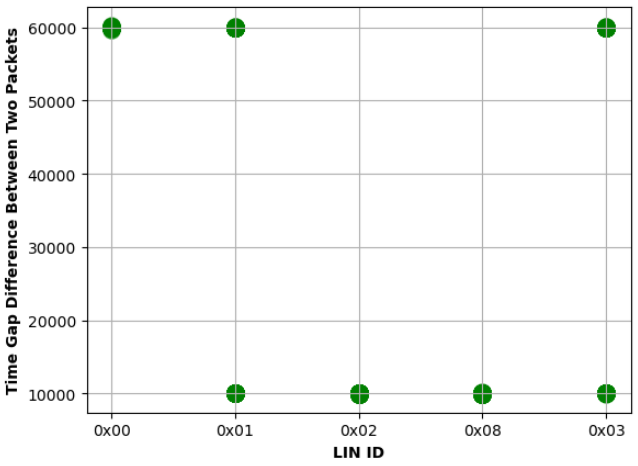


FIGURE 7: LIN ID vs. Time Gap Difference Between Two Packets

C. RELATION OF CAN AND LIN

CAN and LIN are essential protocols used inside the vehicle and are used in conjunction with each other, as shown in Fig. 8. When CAN requests data from low power actuators or sensors, it receives the necessary data through LIN communication and processes tasks. The connection between CAN and LIN plays an important role in improving the safety and efficiency of the vehicle and is used to coordinate communication between various devices of the system [33].

For example, in a parking assistance system, when the BCM needs information about the distance to an obstacle, the BCM requests information from the slave node. When an information request is received, the slave node emits ultrasonic waves and measures the return time from the obstacle to the ultrasonic sensor when it contacts an obstacle. This information is transmitted through LIN communication, and the BCM that obtains the distance information is transmitted to the cluster through CAN communication to provide the driver with an alarm sound or visual information based on the distance between obstacles.

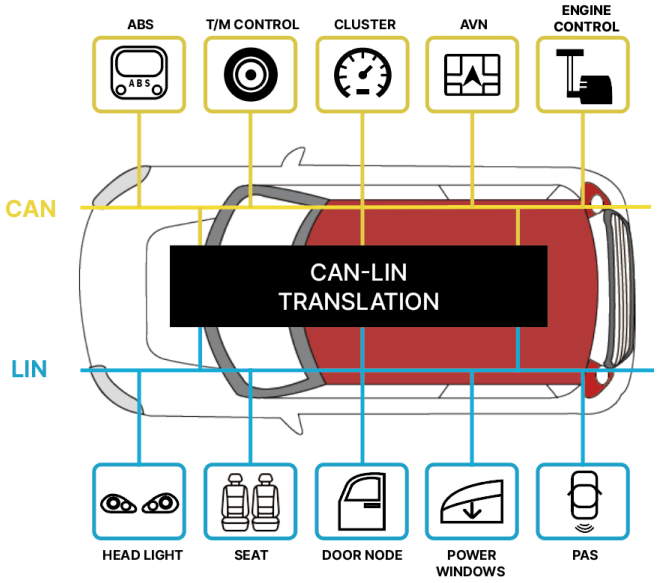


FIGURE 8: Affected Sensors and Actuators Connected to Both LIN and CAN

Consequently, the key aspect of this experiment in analyzing the relationship between CAN and LIN is the ability to examine both protocols simultaneously. An experiment was conducted to identify the CAN ID associated with the ultrasonic sensor and verify whether the CAN data can be altered through LIN changes. CAN data was collected for about 1 minute while the gear was in stage R, divided into cases with and without obstacles. When there were no obstacles, payloads with the values '0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00' were collected every 100 ms for data with CAN ID 0x510. When an obstacle was installed in front of the ultrasonic sensor, ID 0x510 appeared consistently every 100 ms, and it was confirmed that the payload value changed at each stage of the obstacle. Therefore, it was found that the ID related to the ultrasonic sensor was 0x510.

Obstacle warnings: 1 (61-120cm), 2 (31-60cm), 3 (0-30cm) using ultrasonic sensors - Sensor One (LIN ID: 0x00), Sensor Two (LIN ID: 0x01), Sensor Three (LIN ID: 0x02) and Sensor Four (LIN ID: 0x03) for approximately 83 seconds. Data were

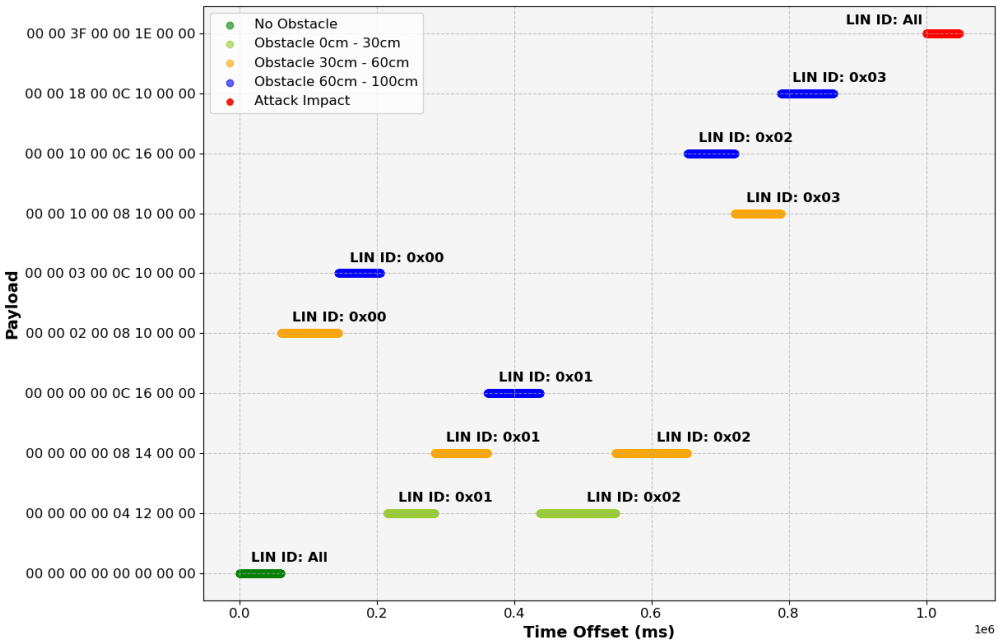


FIGURE 9: Payload Distribution on CAN Networks on ID 0x510: A Comparative Analysis Under Different Condition on LIN

collected in three steps. As a result of the measurement, the CAN data changed depending on the distance of the obstacle, and the CAN data payload results corresponding to each LIN ID are shown in Fig. 9. Additionally, when the gear was in gear P (Parking), gear D (Driving), and gear R (Reverse) but there were no obstacles, the CAN data payload value was confirmed to be '0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00'.

V. EXPERIMENT RESULT

This section describes the generation of error data on the CAN by fabricating injection tools for message inject ability tests and sending externally generated LIN data to the vehicle's internal LIN bus.

Data is collected with obstacles located at primary (30 cm or less), secondary (31-60 cm), and tertiary (61-120 cm) distances per manufacturer-provided distance, and the collection conditions must include data for primary, secondary, and tertiary distance alarms. The dataset scenario also collects 200 seconds of primary distance obstacles, 200 seconds of secondary distance obstacles, and 200 seconds of tertiary distance obstacles to include data from primary, secondary, and tertiary distances for 600 seconds of LIN data.

A. DEVELOPMENT OF A LIN DATA INJECTION TOOL

In order to test for potential attacks against CAN-LIN linkage protocols, the message injection possibility must be tested first. Therefore, an injection tool is needed to inject invalid data into the real LIN protocol. The message injection possibility test is first performed in the test testbed environment for safety, and then if the message injection test is successfully performed, the test is performed in the real-vehicle. The LIN

TABLE 3: Specifications of LIN Collection and Injection devices and software

Features	Description
Manufacturer	PEAK system
Hardware Device	PLIN-USB
Standard	LIN connection (ISO 17987)
Hardware Spec	LIN interface for the USB connection
Software	PLIN-View Pro
API	PLIN-API
System	Windows11(x64/ARM64),10(x86/x64)
Function	Display of incoming LIN frames

data injection tool was created using PLIN-API provided by Peak System, which can be divided into master mode and slave mode by using Python and adjusting the parameter values. The LIN communication speed is defined in the range of 1-20 kbps, and the communication speed of the LIN used in real-vehicles is mainly 19.2 kbps. Therefore, the injection tool is set to be injected at a speed of 19.2 kbps in the master mode, and a procedure to connect the LIN bus to PLIN-USB hardware is required to help PC connection.

The provided Algorithm 1 outlines an experimental setup for testing LIN message injection using the PLIN API. It begins by registering a client and initializing hardware. Then, it creates injection tests by iterating over a list of messages containing frame IDs, data, and checksums, constructing LIN message objects accordingly. These messages are stored for later use. The Algorithm 1 then enters a loop to send these stored messages continuously for a fixed number of iterations (500 in this case), with a 0.1-second delay between transmis-

Algorithm 1: Experimental Layout on LIN Injection

```
Data: List of messages with frame_id, data, checksum
Result: Write results
Step 1: Register Client;
Step 2: Hardware Identify;
Step 3: Connect the Client;
Step 4: Initialize the Hardware;
Step 5: Create Injection Tests;
Initialize: injection_tests        ▷ List to store TLINMsg objects;
for each (frame_id, data, checksum) in messages do
    injection_test ← new TLINMsg();
    injection_test.FrameId ← frame_id;
    injection_test.Length ← len(data);
    injection_test.ChecksumType ← TLIN_CHECKSUMTYPE_ENHANCED;
    injection_test.Direction ← TLIN_DIRECTION_PUBLISHER;
    injection_test.Data ← data;
    injection_test.Checksum ← checksum;
    injection_tests.append(injection_test)        ▷ Add message to the list
Step 6: Send Messages;
for i in range(500) do
    for each injection_test in injection_tests do
        wResult ← LIN.Write(hClient, hHw, injection_test);
        if wResult == TLIN_ERROR_OK then
            Print: Write Success! Frame ID: 0x + injection_test.FrameId;
        else
            Print: Error: + wResult;
        time.sleep(0.1);
```

sions. For each transmission attempt, it prints whether the write operation was successful, the frame ID, or the error code if unsuccessful. The 0.1-second time interval between injections aims to prevent data collisions. This successful message injectability test enables attacks such as Denial of Service (DoS), Replay, and Fuzzing, where large amounts of data can be injected or data fields manipulated with custom values.

B. ANOMALY ANALYSIS ON LIN AND CONSEQUENTIAL EFFECTS ON CAN

A total of 4 data payloads were being collected for LIN ID 0x08, and among them, data payloads of '0x0E, 0x03, 0x01, 0x04, 0x0E' of LIN ID '0x08' were injected using a data injection tool in master mode. Data injection is implemented considering all conditions, including no obstacles, obstacles at 0-30 cm, obstacles at 31-60 cm, and obstacles at 61-120 cm, for both normal and attack situations. A change in the data payload of CAN ID (0x510) related to the ultrasonic sensor was confirmed due to LIN data payload injection shown in Fig. 9 and Fig. **??**. CAN data collection was conducted in a real vehicle environment. When the gear was in reverse, an obstacle was placed in front of the ultrasonic sensor and an injection test of the LIN data payload was performed for about 20 seconds through the LIN injection tool and monitored CAN and LIN simultaneously and collect data in real time.

Upon analyzing the CAN data, it was noted that injecting LIN data payloads in the LIN master node resulted in significantly different generated data compared to the payload collected under normal circumstances. The CAN data was confirmed to have the following value: '0x00, 0x00, 0x3F, 0x00, 0x00, 0x1E, 0x00, 0x00', as depicted in Fig. 9 and Fig. **??** illustrating the impact of the attack. An analysis was conducted on data collected in both LIN master and slave modes using the same method. In the master mode, along with the injection, three additional invalid IDs (0x15, 0x28, 0x3E) and varying payloads were created and transmitted to the slave, resulting in synchronization errors and slave response errors. Fig. **??** illustrates the difference in LIN data between the normal state and the attack state on the slave node concerning the IDs. Specifically, 500 specific data payloads were injected with ID 0x08, revealing various payloads for LIN ID 0x08. In the normal state, five IDs and data payloads were transmitted and collected at regular intervals on the slave node. However, upon commencement of the injection, additional LIN IDs (0x0E, 0x05, 0x15, 0x28) and distinct payload volumes were observed on the slave node, with the amount of data for each ID approximately doubling compared to the normal state.

During the experiment, manipulation of both CAN data and LIN data was achieved through the injection of a data payload on LIN. Subsequently, it was observed that as an obstacle approached the ultrasonic sensor, the customary warning sound within the vehicle failed to activate, and the obstacle directly in front remained unrecognized. Success in injecting LIN data demonstrated the capability to disable the functionality of the ultrasonic sensor. Furthermore, injecting a substantial volume of data payloads with specific LIN IDs revealed the potential to alter CAN data and disrupt the CAN protocol system. This raises significant concerns within the self-driving automotive industry, as it implies a vulnerability that could compromise the proper functioning of autonomous vehicles.

VI. COUNTERMEASURES

A. FEATURE EXTRACTION

In other words, when parking, the car always moves from the first alarm to the third alarm. In Chapter 4, data patterns in normal situations were analyzed to understand this series of order by referring to the CAN data derived for each distance step for CAN ID 0x510.

As a result of data analysis, CAN data in normal state when parking is shown in Figure. Starting with '00 00 00 00 04 12 00 00' representing the first alarm alarm, '00 00 00 00 08 14 00 00' representing the second alarm alarm, '00 00

02 00 08 14 00 00' data, and the third alarm. '00 00 02 00 0C indicates an alarm alarm. The data flow was observed in the following order: 16 00 00', '00 00 12 00 0C 16 00 00', and '00 00 10 00 0C 16 00 00'. Additionally, it is assumed that the data representing the distance of an object generate a warning alarm based on the 5-byte and 6-byte values of the data payload.

The first replay attack is an attack that replays data containing the pattern '0x09' among the LIN data derived from the physical error injection test to the LIN bus. When LIN data containing the 0x09 pattern are replayed, an 8-byte data payload with a CAN ID of 0x510 is completely filled with 0x00. The pattern of CAN data generated through the first LIN replay attack is shown in Figure V-1-38.

Result of data analysis, after the first warning sound was sounded, the forms of '00 00 00 00 00 00 00 00', '00 00 00 00 00 10 00 00', and '00 00 3F 00 00 1E 00 00' were extracted, indicating that the LIN data were replayed at a distance, and the secondary. It was confirmed that the same pattern was applied.

In addition, the normal data showed a pattern of the 1st alarm, 2nd alarm, and 3rd alarm sequence, but compared to the normal data, in the first replay attack, 1st alarm → attack → 2nd alarm → attack → 3rd alarm → An unusual pattern of attacks was observed. This means that even if the data looks like normal data, it can be considered an attack because the pattern appears differently.

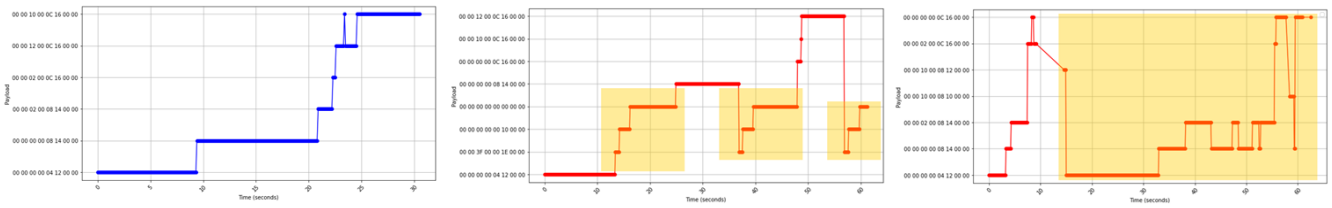
The second replay attack is an attack that replays LIN data generated under normal conditions in the order of the first alarm, second alarm, and third alarm. The CAN data pattern according to the normal LIN data replay result is as shown in. Until about 10 seconds ago, the same pattern observed in normal situations appeared in the order of the first alarm, second alarm, and third alarm, but 5 seconds after the third alarm sounded, the second alarm sounded once. An abnormal pattern was observed starting immediately from the first alarm sound.In a real situation, assuming that the object is stationary, it is physically impossible for the object to move away from the closest position in 5 seconds. This is an abnormal pattern that differs from normal sensor data flow, and suggests the possibility of data manipulation caused by a replay attack. In addition, the results shown in the second replay attack can be judged as a sign of a replay attack because the pattern shows an alarm sequence that is different from the normal state even if the CAN data form appears normal.

B. INTRUSION DETECTION

The modern self-driving automotive industry is based on a complex network of sensors, internal communication systems such as the IVN, and various data transmissions [34]. However, the vulnerability of these systems poses a significant threat to their reliability and security. In particular, the LIN protocol has been investigated for its potential security weaknesses and their impact on the CAN. Since IVN is a resource-constrained system, introducing an additional IDS specifically for LIN would increase the complexity and power consumption. To address this, we propose an integrated IDS that operates within the existing CAN IDS framework, providing security for both CAN and LIN networks. Our approach leverages a lightweight deep learning model with time-series prediction capabilities. Specifically, we employ a supervised Gated Recurrent Unit (GRU) model as an effective Intrusion Detection System (IDS) for LIN, due to its ability to accurately classify and detect anomalies in time-series data. Another issue is that CAN IDSs perform according to the input feature, and some IDSs use only specific features such as CAN ID, time gap, or payload [35], but the impact on CAN from LIN attacks is shown primarily in the payload. As a result, existing CAN IDSs are unable to detect attacks on LIN without proper labeling. Therefore, a dedicated IDS customization is required to secure LIN through CAN.

The dataset used for training and evaluation consists of three categories: attack-free, fuzzing attack, and replay attack, with a total of 552,634 attack-free samples, 4,089 fuzzing attack samples, and 1,777 replay attack samples. For validation, 100 samples were selected, while the test dataset comprised 1,000 samples, evenly divided between attack-free and attack instances (500 each). The LIN attack specifically impacts the payload of CAN IDs in the "0x0510" range, leading to a significantly larger number of attack-free samples compared to attack samples within the same 10-minute observation period. Since this is a binary classification task, the compilation of chunks was applied to segment the data into fixed-size sequences, ensuring effective learning of temporal dependencies. Each chunk, defined by a specified gap size, was assigned a label of 1 if any attack was present within the sequence; otherwise, it was classified as 0 (attack-free), maintaining a structured input format for training and evaluation. To enhance feature representation, CAN ID, Payload (eight-byte hexadecimal number), and time gap were used as input features. The CAN ID and payload were first encoded using a label encoder, followed by Min-Max scaling applied to the entire dataset, ensuring effective normalization. This preprocessing step improved the model's ability to capture attack patterns while preserving dataset integrity.

The IDS is implemented using a binary classification GRU model, optimized to detect anomalies in the LIN communication network through CAN. The model is configured with an input size of 10, representing key features such as the CAN ID, Payload (eight-byte hexadecimal values), and Time Interval Between Messages. A sequence length (chunk) of 10 is used to capture temporal dependencies, with the chunk size determined through a wrapper method that evaluated values of 5, 10, 15, 20, 25, and 30. Considering response time (8 ms approximate) as a key factor, the 10-chunk configuration provided the best balance between detection performance and computational efficiency. The architecture, as shown in 4, consists of two GRU layers, each with a hidden size of 128 and a dropout rate of 0.3 to avoid overfitting, with predictions generated through a single fully connected layer followed by a linear output layer. The training process was conducted using the Adam optimizer and a Binary Cross-Entropy loss function



(a) Attack Free (b) Replay Situation One (c) Replay Situation Two

FIGURE 10: Confusion Matrix on High-Frequency Anomaly Detection

over 300 epochs, with early stopping applied after 93 epochs. The model was trained with a batch size of 32 and a learning rate of 0.001, ensuring an efficient and robust IDS capable of classifying LIN communication patterns while maintaining low latency.

TABLE 4: Deep Learning Hyperparameters

Hyperparameter	Value
Model Type	GRU
Input Size	10
Sequence Length	10
Number of GRU Layers	2
Hidden Size (GRU)	128
Dropout	0.3
Fully Connected Layers	1
Output Layer	Linear
Loss Function	Binary-Cross-Entropy
Optimizer	Adam
Number of Classes	2
Number of Epochs	300
Training Patience	93 Epochs (Early Stopping)
Batch Size	32
Learning Rate	0.001

The performance of the model in the detection of LIN intrusion is visually represented in Figure 11, which illustrates the confusion matrix. The results show that 499 attack-free instances were correctly classified, with only one false positive (misclassified as an attack), achieving 99.80% classification accuracy for this class. Similarly, of 500 real attack cases, 486 were correctly identified, with 14 false negatives, resulting in a 97.20% recall for attack detection. These results emphasize the model's robustness in distinguishing between normal and attack conditions, ensuring a high detection rate with minimal misclassification. The classification report and performance metrics, presented in Table 5, further validate these findings. The model achieved a precision of 0.97, recall of 1.00, and F1-score of 0.99 for the 'Attack Free' class, while the 'Attack' class attained 1.00 precision, 0.97 recall, and 0.98 F1-score. The overall accuracy was 0.98, with macro and weighted averages of 0.99 for precision and 0.98 for recall and F1-score. Additionally, the ROC AUC score reached 0.9984, indicating a near-perfect classification capability. The exceptionally low false positive rate of 0.0020 highlights the model's reliability in using CAN data to detect LIN-based anomalies with minimal false alarms. By leveraging CAN data, the model effectively secures LIN communication, accurately identifying and classifying potential intrusions within the vehicle network. This demonstrates the model's capability to enhance in-vehicle security by bridging CAN and LIN data analysis, ensuring robust intrusion detection across both communication protocols, which shows the potential integration of CAN IDS for securing LIN.

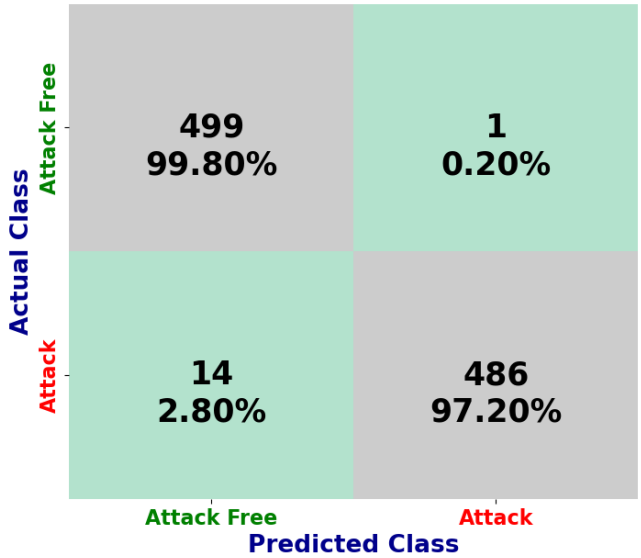


FIGURE 11: LIN Intrusion Detection Confusion Matrix

TABLE 5: Classification Report and Performance Metrics

Class	Precision	Recall	F1-Score
Attack Free	0.97	1.00	0.99
Attack	1.00	0.97	0.98
Accuracy		0.98	
Macro Avg	0.99	0.98	0.98
Weighted Avg	0.99	0.98	0.98
ROC AUC Score		0.9984	
False Positive Rate		0.0020	

VII. CONCLUSION

The research conducted has brought into focus the potential weaknesses of the LIN protocol, which is used in automotive systems. The study highlights the significant impact of these vulnerabilities on the security of CAN networks. It highlights the need for advanced protective measures against cyber threats, and the deployment of a deep learning-based IDS is a notable step towards mitigating unauthorized data injections and enhancing automotive cybersecurity. Testbed and Real-world experiments have shown that LIN message injections can compromise vehicle safety and CAN data integrity, which underlines the interconnectedness of LIN and CAN systems and the critical need for security measures. The proposed LIN IDS, with its 100% detection rate, is an effective defense mechanism that sets a precedent for future research to fortify automotive networks against evolving cyber threats. By taking proactive measures to secure our vehicles, we can ensure the safety of drivers and passengers and protect against malicious attacks.

VIII. DISCUSSION AND FUTURE WORK

This paper confirmed that only one LIN ID of the target model vehicle affects CAN, and it is expected that this method can be applied to other models of vehicles by utilizing the same method. In addition, the influence of LIN data on CAN data among the three types of CANs in the vehicle's internal network was analyzed based on the most important and core C-CAN network. However, it is expected that other M-CAN and B-CAN can also be influenced because they provide other functions related to ultrasonic sensors. Therefore, it is expected that future research can build a hybrid IDS system that utilizes CAN IDS, with a lot of background research needed to build IDS for LIN.

References

[1] H. Vdovic, J. Babic, and V. Podobnik, "Automotive software in connected and autonomous electric vehicles: A review," *IEEE Access*, vol. 7, pp. 166 365–166 379, 2019.

[2] M. Di Natale, H. Zeng, P. Giusto, and A. Ghosal, *Understanding and using the controller area network communication protocol: theory and practice*. Springer Science & Business Media, 2012.

[3] H. Chen, J. Liu, J. Wang, and Y. Xun, "Towards secure intra-vehicle communications in 5g advanced and beyond: Vulnerabilities, attacks and countermeasures," *Vehicular Communications*, vol. 39, p. 100 548, 2023.

[4] G. Funes, M. Siller, and J. Horta, "Modelling, simulation, and performance analysis of intra-vehicular heterogeneous networks," *Wireless Personal Communications*, pp. 1–55, 2024.

[5] S. Jadhav and D. Kshirsagar, "A survey on security in automotive networks," in *2018 Fourth international conference on computing communication control and automation (ICCUBEA)*, IEEE, 2018, pp. 1–6.

[6] S. Jeong, C. Choi, J. Oh, *et al.*, "Low cost design of parallel parking assist system based on an ultrasonic sensor," *International Journal of Automotive Technology*, vol. 11, pp. 409–416, 2010.

[7] C. Yan, W. Xu, and J. Liu, "Can you trust autonomous vehicles: Contactless attacks against sensors of self-driving vehicle," *Def Con*, vol. 24, no. 8, p. 109, 2016.

[8] A. Anwar, A. Anwar, L. Moukahal, and M. Zulkernine, "Security assessment of in-vehicle communication protocols," *Vehicular Communications*, vol. 44, p. 100 639, 2023.

[9] I. SC31, *Iso 17987-1: 2016,"road vehicles-local interconnect network (lin)-part 1: General information and use case definition"*.

[10] Z. El-Rewini, K. Sadatsharan, D. F. Selvaraj, S. J. Plathottam, and P. Ranganathan, "Cybersecurity challenges in vehicular communications," *Vehicular Communications*, vol. 23, p. 100 214, 2020.

[11] F. Oberti, E. Sanchez, A. Savino, F. Parisi, M. Brero, and S. Di Carlo, "Lin-mm: Multiplexed message authentication code for local interconnect network message authentication in road vehicles," in *2022 IEEE 28th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, IEEE, 2022, pp. 1–7.

[12] S.-H. Kim, S.-H. Seo, J.-H. Kim, *et al.*, "A gateway system for an automotive system: Lin, can, and flexray," in *2008 6th IEEE International Conference on Industrial Informatics*, IEEE, 2008, pp. 967–972.

[13] "LIN Specification Package Revision 2.1." (2006), [Online]. Available: http://www.lin-subbus.org.

[14] P. Guo, H. Kim, L. Guan, M. Zhu, and P. Liu, "Vcids: Collaborative intrusion detection of sensor and actuator attacks on connected vehicles," in *Security and Privacy in Communication Networks: 13th International Conference, SecureComm 2017, Niagara Falls, ON, Canada, October 22–25, 2017, Proceedings 13*, Springer, 2018, pp. 377–396.

[15] J. M. Ernst and A. J. Michaels, "Lin bus security analysis," in *IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society*, IEEE, 2018, pp. 2085–2090.

[16] B. S. Lim, S. L. Keoh, and V. L. Thing, "Autonomous vehicle ultrasonic sensor vulnerability and impact assessment," in *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, IEEE, 2018, pp. 231–236.

[17] F. Paez and H. Kaschel, "Towards a robust computer security layer for the lin bus," in *2021 IEEE International Conference on Automation/XXIV Congress of the Chilean Association of Automatic Control (ICA-ACCA)*, IEEE, 2021, pp. 1–8.

[18] J. Takahashi, Y. Aragane, T. Miyazawa, *et al.*, "Automotive attacks and countermeasures on lin-bus," *Journal of Information Processing*, vol. 25, pp. 220–228, 2017.

[19] W. Xu, C. Yan, W. Jia, X. Ji, and J. Liu, "Analyzing and enhancing the security of ultrasonic sensors for au-

tonomous vehicles," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 5015–5029, 2018.
[20] F. Páez and H. Kaschel, "Design and testing of a computer security layer for the lin bus," *Sensors*, vol. 22, no. 18, p. 6901, 2022.
[21] S. Behrad, E. Bertin, S. Tuffin, and N. Crespi, "A new scalable authentication and access control mechanism for 5g-based iot," *Future Generation Computer Systems*, vol. 108, pp. 46–61, 2020.
[22] M. Wazid, A. K. Das, V. Bhat, and A. V. Vasilakos, "Lam-ciot: Lightweight authentication mechanism in cloud-based iot environment," *Journal of Network and Computer Applications*, vol. 150, p. 102 496, 2020.
[23] W. Rafique, L. Qi, I. Yaqoob, M. Imran, R. U. Rasool, and W. Dou, "Complementing iot services through software defined networking and edge computing: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1761–1804, 2020.
[24] W. Yin, P. Hu, J. Wen, and H. Zhou, "Ack spoofing on mac-layer rate control: Attacks and defenses," *Computer Networks*, vol. 171, p. 107 133, 2020.
[25] S. Lee, W. Choi, and D. H. Lee, "Securing ultrasonic sensors against signal injection attacks based on a mathematical model," *IEEE Access*, vol. 7, pp. 107 716–107 729, 2019.
[26] J. Lou, Q. Yan, Q. Hui, and H. Zeng, "Soundfence: Securing ultrasonic sensors in vehicles using physical-layer defense," in *2021 18th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, IEEE, 2021, pp. 1–9.
[27] Y. L. Aung, S. Wang, W. Cheng, S. Chattopadhyay, J. Zhou, and A. Cheng, "Vnguard: Intrusion detection system for in-vehicle networks," in *International Conference on Information Security*, Springer, 2023, pp. 79–98.
[28] Y. An, J. Park, I. Oh, M. Kim, and K. Yim, "Design and implementation of a novel testbed for automotive security analysis," in *Innovative Mobile and Internet Services in Ubiquitous Computing: Proceedings of the 14th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS-2020)*, Springer, 2021, pp. 234–243.
[29] B. Stern, "Silicon photonic microresonators for multiplexing and coherent optical sources," 2018.
[30] Y. Koh, S. Kim, Y. Kim, I. Oh, and K. Yim, "Efficient can dataset collection method for accurate security threat analysis on vehicle internal network," in *International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, Springer, 2022, pp. 97–107.
[31] S. Kim, R. Shrestha, S. Kim, and R. Shrestha, "In-vehicle communication and cyber security," *Automotive Cyber Security: Introduction, Challenges, and Standardization*, pp. 67–96, 2020.
[32] A. Emadi, S. S. Williamson, and A. Khaligh, "Power electronics intensive solutions for advanced electric, hybrid electric, and fuel cell vehicular power systems," *IEEE Transactions on power electronics*, vol. 21, no. 3, pp. 567–577, 2006.
[33] T. Romppanen, "Developing a lin responder test tool," 2023.
[34] M. Kim, I. Oh, K. Yim, M. Sahlabadi, and Z. Shukur, "Security of 6g enabled vehicle-to-everything communication in emerging federated learning and blockchain technologies," *IEEE Access*, 2023.
[35] M. R. Islam, M. Sahlabadi, K. Kim, Y. Kim, and K. Yim, "Cf-aids: Comprehensive frequency-agnostic intrusion detection system on in-vehicle network," *IEEE Access*, 2023.

IX. BIOGRAPHY SECTION

INSU OH received Ph.D. degree from the Department of Information Security Engineering, Soonchunhyang University, Asan, South Korea, in 2023, respectively, where he is currently Post-doc Researcher. His research interests include vulnerability analysis, mobile baseband security, automotive security, and V2X security.

YOONJI KIM received her B.S. in 2023, from the Department of Information Security Engineering at Soonchunhyang University in Asan, South Korea. Currently, she is studying a Master's degree in Mobility Convergence Security at Soonchunhyang University in South Korea. Her research interests include external sensors vulnerability analysis and automotive security.

MD REZANUR ISLAM received B.Sc. in Electrical and Electronic Engineering from the University of Asia Pacific, Bangladesh, in 2016, and an M.Sc. in Mobility Convergence from Soonchunhyang University, South Korea, in 2023. Currently pursuing a Ph.D. in Software Convergence at Soonchunhyang, his research focuses on deep learning, anomaly detection, malware detection, computer vision, with a specialization in driver state recognition.

MAHDI SAHLABADI an IEEE Senior Member, holds a Ph.D. in Industrial Computing from the National University of Malaysia. His academic journey includes research positions at the Japan Advanced Institute of Science and Technology (JAIST), Singapore Management University (SMU), Sharif University of Tehran (SUT), University Kebangsaan Malaysia(UKM), and Soonchunhyang University (SCH), South Korea. His areas of research interest are process mining, software architecture, cybersecurity, and quality assurance.

KANGBIN YIM is a Professor in the Department of Information Security Engineering at Soonchunhyang University, where he has been since 2003. He received his B.S., M.S., and Ph.D. degrees in Electronics Engineering from Ajou University, South Korea, in 1992, 1994, and 2001, respectively. For over 20 years, his primary research has focused on vulnerability identification, threat analysis, and proof-of-concept (PoC) development for both software and hardware. He is also passionate about designing and implementing hardware and software frameworks for system evaluations and commercial services. His recent work has centered on HILS-based dynamic analysis for distributed embedded software, leading a research team of over 30 members in his lab, LISA. Currently, the lab's top priorities include deep-learning-driven analysis of heterogeneous field data, with a particular focus on automotive vehicles, industrial control systems, and mobile baseband.