Research of FlexRay Network Security based on Star Topology

Jin-Hui Piao, Yu-Jing Wu, Yi-Hu Xu, and Yi-Nan Xu

***Abstract*—FlexRay is a high fault-tolerant vehicle bus with real-time and flexibility. Most of the current research on FlexRay security issues focuses on bus topology. However, we found that once a hacker successfully invades one of the nodes in the bus topology system, he can monitor the communication content of other nodes or send false control information to any node. This vulnerability of the bus topology increases the risk of the system. Aiming at the characteristics of flexible topology structure of FlexRay, this paper proposes a network security scheme of vehicle FlexRay based on active star topology. This article combines key agreement, data encryption, data compression, and message authentication. Compared with other security protocols, it can reduce the verification steps and reduce the calculation time by 45.78% on the basis of ensuring the security of nodes.**

***Index Terms*—In-vehicle network, security, FlexRay, authentication.**

I. Introduction

With the development of electronic and communication technology, automotive electronic systems face more and more network security problems. At present, the research work related to vehicle communication network security mainly focuses on the CAN (Controller Area Network) protocol [1]. While the most widely used protocol for building in-vehicle networks remains CAN, many vehicle and safety-related critical functions are based on other communication protocols. For a long time, due to the lack of network security-related content in the communication protocol of the vehicle bus network, many network security risks have been caused. In July 2015, at the Black Hat Security Conference in Las Vegas, Valasek and Miller, released the results of their research on remote vehicle attacks. In addition, the protection systems of Toyota, Tesla, Audi, and other brands have also been breached. Attackers use the network security vulnerabilities of vehicles to launch various attacks, which not only cause property but also casualties. Therefore, more and more people focus their attention on the research field of in-vehicle network security.

The network security research content of vehicle bus is mainly based on encryption, message authentication and intrusion detection technology to improve vehicle security. In order to solve the authentication transmission problem of FlexRay message information, Murvay *et al.* proposes an authentication protocol that adapts to the time-triggered characteristics of FlexRay communication on the basis of the uncertain transmission characteristics existing in the dynamic segment [2]. However, to cope with the time and memory requirements, we still need to consider the generation and storage of a single key chain. Reference [3] applies Hash Message Authentication Code (HMAC) to specific messages, providing secure communication between ECUs to prevent network attacks and reduce the number of messages transmitted over the CAN bus. This method can provide message authentication to ensure that the message is not leaked. However, they do not consider key management during the driving of the vehicle. In [4], Wang et al. proposed a Vecure key distribution mechanism based on the trust degrees of different ECU groups. Vecure has efficient computation and key distribution performance, but it relies on fixed groups. When one ECU within a group is compromised, network security will be greatly affected. In [5], a semi-centralized dynamic key management framework for in-vehicle networks was proposed. It combines IBE (Identity-Based Encryption) and IBBE (Identity-Based Broadcast Encryption) to propose a semi-centralized key distribution for modern and future in-vehicle networks using identity-based cryptography. Decentralized session key generation isolates different subnets and solves the vehicle's single point of failure. Improves the security and reliability of automotive networks while providing temporary access to external devices and limiting the ability of non-critical ECUs. Reference [6] distributes keys and efficiently manages FlexRay communication data by using a reverse hash chain. The mechanism selects a dual-channel bus topology to split authentication labels on two physically independent channels, which not only improves security but also enables the system to have a higher degree of fault tolerance. However, the real-time problem of ECU two-way authentication in resource-constrained environments is not discussed.

In this paper, we propose a FlexRay network security scheme that integrates key negotiation, data encryption, data compression and message authentication. In Section II the FlexRay protocol is introduced. Section III presents our main contributions. We proceed with an evaluation and discussion of the proposed schemes in Section IV. Finally, we conclude this work in Section V.

II. The FlexRay Protocol

FlexRay provides a higher communication rate, more fault-tolerant and time-triggered communication protocol to meet

Manuscript received August 18, 2022; revised December 22, 2022. This work was supported by the National Natural Science Foundation of China (62161049 and 61763047).

The authors are with the College of Engineering of Yanbian University, Yanji, China (e-mail: ynxu@ybu.edu.cn).

doi: 10.18178/ijfcc.2022.11.4.594

the ever-increasing volume of in-vehicle communication data. The characteristics of FlexRay can be summarized into the following points: high reliability, topology flexibility, high communication rate and time-triggered communication. FlexRay supports dual-channel transmission, the transmission rate of each channel can reach up to 10Mbit/s, and the dual-channel can reach up to 20Mbit/s [7]. At the same time, the FlexRay bus supports a variety of topologies, including bus, star and hybrid topologies. Thus, flexibility is provided for the configuration of the network.

The communication of FlexRay is carried out in a cycle, and each communication cycle is divided into static segment, dynamic segment, symbol window and network idle time as shown in Fig. 1. The static segment and the dynamic segment occupy most of a communication cycle and are used to send application data. In a communication cycle, the static segment is a mandatory option, and others can be selectively added or deleted according to requirements. In addition, the length of each part is designed and fixed during network configuration, and cannot be changed after the communication is officially started. The static segment adopts a strict time division multiple access communication method to send time-triggered communication data. This part of the data is usually critical or safety-related data in the automotive system and needs to be updated frequently [9]. The static segment may be divided down into static slots of equal length. Likewise, the static slot length is also configured offline and fixed. Each static time slot is fixedly allocated to a data frame according to the network schedule, and the time slot number is correspondingly equal to the data frame ID. The dynamic segment uses FTDMA technology to achieve event triggering, and is usually used to send data that is not real-time or whose update frequency is extremely low compared to the communication cycle [8]. Therefore, the design of dynamic segments not only enhances the flexibility of network scheduling, but also enables more efficient use of bandwidth.

According to the analysis of main attack methods of FlexRay proposed at present, the network attack methods faced by FlexRay are mainly divided into the following three types. Repeatedly sending high-priority packets, the message that needs to be transmitted cannot be sent normally because the bus is occupied. To interfere with normal driving, the hacker sends the sent message again by monitoring the bus. Since this kind of attack is a purposeless attack, there is no major threat. The most dangerous is sending specific packets for targeted attacks. The attacker first obtains the packets corresponding to the operating system, and attacks the vehicle with specific functions at the characteristic moment. Connect the monitoring device to the FlexRay bus network through the physical layer to observe the previous communication of each node. While controlling the vehicle, find the corresponding relationship between the message and the control from the monitoring equipment. Then, use the OBD-II port, physical layer, etc. to access the ECU and the remote communication module to send data to the FlexRay bus to interfere with normal driving.

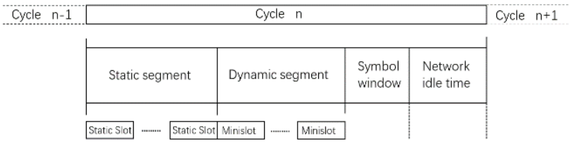


Fig. 1. Structure of a FlexRay communication cycle.

III. Securing in-Vehicle FlexRay Networks

A. Dual-Channel Cascaded Star Topology Network Structure Design

At present, the research on FlexRay bus focuses on the bus topology. However, when attacked by a node, FlexRay with a bus topology cannot directly determine the source of the attack and shield it accordingly. In order to eliminate the problem of different propagation delays caused by the bus topology, the active star topology is a good choice. The star source is no longer a connection point, but an electronic device that forwards data and, if necessary, cuts off a failed node. Therefore, it can also be understood as a topology with a central gateway. Under this structure, the robustness of the network is enhanced, and the propagation distance of the signal can also be extended.

As shown in Fig. 2, active stars can be cascaded in the system at speeds of 2.5Mbits/s and 5Mbit/s. This means that two active stars communicate with each other through a point-to-point connection. Its main advantages are as follows: 1) Assuming that hackers cannot control multiple star sources at the same time, the star source can immediately identify the source and cut off the malicious node after discovering the malicious node. 2) The star sourses of the same channel can check each other, which provides possible space for the application of fault-tolerant algorithms. 3) Each branch uses a separate interface to transmit data, and the physical characteristics of the channel are more obvious. In the active star topology, point-to-point communication is used between nodes. Since the channels are isolated from each other, it is easier to solve the problem of identifying the source of anomalies than the bus topology. Take corresponding remedial measures such as the star center can send reset or restart instructions to abnormal nodes.

The internal structure of the star source equipment is shown in Fig. 3. The chosen topology should stay within the decoder's asymmetric delay acceptance. The "Single-Branch" modules should contain a transmitter, a receiver, and a bus fault detector to act as the wake-up detector they use to detect wake-up things. The "Central_Logic" unit can connect the functions of other modules to make the whole system work successfully. But it is not specified separately in the protocol, so the "Central_Logic" unit requires additional security measures (design dedicated chips or modules to accelerate more complex security algorithms). Since the agreement stipulates a maximum of four-star sources, the cost will not increase exponentially. We can minimize the cost of additional hardware while maintaining security performance.

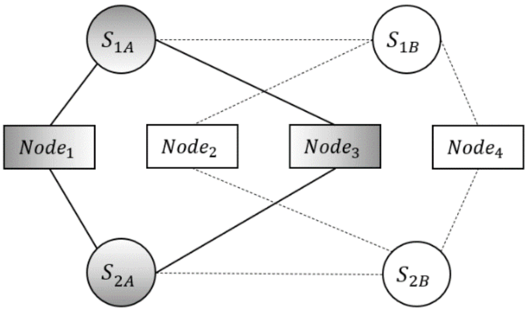


Fig. 2. Double cascaded star network topology.

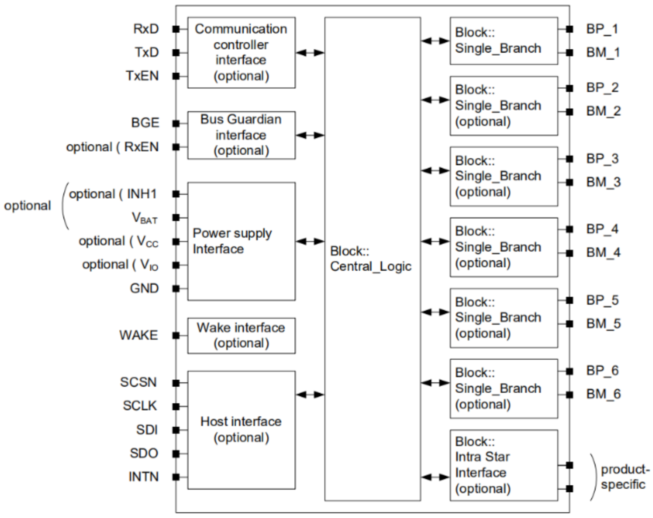

RxD
TxD
TxEN
Communication controller interface (optional)
BGE
optional (RxEN
Bus Guardian interface (optional)
optional
optional (INH1
V_{BAT}
optional (V_{CC}
optional (V_{IO}
GND
Power supply Interface
WAKE
Wake interface (optional)
SCSN
SCLK
SDI
SDO
INTN
Host interface (optional)
Block:: Central_Logic
Block:: Single_Branch
Block:: Single_Branch (optional)
Block:: Single_Branch (optional)
Block:: Single_Branch (optional)
Block:: Single_Branch (optional)
Block:: Single_Branch (optional)
Block:: Intra Star Interface (optional)
BP_1
BM_1
BP_2
BM_2
BP_3
BM_3
BP_4
BM_4
BP_5
BM_5
BP_6
BM_6
product-specific

Fig. 3. Star source structure equipment diagram.

Based on the vehicle-by-wire system model and the characteristics of the FlexRay protocol, this paper gives the following settings when designing the security architecture based on the dual-channel cascaded star topology:

1) There are four star sources in the network that are responsible for authenticating other nodes and issuing communication keys.
2) The key negotiation is performed in the remaining mini-slots after the data is sent in the dynamic segment.
3) When the car is started, star sources and each node generate the public and private keys of the ECDH algorithm, and update the generated random numbers.
4) Each communication time slot of a node corresponds to a local communication counter, which is used to manage and synchronize the number of data sent and received by the node in this time slot.
5) Star sources has stronger computing power and storage capacity than other affiliated nodes.

B. Key Agreement Design Based on ECDH-SHA3 Algorithm

If a fixed key is used for communication within the vehicle, an attacker who obtains the key for the corresponding vehicle can easily compromise the car. Therefore, we use ECDH algorithm and SHA-3 algorithm for key initialization every time the vehicle starts. Furthermore, the real-time nature of computing resources and algorithms in automotive embedded microprocessors has always been a sensitive topic. Even the extra computation time for network security exceeding 2ms will affect the real-time performance of the system. However, additional computation time for initial session key distribution within 300ms is acceptable for security during the car ignition start-up phase.

The initial key distribution module is activated every time the car ignition starts. ECDH algorithm is used by star sources to negotiate the initial key with connected star sources and nodes during the vehicle ignition start-up phase. SHA-3 algorithm is used to verify the authenticity of the key agreement data, which can effectively improve the problem that the original ECDH algorithm is insufficient in fighting against man-in-the-middle attacks [9], [10]. As shown in Fig. 4, Node_1 and Node_3 communicate via two uncascaded star sources $S_{1A}\mathrm{S}_{1\mathrm{A}}$ and S_{2A}. The specific steps of the initial key agreement process are as follows:

1) S_{1A}, S_{2A}. and $Node_1$ generate random numbers R_{1A}^1, R_{2A}^1, R_1^{1A} and $\mathrm{R}_1^{2\mathrm{A}}$ with a random number generator. $\mathrm{R}_{1\mathrm{A}}^1$ and R_{2A}^1 respectively represent the private keys corresponding to S_{1A} , S_{2A} .and $Node_1\mathrm{Node}_1$ to calculate the shared key. Similarly, $\mathrm{R}_1^{1\mathrm{A}}$ and $\mathrm{R}_1^{2\mathrm{A}}$ respectively represent the private key corresponding to the shared key calculated by Node_1, S_{1A} and S_{2A}.
2) Star sources and nodes first choose their ellipse common values: Ep (a, b) defines the elliptic curve in the Fp domain and the base point G of the shared curve parameters that have been calculated and stored. Then, Use Equation (1), take S_{1A} and $Node_1$ as example : $\mathrm{Pub}_{1\mathrm{A}}^1 = \mathrm{R}_{1\mathrm{A}}^1 \times \mathrm{G}_1^{1\mathrm{A}}\mathrm{Pub}_{1\mathrm{A}}^1 = \mathrm{R}_{1\mathrm{A}}^1 \times \mathrm{G}_1^{1\mathrm{A}}$ indicates the public key that S_{1A} wants to send to $Node_1$ through the channel; $Pub_1^{1A} = R_1^{1A} \times G_1^{1A}$ $\mathrm{Pub}_1^{1\mathrm{A}} = \mathrm{R}_1^{1\mathrm{A}} \times \mathrm{G}_1^{1\mathrm{A}}$ indicates the public key that $Node_1$wants to send to $\mathrm{S}_{1\mathrm{A}}$.

$$Pub = R \times G \tag{1}$$

3) As shown in equation (2), S_{1A} and S_{2A}. use HMAC-SHA3 to calculate MAC and append to $\mathrm{Pub}_{1\mathrm{A}}^1$ and $\mathrm{Pub}_{2\mathrm{A}}^1$ can get key negociation data $\mathrm{Ika}_{1\mathrm{A}}^1$ and $\mathrm{Ika}_{2\mathrm{A}}^1$ (send to Node_1). Similarly, the key negotiation data $\mathrm{Ika}_1^{1\mathrm{A}}$ and $\mathrm{Ika}_1^{2\mathrm{A}}$ sent by Node_1 to S_{1A} and $\mathrm{S}_{2\mathrm{A}}$ S_{2A}. can be obtained.

$$Ika = Pub||HMAC(Pub) \tag{2}$$

4) The receiver first uses the HMAC-SHA3 algorithm to verify the legitimacy of the key agreement data to prevent man-in-the-middle attacks. If the verification is passed, go to the next step to calculate the initial session key.
5) The receiver calculates the initial session key Isk by equation (3).

$$Isk = Pub \times Ika \tag{3}$$

Take the initial session key between $Node_1$ and S_{1A}, $Isk_1^{1A} = (Pub_{1A}^1 \times R_1^{1A}) = (Pub_1^{1A} \times R_{1A}^1) = Isk_{1A}^1$, initial session key negotiation succeeded. Through the above-mentioned ECDH-SHA3 key agreement process, the dual transmission and reception can ensure that the initial session key can be negotiated securely when the attacker obtains the key agreement data. In addition, the introduction of HMAC-SHA3 message authentication algorithm can also prevent man-in-the-middle attacks and further ensure the authenticity of the initial session key.

In the above initial session key distribution, we perform a key initialization during the car startup phase. During the real-time operation of the car, the key agreement algorithm is used to continuously update the key, which can not only improve the security of the message, but also improve the ability of the system to resist replay attacks.

Considering that during the communication process, the additional transmission of negotiated key data will increase the load of the bus and reduce the real-time performance of communication. Too frequent and rapid update of session keys will also increase the burden of computing resources, which is not conducive to the system's decision-making response speed and the ability to expand to new applications. To sum up, we propose a real-time key agreement process based on the remaining time slots of the dynamic segment, that is, the data used for key negotiation is transmitted in the remaining time slots of the dynamic segment. In the dynamic segment, if there are two or more data frames requesting to send at the same time, the arbitration will be based on the size of the dynamic ID (Dynamic ID) of the data frame who will send it first. Therefore, using this feature we can set the dynamic ID used for the key agreement message (mk) to be larger than the original message used for control (m).

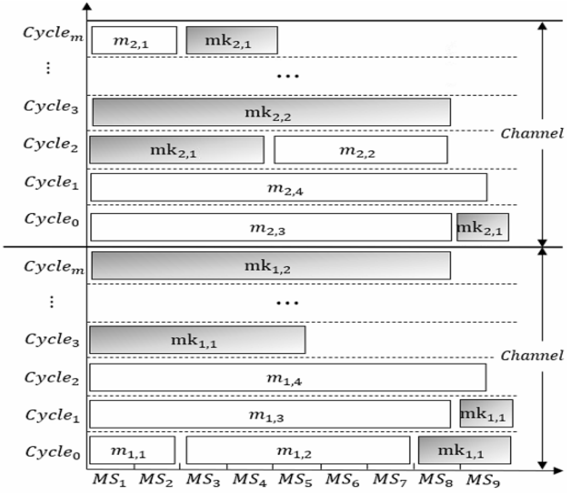


Fig. 4. Dynamic segment key agreement arrangement.

The specific arrangement of dynamic segment key negotiation is shown in Fig. 4. The benefits of this design are as follows: The bandwidth utilization of FlexRay can be maximized by sending key agreement data using passband segments that are otherwise not fully utilized. Normal data is always sent prior to key agreement data, which can minimize the impact of additional key negotiation data on the real-time communication of dynamic message frames. The session key is always calculated after the session key agreement data is received, so as not to consume a lot of computing resources by negotiating the key too frequently. Besides, the calculation process of the session key agreement in the communication process is the same as the calculation process of the initial session key agreement.

C. Data Encryption and Authentication

Encryption of data is crucial to address the security vulnerabilities faced by vehicle nodes in normal driving. In-vehicle bus communication requires security and real-time. In order to further ensure the real-time nature of the communication data, the symmetric encryption algorithm AES-128 is selected to encrypt the data transmitted in the FlexRay network environment. After the vehicle is started, the legal nodes in the FlexRay network have passed the identity authentication and obtained the session key required in the communication phase. By applying the idea of the ICANDR algorithm in the Reference [11], the data transmitted in the FlexRay bus environment is compressed to improve the bus utilization. The basic idea of this data compression method is to send only the variation of the data, not the complete value of the data.

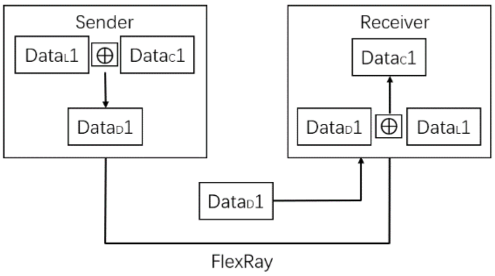


Fig. 5. Transceiver node data compression.

At the sending and receiving ends of the communication, for each signal, a buffer is maintained, which stores the complete signal value of the signal in the previous cycle. When a certain communication cycle is at the sending end and the data of a certain signal is ready to be sent, the complete value of the current cycle of the signal is XORed with the complete value of the previous cycle, and the result is used as a variation, that is, compressed data, and sent to Receiving end. At the receiving end, since it also holds the complete signal value of the previous cycle of the signal, the original signal value can be obtained by XORing it with the received data. The specific operation is shown in Fig. 5, where DataL, DataC and DataD respectively represent the complete value of the previous cycle of the signal, the complete value of the current cycle and the change amount of the signal value. Finally, after the node and session keys are authenticated, the compressed data is encrypted using the AES algorithm. The specific encryption process is as follows:

Step 1 The sending node $Node_i$ uses the ICANDR algorithm to compress the data to obtain M_C.

Step 2 $Node_i$ use AES algorithm to encrypt the compressed data M_C to get ciphertext C. In equation (4), $\mathrm{CTR}_{\mathrm{i,j}}$ is the local counter of $Node_i$ at slot j.

$$C = F(CTR_{i,j}, Isk) \oplus M_C \mathrm{C} = \mathrm{F}(\mathrm{CTR}_{\mathrm{i,j}}, \mathrm{Isk}) \oplus \mathrm{M}_{\mathrm{C}} \tag{4}$$

Step 3 MAC operation is performed on C, R, IDi, and Isk of the sending node to generate a message verification code Ms.

$$\mathrm{M}_{\mathrm{s}} = \mathrm{MAC}(\mathrm{C}, \mathrm{Ri}, \mathrm{IDi}, \mathrm{Isk}) \tag{5}$$

Step 4 The sending node $Node_i$ appends the message verification code Ms to the ciphertext C, sends it to the receiving node $\mathrm{Node}_{\mathrm{j}}$, and calculates Ms'.

$$\mathrm{M}_{\mathrm{s}}' = \mathrm{MAC}(\mathrm{C}, \mathrm{Rj}, \mathrm{IDi}, \mathrm{lsk}) \tag{6}$$

Step 5 Determine the following equation:

$$\mathrm{M}_{\mathrm{s}}^{'} = \mathrm{M}_{s} \tag{7}$$

If equation (7) holds, the communication is successful, otherwise the message may be leaked, terminating the communication.

IV. Performance Analysis

In this paper, Diffie Hellman key exchange algorithm, RSA algorithm and Elliptic Curve Diffie Hellman (EC-DH) algorithm are implemented in hardware. Table I summarizes the time delay comparison of the following key negociation algorithms in 100 communication cycles. The ECDH algorithm has the advantages of low power consumption, light weight and robustness. After comparing the performance parameters, this paper considers using the ECDH algorithm in the key agreement part.

TABLE I: Time Delay Comparison of Various Algorithms

Delay Time(ms)	Diffi-Hellman	RSA	ECDH
Software(runtime)	6.7	8.0	5.9
Critical path delay	48.9	89.9	23.2
Latency	49.8	91.0	21.8

To further verify the feasibility and effectiveness of the in-vehicle FlexRay bus security scheme proposed in this paper, we use a bus development environment CANoe simulation verification from Vector Company in Germany. Since algorithms such as data encryption and message authentication consume a lot of computational resources, and data is sent every cycle to execute security algorithms. This will occupy the processor too much and cause the normal tasks of the ECU to fail to run normally [12]. The occupied time increases linearly with the increase of the amount of data to be processed. Therefore, the amount of data to be processed needs to be reduced by means of compression. The number of messages with IDs 4 and 63 used in this study is 96,000 data (96,000 communication cycles) and 12,000, respectively. The length of the message with ID 4 before compression is 20 bytes, and the message with ID 63 is 6 bytes. When using the AES-128 algorithm, the required encryption times are two and one respectively. Fig. 6 shows the data sending of formal communication on the analog bus in the CANoe environment. The data comparison of ID4 in the ninth cycle and the eighth cycle is shown in Fig. 7: when the content of the corresponding byte changes, that is, the XOR value is not 0, the indicator bit is set to "1". "HEX" is the hexadecimal representation of the indicator bits. Due to the FlexRay protocol, the amount of data actually sent will not be reduced, and most of the data sent is "0". Although this part of data is sent, it does not carry any information. After using the compression method, the length of the compressed data of ID4 is all less than 10 bytes, and there are 10999 data lengths of 0 bytes. Most of the data in ID48 are 0 bytes.

For a data frame, the total processor time occupied by each safe operation on it can be calculated by Equation (8):

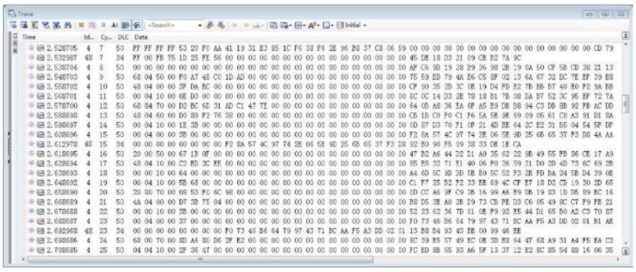
$$D_{IDi} = D_{En} \times T_{En} + D_{ECDH} + D_{MAC} \tag{8}$$

D_{IDi} is the total time occupied by the processor for the IDi security operation, and D_{En} is the time required for encryption. T_{En} is the number of times required for encryption, D_{ECDH} is the time required for key negotiation, and D_{MAC} is the time required to calculate the message authentication code. According to the hardware actual measurement time of each security algorithm, the total time required by the processor for security operations of ID4 and ID48 on the sender and receiver ends is shown in Table II.

TABLE II: Safe Operation Consumes Processor Time

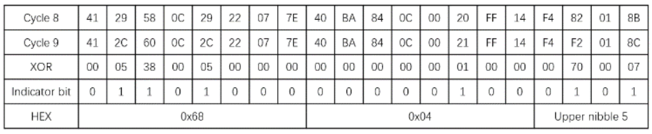
Sender(ms)		Receiver(ms)	
ID4	ID48	ID4	ID48
4.904	4.572	5.936	5.604

In order not to lose generality, this paper continues to perform compression tests on 10 data frames sent by different ECUs. The length of the data frame in the experiment is 8 bytes. Since the number of samples of each data frame is different, the number of samples of different lengths after compression of each data frame is counted, and the value of this number is used as a quotient with the total number of samples of the data frame, and the result is shown in the form of a percentage as shown in Fig. 8 shown. In terms of data encryption after compression, the average occupied time and sum decreased by 45.78%, and the occupied time of all security operations at the sending and receiving ends decreased by 5.35% and 4.30% respectively.



```
Time          Id..  Cy..  DLC  Data
2.528705      4     7     50   FF FF FF FF 53 20 F0 AA 41 19 31 E3 85 1C F6 33 F6 2E 96 D8 37 C8 06 59 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 CD 79
2.532987      48    7     34   FF 00 FB 75 1D 25 FE 56 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 45 DE 18 D3 21 09 CB B2 7A 9C
2.538704      4     8     50   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 AF C6 9D 19 28 B9 36 98 2B 19 0A 50 CF 5B CD 38 21 13
2.548703      4     9     50   68 04 50 00 F0 A7 48 C0 1D AD 00 00 00 00 00 00 00 00 00 00 00 00 00 00 75 59 ED 79 4A E6 C5 8F 02 13 6A 67 32 DC 7E EF 39 B8
2.558702      4     10    50   48 04 00 00 3F DA BC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 CF 90 35 2D 3C 0B 19 D4 FD E2 7B BB B7 40 B0 F2 9A BB
2.568701      4     11    50   00 04 10 00 0E D3 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 8C 0C 14 D3 2B 78 18 B1 7B 08 DA B7 52 3C 95 EF 72 7A
2.578700      4     12    50   68 84 70 00 D2 BC 6D 31 AD C1 47 7E 00 00 00 00 00 00 00 00 00 00 00 00 64 0B A8 36 EA 6F A5 E9 DB D8 94 C3 DB 8B 92 FB AC DD
2.588698      4     13    50   48 04 60 00 D0 89 F2 76 38 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 CB 1B C0 F0 C1 F6 5A 5E 9E 69 09 05 61 C8 A3 91 D1 8A
2.598697      4     14    50   00 04 10 00 1E 3B 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 CD 87 D3 70 F1 0F 21 4D EE 64 2C E2 31 B5 04 54 5F DF
2.608696      4     15    50   00 04 00 00 3B 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 F2 8A 57 4C 97 74 2B 06 5E 9D 25 6B 65 37 F3 D8 4A AA
2.612978      48    15    34   00 00 00 00 00 00 00 00 F2 8A 57 4C 97 74 2B 06 5E 9D 25 6B 65 37 F3 D8 92 B0 90 F5 09 38 33 DE 1E CA
2.618695      4     16    50   20 00 50 00 67 1B 0F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 47 B2 A6 44 D2 D1 A0 35 62 22 9B 49 55 FB 86 CE 17 A9
2.628694      4     17    50   48 04 10 00 C2 BD 2C EE 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 85 E5 32 71 E1 40 06 F0 26 59 21 D0 2D 4D 73 6C 69 2B
2.638693      4     18    50   00 00 10 00 64 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 A4 6D 5C 9D DD 5B E0 5C 52 F3 2B FD BA 24 5B D4 39 0E
2.648692      4     19    50   00 04 10 00 5B 68 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 C1 F7 25 B2 F2 33 EE 69 4C CF E7 18 D2 CD 19 30 2D 65
2.658690      4     20    50   20 80 70 00 08 53 F0 6C 98 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 CD CC A6 9F C9 2B 16 99 A6 E9 DB 19 83 1D D5 89 EC 16
2.668689      4     21    50   4A 04 00 00 D7 3B 75 04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 B8 D5 3E A0 2B D9 73 CB FE D3 C6 05 49 8C C7 F9 FE 21
2.678688      4     22    50   00 00 10 00 3B 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 52 23 62 36 7D 01 0E F9 92 E5 44 D1 65 B0 A2 C3 70 87
2.688687      4     23    50   00 04 00 00 37 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 F0 73 46 B6 64 79 97 43 71 BC AA F5 A3 DD 02 01 B1 AE
2.692968      48    23    34   00 00 00 00 00 00 00 00 F0 73 46 B6 64 79 97 43 71 BC AA F5 A3 DD 02 01 13 B8 B4 93 43 EB 00 99 46 EE
2.698686      4     24    50   68 00 70 00 8D A6 80 D6 2F E2 00 00 00 00 00 00 00 00 00 00 00 00 00 00 8C 39 E5 57 49 EC 0E 3D E8 64 47 68 A9 31 A4 F5 EA C2
2.708685      4     25    50   04 04 10 00 2F 36 47 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FC ED 8B 65 93 A6 5F 13 37 12 E2 8C 85 54 88 16 06 35
```

Fig. 6. Bus data transmission after applying the security protocol.



Cycle 8	41	29	58	0C	29	22	07	7E	40	BA	84	0C	00	20	FF	14	F4	82	01	8B
Cycle 9	41	2C	60	0C	2C	22	07	7E	40	BA	84	0C	00	21	FF	14	F4	F2	01	8C
XOR	00	05	38	00	05	00	00	00	00	00	00	00	00	01	00	00	00	70	00	07
Indicator bit	0	1	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1
HEX	0x68								0x04								Upper nibble 5			

Fig. 7. Data Compression Comparison of ID4.

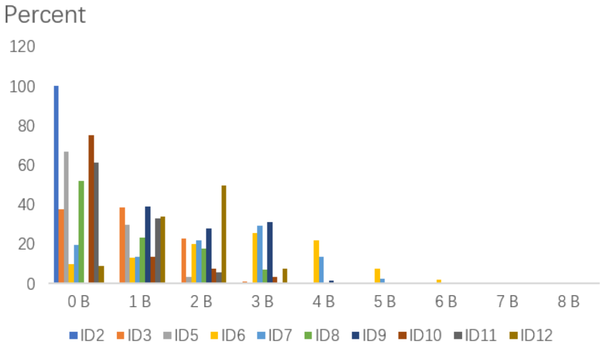


Fig. 8. Distribution of Multiple ECU Data Frames after Compression.

For the convenience of observation, we choose to intercept the first 4 bytes of the calculated message verification code and fill it with valid data communicated in the FlexRay bus network as a valid message verification code. If the attacker wants to tamper with the data to carry out a forgery attack, he can only select the message authentication code by brute force cracking (that is, select one of the 232 message authentication codes). However, the communication cycle of the FlexRay bus is limited, and it takes too long to test the selected message authentication codes one by one, and the attack is very difficult.

It is assumed that the attacker can monitor the information through the external device. In this paper, the message authentication code is obtained by calculating the message and the communication counter together through the HMAC-SHA1 algorithm. Since the sending and receiving nodes share a communication counter, the message verification codes calculated by different sending times or data are different. As shown in Table III, for the convenience of observation, the first 8 bytes of some messages in the experiment and the first 4 bytes of their message authentication codes are given. Among them, the data with sequence numbers 1 to 6 are legal frames sent by E0 and E1 nodes. The 7th data show the message packets sent by the attacking node E2 in the replay attack. The receiving node can successfully detect the replay attack when it receives the replay frame to ensure the authenticity of the data. The 8th and 9th data are forged data frames when the attacking node E2 tampered with the data to conduct a forgery attack. From the information in the table, we can see that we can successfully detect the forgery attack to ensure the authenticity of the data.

TABLE III: THE DETECTION RESULTS OF THE BUS

	Sender	Time Slot	Data (8 Bytes)	MAC (4 Bytes)	Legal
1	E_0	7	67 22 … 00 08 03	c5 bf 3e 67 c5	Yes
2	E_0	9	67 4c … 11 10 00	c5 2c 74 c7 cf	Yes
3	E_0	1	0x 87 … 09 00 0c	c5 1b 24 9c 3a	Yes
4	E_1	5	0x 11 … 1d 00 09	c5 b7 f0 47 99	Yes
5	E_1	6	0x 44 … 0c 0d 0e	c5 2e 56 25 13	Yes
6	E_1	44	67 09 … 03 00 07	c5 a9 4f 65 c7	Yes
7	E_2	41	0x 11 … 1d 00 09	c5 b7 f0 45 08	No
8	E_2	3	0x 22 … 12 00 11	c5 37 4e 4e 54	No
9	E_2	4	67 08 … 20 08 04	c5 22 a4 45 18	No

V. CONCLUSION

Aiming at the network security of FlexRay bus, this paper innovatively proposes a security solution for vehicle FlexRay bus based on active star topology. Compared with the traditional bus topology, the active star topology used in this paper can detect anomalies in time and find the source. Aiming at the security defects of the FlexRay bus network, a secure communication protocol that integrates data encryption, message authentication and key agreement strategies is proposed to resist message retransmission and forgery attacks by external malicious nodes on the bus network. To solve the problem that the operation time of the security algorithm in the main processor increases linearly with the increase of the amount of data, the idea of compressing data using ICANDR is used to process the data frame of FlexRay. This reduces the amount of data that the safety algorithms need to process and leaves enough time for the ECU main processor to complete other functional tasks. Finally, the proposed security scheme is proved to be feasible through vehicle ECU and CANoe bus simulation software. On the basis of ensuring security, this scheme reduces the average occupied time of the encryption algorithm to 45.78%. And reduce the average occupied time of all security operations at both ends by 5.35% and 4.30% respectively. Finally, this paper takes advantage of the flexible structure of the star topology, which improves the security of the bus.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Jin-Hui Piao and Yu-Jing Wu: Methodology and writing of original draft; Yi-Hu Xu: Software and formal analysis; Yi-Nan Xu: Conceptualization and supervision.

ACKNOWLEDGMENT

This research was supported by National Natural Science Foundation of China (62161049 and 61763047).

REFERENCES

[1] B. Groza and P. Murvay, "Security solutions for the controller area network: Bringing authentication to in-vehicle networks," *IEEE Vehicular Technology Magazine*, vol. 13, no. 1, pp. 40-47, 2018.

[2] P. S. Murvay, L. Popa, and B. Groza, "Accommodating time-triggered authentication to FlexRay demands," in *Proc. Third Central European Cybersecurity Conference (CECC)*, New York, USA, 2019.

[3] H. Mun, K. Han, and D. H. Lee, "Ensuring safety and security in CAN-based automotive embedded systems: A combination of design optimiza tion and secure communication," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7078–7091, Jul. 2020.

[4] Q. Wang and S. Sawhney, "VeCure: A practical security framework to protect the CAN bus of vehicles," in *Proc. IEEE Int. Conf. Internet Things*, 2014, pp. 13–18.

[5] E. Carvajal-Roca, J. Wang, J. Du, and S. Wei, "A semi-centralized dynamic key management framework for in-vehicle networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 10, pp. 10864-10879, Oct. 2021.

[6] D. Püllen, N. A. Anagnostopoulos, T. Arul *et al.*, "Securing FlexRay-based in-vehicle networks," *Microprocessors and Microsystems*, 2020, vol. 77.

[7] F. Sagstetter, M. Lukasiewycz, and S. Chakraborty, "Generalized asynchronous time-triggered scheduling for FlexRay," in *Proc. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 2, pp. 214-226, Feb. 2017.

[8] K. Klobedanz, A. Koenig, and W. Mueller, "A reconfiguration approach for fault-tolerant flexray networks," *Design, Automation and Test in Europe (DATE)*, 2011

[9] C. Guo and B. Gong, "Efficient scalar multiplication of ECC using SMBR and fast septuple formula for IoT," *EURASIP Journal on Wireless Communications and Networking*, 2021.

[10] M. F. Moghadam, M. Nikooghadam, M. A. B. A. Jabban, M. Alishahi, L. Mortazavi, and A. Mohajerzadeh, "An efficient authentication and key agreement scheme based on ECDH for wireless sensor network," *IEEE Access*, vol. 8, pp. 73182-73192, 2020.

[11] Y. J. Wu and J. G. Chung, "An improved controller area network data reduction algorithm for in-vehicle networks," *IEICE Trans. Fundamentals*, vol. E100-A, no. 2, pp. 346-352, Feb. 2017.

[12] S. Y. Jin, M. Z. Liu, Y. J. Wu, Y. H. Xu, J. N. Jiang, and Y. N. Xu, "Research of message scheduling for in-vehicle FlexRay network static segment based on next fit decreasing (NFD) algorithm," *Applied Science*, vol. 8, no. 2071, pp. 1-13, 2018

Jin-hui Piao received her M.S. in electronics engineering from Yanbian University, China, in 2022.

She is currently working toward a doctor degree in the area of In-vehicle Network and automobile control in Chonbuk National University, South Korea.

Yu-Jing Wu was born at Jilin province of China.

She received her M.S. and Ph.D in electronic and information engineering from Chonbuk National University, South Korea, in 2013 and 2016, respectively.

She is a lecturer of the division of electronic and communication engineering of Yanbian University, China. Her research interests include the In-vehicle communication networks.

Yi-Hu Xu was born at Jilin province of China. He received the Ph.D. degree in electronics engineering from the Chonbuk National University, South Korea, in 2014.

He is an associate professor of the division of electronic and communication engineering of Yanbian University, Yanji, China. His research interests include the automobile electronic control and network.

Yi-Nan Xu was born at Jilin province of China. He received the Ph.D. degree in electronics engineering from the Chonbuk National University, South Korea, in 2009.

He is a professor of the division of electronics and communication engineering of Yanbian University, Yanji, China. His research interests include the In-vehicle network and automobile electronic control.