symmetry

Article

SGXAP: SGX-Based Authentication Protocol in IoV-Enabled Fog Computing

Tsu-Yang Wu [1], Xinglan Guo [1], Yeh-Cheng Chen [2], Saru Kumari [3] and Chien-Ming Chen [1,*]

[1] College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China; wutsuyang@sdust.edu.cn (T.-Y.W.); xinglan2021@sdust.edu.cn (X.G.)
[2] Department of Computer Science, University of California, Davis, CA 001313, USA; ycch@ucdavis.edu
[3] Department of Mathematics, Chaudhary Charan Singh University, Meerut, Uttar 250004, Uttar Pradesh, India; saru@ccsuniversity.ac.in
* Correspondence: chienmingchen@sdust.edu.cn

Abstract: With the maturity and popularization of the Internet of Things, we saw the emergence of the Internet of Vehicles. This collects and processes real-time traffic information, alleviates traffic congestion, and realizes intelligent transportation. However, sensitive information, such as real-time driving data of vehicles, are transmitted on public channels, which are easily to steal and manipulate for attackers. In addition, vehicle communications are vulnerable to malicious attacks. Therefore, it is essential to design secure and efficient protocols. Many studies have adopted asymmetric cryptosystems and fog computing to in this environment, but most of them do not reflect the advantages of fog nodes, which share the computational burden of cloud servers. Therefore, it is challenging to design a protocol that effectively uses fog nodes. In this paper, we design an authentication protocol based on a symmetric encryption algorithm and fog computing in the Internet of Vehicles. In this protocol, we first propose a four-layer architecture that significantly reduces the computational burden of cloud servers. To resist several well-known attacks, we also apply Intel software guard extensions to our protocol. This is because it can resist privileged insider attacks. We prove the security of the proposed protocol through the Real-Or-Random model and informal analysis. We also compare the performance of the proposed protocol with recent protocols. The results show better security and a lower computational cost.

Keywords: authentication; Internet of Vehicles; fog computing; SGX; symmetric encryption

check for updates

Citation: Wu, T.-Y.; Guo, X.; Chen, Y.-C.; Kumari, S.; Chen, C.-M. SGXAP: SGX-Based Authentication Protocol in IoV-Enabled Fog Computing. *Symmetry* **2022**, *14*, 1393. https://doi.org/10.3390/sym14071393

Academic Editor: Giuseppe Bagliesi

Received: 31 May 2022
Accepted: 1 July 2022
Published: 6 July 2022

1. Introduction

With the maturity and popularization of the Internet of Things (IoT) [1,2], a special network connecting vehicles through the Internet has emerged: the Internet of Vehicles (IoV) [3–5]. The IoV is a subset of the IoT that realizes communication by vehicle-to-vehicle (V2V), vehicle-to-pedestrian (V2P), and vehicle-to-infrastructure (V2I) connections. The IoV collects, processes, and shares road information in real time, alleviates traffic congestion in traffic control, and reduces traffic accidents through early warnings to ensure vehicle safety. It also realizes intelligent transportation to improve its efficiency.

Previously, researchers introduced cloud computing into the IoV to efficiently process large amounts of real-time road information. With an increasing number of vehicles, the computational burden of the cloud server (CS) is also increasing. The authors of [6–8] proposed a definition of fog computing. Compared with cloud computing, fog computing has the characteristics of low latency, large numbers, wide distribution, and lighter computing. Fog and cloud computing are complementary but not substitutes. Cloud computing realizes the calculation or storage of a large amount of data, compensating for the lack of computing resources for fog nodes.

Recently, the literature [9–12] has involved research on authenticated key agreement (AKA) protocols for applying fog computing to the IoV. In 2019, Ma et al. [9], based on asymmetric cryptosystems and fog computing, proposed an AKA protocol. The protocol used the traditional three-layer architecture: “vehicle–fog node–CS”, where the vehicle and roadside unit (RSU) play the participant (i.e., vehicle). However, the protocol showed that the participation of a CS was required for each authentication, which did not reflect the advantages of fog nodes and did not realize the function of fog nodes sharing the computational burden of a CS. Moreover, Eftekhari et al. [10] found that the protocol [9] was insecure and vulnerable to stolen smart card attacks, known session-specific temporary information disclosure attacks, and privileged insider attacks. Based on this architecture, Eftekhari et al. [10] designed an improved protocol that did not reflect the advantages of fog nodes. In 2020, Wu et al. [11] proposed a fog-based AKA protocol based on the traditional three-layer architecture. However, the RSU was a fog node. Each communication required the participation of a CS. In 2021, following the architecture presented in [11], Wu et al. [12] proposed a lightweight AKA protocol that still did not realize the function of fog nodes sharing the computational burden of a CS.

Although the above AKA protocols [9–12] use fog computing, they fail to realize the function of fog nodes sharing the computational burden of a CS. This is because in the conventional architecture, fog nodes actually replace the RSU, and computing is still on the CS, which does not reduce the computational burden of the CS. Therefore, we first propose a four-layer architecture: “vehicle–RSU–fog node–CS”. The four-layer architecture of the IoV based on fog computing is shown in Figure 1. In this architecture, when a vehicle enters the road and wants to communicate with the RSU, the communication modes are divided into the following two cases:

1. **Case 1:** The RSU judges that the vehicle communicates with itself for the first time and then sends a data request to the CS. The CS sends a response to the RSU accordingly and simultaneously sends the data response to the fog node. Thereafter, the vehicle and RSU realize their communication with the assistance of the fog node. The four entities involved in this communication process are the vehicle, RSU, fog node, and CS.
2. **Case 2:** This extends from Case 1. Only the fog node (without the participation of the CS) can help the vehicle and RSU to realize communication. Here, this architecture effectively realizes the function of the fog node sharing the computational burden of the CS.

The IoV environment still has some security challenges. For example, sensitive information such as vehicle real-time driving data are transmitted on a public channel, which is easy to steal from and manipulate for an attacker, resulting in the disclosure of vehicle privacy. In addition, the process of vehicle communication is vulnerable to replay attacks [13], impersonation attacks [14–16], and privileged insider attacks [9], among others. Therefore, to ensure communication and protect the sensitive data of vehicles, a safe and effective protocol must be designed.

Due to the above security challenges of the IoV environment, researchers are committed to enhancing the security of the IoV. Therefore, a hardware-based, trusted execution environment called software guard extensions (SGX) [17–19] has emerged. SGX is secure hardware developed by Intel. The difference between SGX and other security software is that it only includes hardware, which avoids software vulnerabilities and malicious threats in the system and largely ensures system security. In addition, SGX provides a trusted execution environment, as malicious code cannot access or tamper with any sensitive data stored in SGX, guaranteeing data confidentiality and integrity. The structure of SGX mentioned in [17]. Preserved random memory (PRM) is a reserved area for SGX in the dynamic memory, and the Enclave Page Cache (EPC) is a part of the PRM. SGX has a secure container called Enclave, which is stored in the EPC to store sensitive data and code. The user enters the value into the Enclave through the Ecall. After SGX completes the confidential computing in the Enclave, it returns the computational results through Ocall.

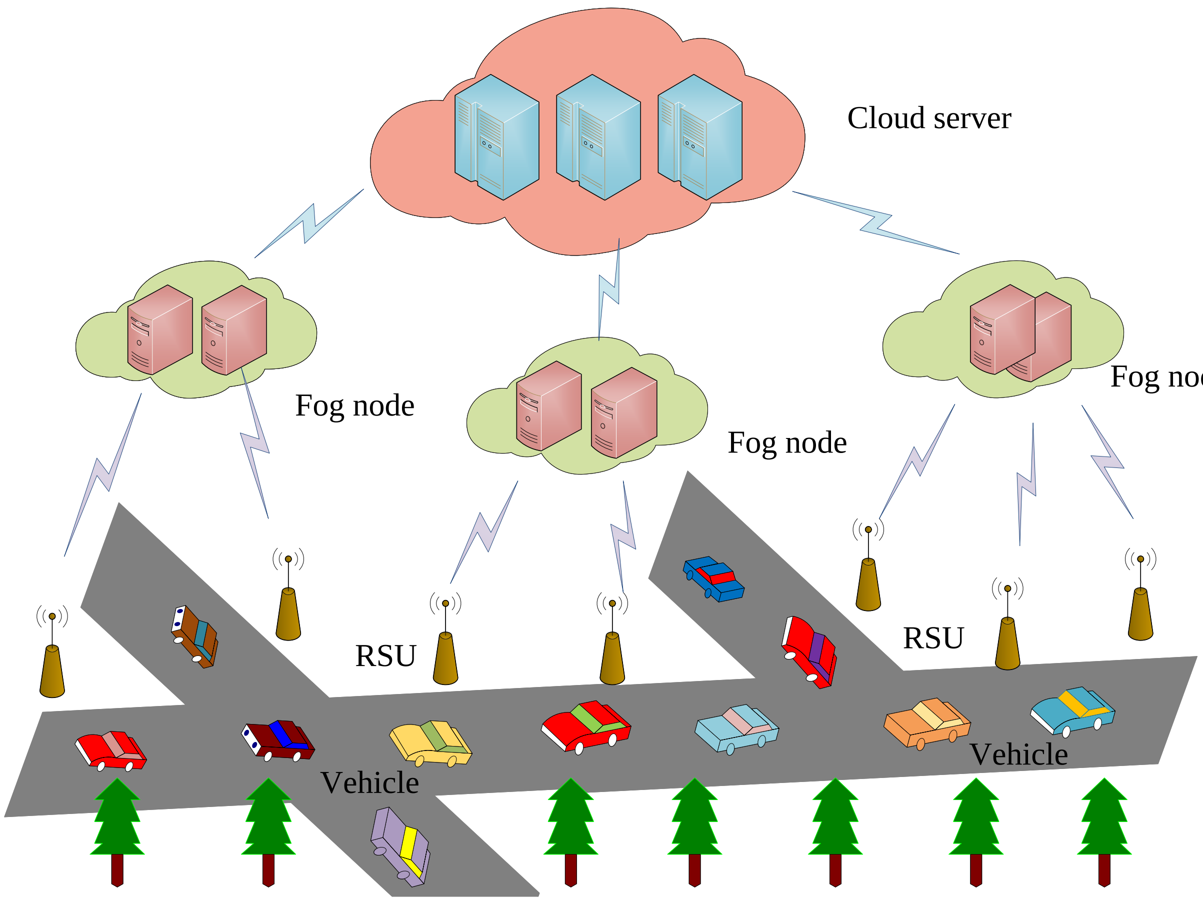


Figure 1. The four-layer architecture of the IoV based on fog computing.

To ensure secure communication and reduce the computational burden of the CS, we propose an authentication protocol under a four-layer architecture. We also apply SGX and a symmetric encryption algortihm to the proposed protocol to resist several well-known attacks. Our main contributions are as follows:

(1) We first propose a four-layer architecture in the IoV environment as shown in Figure 1. Adopting such an architecture reduces the computational pressure of the CS.
(2) We apply SGX to store the private value of the fog node and RSU in SGX so that even if the attacker obtains the data in the authentication table, he or she cannot obtain the private values in SGX. In other words, SGX can make the proposed protocol resist privileged insider attacks and enhance the security of the protocol.
(3) We prove the security of the proposed protocol through the Real-Or-Random model (ROR) and informal analysis. Furthermore, we compare the performance of the proposed protocol with recent protocols, with the results showing better security and a lower computational cost.

The rest of the paper is organized as follows. Section 2 briefly reviews the relevant research work. Section 3 describes the system model and proposed protocol in detail. In Section 4, we use the ROR model and informal analysis to prove the protocol's security. We compare the performance of the proposed protocol with recent protocols and discuss the obtained results in Section 5. Finally, we provide a brief summary of the content of this study in Section 6.

2. Related Work

Researchers have conducted extensive research on security challenges based on the IoV [20,21]. In 2007, Raya et al. [22] determined that information transmission security should be ensured in the IoV. Therefore, cryptographic technology, namely the public key infrastructure (PKI) mechanism, was introduced into the proposed protocol. Although this technology ensured that the vehicles privacy was not leaked, the computational cost was extremely high. In 2011, Huang et al. [23] designed an AKA protocol for value-added services known as ABAKA. The protocol realized anonymity but used elliptic curve cryptography (ECC), and the overhead remained extremely high. In 2017, Ying et al. [13] designed a protocol and claimed that it realized anonymous and secure communication. The protocol had a low computational cost and was lightweight. Mohit et al. [14] proposed a secure authentication protocol that configured vehicle sensors for the IoV to monitor the vehicle and the surrounding environment. However, Yu et al. [15] found that the protocol [14] was vulnerable to user impersonation attacks and could not provide anonymity or mutual authentication. Yu et al. [15] designed an improved protocol for ensuring communication security. However, Sadri et al. [24] found that the protocol [15] was vulnerable to user impersonation and sensor capture attacks and could not provide untraceability. Sadri et al. [24] designed a secure protocol based on this. In 2021, Jiang et al. [25] designed an authentication protocol based on a physical unclonable function (PUF). This protocol effectively combined biometrics and a PUF to achieve secure identification and authentication. In the same year, Kurma et al. [26] designed a new authentication protocol for the IoV. The protocol [26] was based on radio frequency identification, which increased the protocol security and used ECC at a high computational cost.

In previous years, to achieve efficient authentication, the literature [27,28] adopted the cloud computing architecture to different environments. Later, researchers found that fog computing is more suitable for the IoV than cloud computing with a large amount of real-time data to process. Therefore, scholars have applied fog computing to the IoV to reduce the computational burden of the CS. Wazid et al. [29] proposed a protocol based on authentication key management (AKM) between two different entities, namely AKM and the IoV, to realize security authentication in the IoV. Han et al. [30] proposed a security protocol based on fog computing [30], which had two highlights. One was that the vehicle and RSU were self-authenticated without the participation of the trusted authority (TA), which improved the communication efficiency. The other was that the fog node managed the pseudonym of the vehicle to realize privacy protection. Soleymani et al. [31] designed a message authentication protocol that used bilinear pairing primitives with a high overhead. The protocol [31] also used pseudonym management for privacy protection. Ma et al. [9], based on asymmetric cryptosystems and fog computing, proposed an AKA protocol. The protocol used ECC at a high computational cost, and they claimed that the protocol was provably secure. However, Eftekhari et al. [10] found that the protocol [9] was vulnerable to known session-specific temporary information disclosure, privileged insider, and stolen smart card attacks. Eftekhari et al. [10] proposed an improved lightweight protocol. Wu et al. [11] proposed an AKA protocol. The protocol also used ECC, which had a large computational cost and computational burden from the CS under the architecture used. In 2021, Wu et al. [12] designed a lightweight AKA protocol, which was also based on the above architecture and did not realize the function of fog nodes sharing the computational burden of the CS. The main works related to this paper are summarized in Table 1.

Table 1. Summary of authentication protocols.

Protocols	Cryptographic Techniques and Properties	Limitations
Ying et al. [13]	(1) One-way hash function (2) Anonymity	(1) Does not resist replay attacks (2) Does not resist offline identity guessing attacks (3) Does not resist stolen smart card attacks
Mohit et al. [14]	(1) One-way hash function (2) Based on smart card	(1) Does not resist impersonation attacks (2) Does not provide mutual authentication
Yu et al. [15]	One-way hash function	(1) Does not resist user impersonation attacks (2) Does not resist sensor capture attacks
Ma et al. [9]	(1) ECC (2) Based on smart card	(1) Does not resist internal attacks (2) Does not resist stolen smart card attacks (3) Does not resist known session-specific temporary information attacks
Wazid et al. [29]	(1) ECC (2) Anonymity	–
Eftekhari et al. [10]	(1) ECC (2) Anonymity	–
Wu et al. [11]	(1) One-way hash function (2) ECC	–
Wu et al. [12]	(1) One-way hash function (2) Three-factor (3) Based on smart card	–

3. The Proposed Protocol: SGXAP

This section introduces the system model and the proposed protocol in detail. Definitions and specific descriptions of the symbols used in the protocol are listed in Table 2.

Table 2. Notations used in the protocol.

Symbol	Description
V_i	The i-th vehicle
RSU_j	The j-th RSU
FS_m	The m-th fog node
CS	Cloud server
ID_i, ID_j, ID_f	Identities of V_i, RSU_j, and FS_m
K_{CS}	Secret key of CS
SK	Session key
$D_k()$ and $E_k()$	Symmetric encryption and decryption algortihm

3.1. System Model

The system includes four entities: the vehicle, RSU, fog node, and CS. In this model, the RSUs know which fog nodes they are deployed in, and the CS also knows which fog node RSU is deployed. These four entities, namely the definition, function, computing power, and storage capability of each role, are explained in detail below:

(1) **Vehicle:** This refers to the vehicle or vehicle user who selects the appropriate speed or driving route by acquiring the relevant real-time road information collected by the RSU.

(2) **RSU:** This is a semi-trusted device that collects real-time road condition information. It is arranged on both sides of the road and has weak computing power and storage

capacity. The RSU judges whether the vehicle is communicating with itself for the first time.

(3) **Fog node:** This is a semi-trusted entity with certain computing power. It quickly processes the road condition information collected by the RSU and can store data. As a third party, the fog node participates in communicating between the vehicle and RSU.

(4) **CS:** This is a semi-trusted entity which can realize the calculation or storage of a large amount of data. Here, the CS is the registration center, which participates in registering vehicle, RSU, and fog node. For Case 1, it is also the data transmitter.

When the vehicle wants to communicate with the RSU, there are two communication modes:

1. Case 1 is shown in Figure 2. The RSU judges that the vehicle communicates with itself for the first time and sends a data request to the CS. Then, the CS transmits the private value of the vehicle to the fog node and RSU. Here, the RSU stores the private data of the vehicle to judge whether the communication between the vehicle and itself is happening for the first time. The fog node stores information about the vehicle to realize the authentication process. Finally, the vehicle and RSU realize communication through the fog node.
2. Case 2 is shown in Figure 3. This case extends Case 1. Therefore, only the fog node (without the participation of the CS) can help the vehicle and RSU to realize communication. This is because in the first communication, the fog node and RSU know the private information of the vehicle. Here, this architecture dramatically reduces the computational burden of the CS.

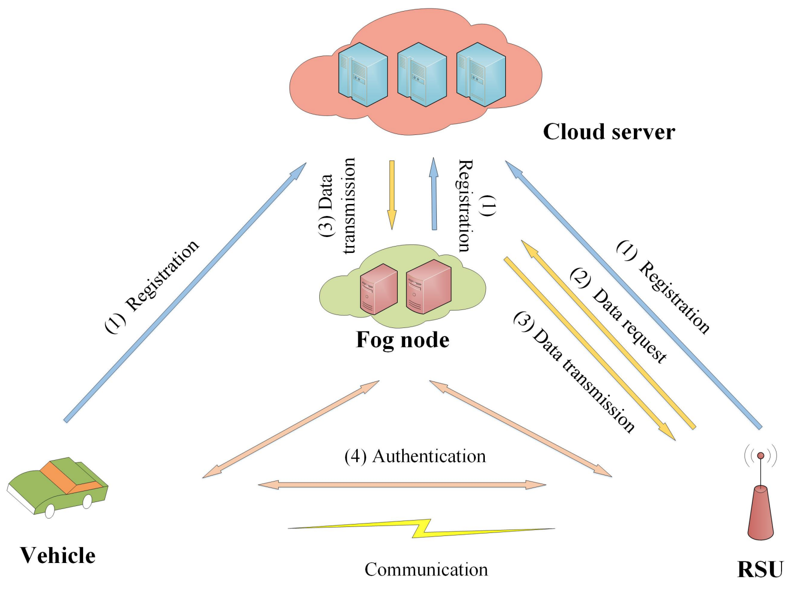


Figure 2. System model of Case 1.

3.2. *The Proposed SGXAP*

The proposed protocol comprises three phases: registration, login authentication, and data transmission.

3.2.1. Registration Phase

The registration phase of the proposed protocol includes the V_i, RSU_j, and FS_m registration phases.

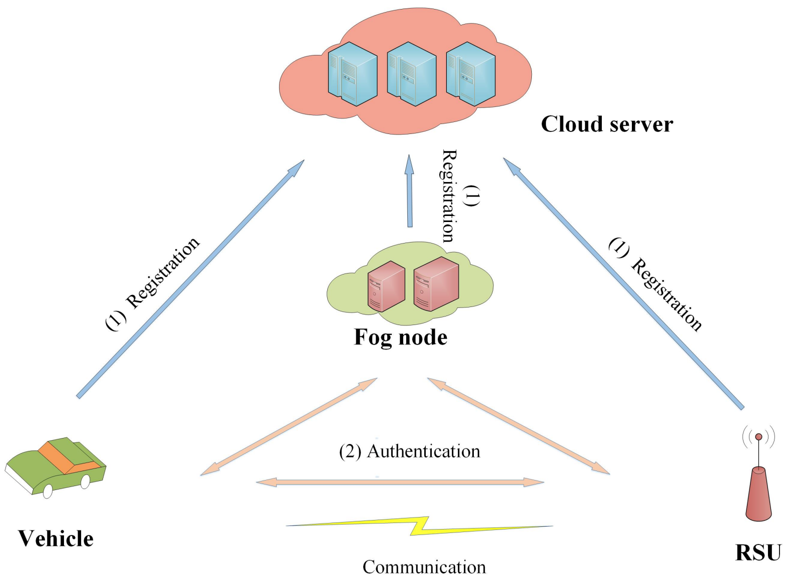


Figure 3. System model of Case 2.

In the V_i *registration phase*, V_i registers with the CS, as shown in Figure 4. The detailed registration steps are as follows:

V_i		CS
Selects ID_i, PSW_i, BIO_i, r_i $Gen(BIO_i) = (\sigma_i, \tau_i)$		
	$\xrightarrow{\{ID_i\}}$	
		Selects r_c $PID_i = h(ID_i \parallel r_c)$ $TK_{V-C} = h(PID_i \parallel K_{CS})$ CS stores $\{PID_i\}$ in database.
	$\xleftarrow{\{PID_i, TK_{V-C}\}}$	
$RPW_i = h(PSW_i \parallel \sigma_i)$ $TR_i = TK_{V-C} \oplus h(RPW_i \parallel r_i)$ $P_i = h(ID_i \parallel RPW_i \parallel r_i)$ V_i stores $\{PID_i, r_i, P_i, TR_i, \tau_i\}$ in SC.		

Figure 4. V_i registration phase.

(1) First, V_i selects the ID_i, password PSW_i, biometrics BIO_i, and random number r_i, computes $Gen(BIO_i) = (\sigma_i, \tau_i)$, and finally transmits $\{ID_i\}$ to the CS.

(2) After CS receives the message $\{ID_i\}$, it selects r_c, computes $PID_i = h(ID_i \parallel r_c)$ and $TK_{V-C} = h(PID_i \parallel K_{CS})$, and then saves the pseudo identity $\{PID_i\}$ in the database before finally transmitting $\{PID_i, TK_{V-C}\}$ to V_i.

(3) After V_i receives the message $\{PID_i, TK_{V-C}\}$, it computes $RPW_i = h(PSW_i \parallel \sigma_i)$, $TR_i = TK_{V-C} \oplus h(RPW_i \parallel r_i)$, and $P_i = h(ID_i \parallel RPW_i \parallel r_i)$. Finally, V_i stores $\{PID_i, r_i, P_i, TR_i, \tau_i\}$ in the smart card (SC).

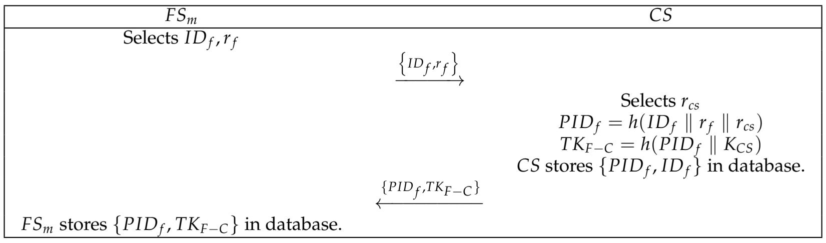
In the RSU_j *registration phase*, RSU_j registers with the CS, as shown in Figure 5. The detailed registration steps are as follows:

RSU_j		CS
Selects ID_j, r_j		
	$\xrightarrow{\{ID_j, r_j\}}$	
		Selects r_s $PID_j = h(ID_j \parallel r_j \parallel r_s)$ $TK_{R-C} = h(PID_j \parallel K_{CS})$ CS stores $\{PID_j, ID_j\}$ in database and sends $\{PID_j, TK_{R-C}, ID_j\}$ to the FS_m.
	$\xleftarrow{\{PID_j, TK_{R-C}\}}$	
RSU_j stores $\{PID_j, TK_{R-C}\}$ in database.		
		FS_m stores $\{PID_j, TK_{R-C}\}$ in SGX and stores $\{PID_j, ID_j\}$ in database.

Figure 5. RSU_j registration phase.

(1) First, RSU_j selects the ID_j and random number r_j and then transmits $\{ID_j, r_j\}$ to the CS.
(2) After the CS receives the message $\{ID_j, r_j\}$, it selects r_s, computes $PID_j = h(ID_j \parallel r_j \parallel r_s)$ and $TK_{R-C} = h(PID_j \parallel K_{CS})$, saves $\{PID_j, ID_j\}$ in the database, transmits $\{PID_j, TK_{R-C}\}$ to RSU_j, and sends $\{PID_j, TK_{R-C}, ID_j\}$ to FS_m.
(3) After RSU_j receives the message $\{PID_j, TK_{R-C}\}$, it stores $\{PID_j, TK_{R-C}\}$ in the database.
(4) After FS_m receives the message $\{PID_j, TK_{R-C}, ID_j\}$, it stores $\{PID_j, TK_{R-C}\}$ in SGX and stores $\{PID_j, ID_j\}$ in the database.

In the FS_m *registration phase*, FS_m registers with the CS, as shown in Figure 6. The detailed registration steps are as follows:



FS_m
CS
Selects ID_f, r_f
$\xrightarrow{\{ID_f, r_f\}}$
Selects r_{cs}
$PID_f = h(ID_f \parallel r_f \parallel r_{cs})$
$TK_{F-C} = h(PID_f \parallel K_{CS})$
CS stores $\{PID_f, ID_f\}$ in database.
$\xleftarrow{\{PID_f, TK_{F-C}\}}$
FS_m stores $\{PID_f, TK_{F-C}\}$ in database.

Figure 6. FS_m registration phase.

(1) First, FS_m selects the identity ID_f and random number r_f and then transmits $\{ID_f, r_f\}$ to the CS.
(2) After the CS receives the message $\{ID_f, r_f\}$, it selects r_{cs}, computes $PID_f = h(ID_f \parallel r_f \parallel r_{cs})$ and $TK_{F-C} = h(PID_f \parallel K_{CS})$, saves $\{PID_f, ID_f\}$ in the database, and transmits $\{PID_f, TK_{F-C}\}$ to FS_m.
(3) After FS_m receives the message $\{PID_f, TK_{F-C}\}$, it stores $\{PID_f, TK_{F-C}\}$ in the database.

3.2.2. Login and Authentication Phase

The vehicle and RSU achieve authentication and establish a session key with the assistance of the fog node to realize secure communication. The authentication of the proposed protocol has the following two cases:

1. **Case 1.** V_i communicates with RSU_j under FS_m for the first time. The authentication process needs the assistance of FS_m and the participation of the CS. When RSU_j receives the message M_1 sent by V_i, it cannot retrieve the private value of V_i through the pseudo-identity PID_i. Then, RSU_j enters the data transmission phase and sends a data request Req_j to the CS. After the CS successfully verifies RSU_j, it sends the private value of V_i to RSU_j and FS_m, and then RSU_j and FS_m store the private value. Finally, V_i, RSU_j, and FS_m continue to realize relevant authentication. The entire process includes the authentication phase of Figure 7 and the data transmission phase of Figure 8.
2. **Case 2.** V_i has communicated with RSU_j under FS_m. Therefore, the private value of V_i is stored in RSU_j and FS_m, and the entire authentication process can be realized without the CS participating. The process is the login authentication phase, as shown in Figure 7.

The entities in the authentication phase include V_i, RSU_j, and FS_m without the CS participating. The specific authentication steps are illustrated in Figure 7.

V_i	RSU_j	FS_m
Inputs ID_i, PSW_i, imprints BIO_i $\sigma_i' = Rep(BIO_i, \tau_i)$ $RPW_i = h(PSW_i \parallel \sigma_i')$ $P_i^* \overset{?}{=} h(ID_i \parallel RPW_i \parallel r_i)$ Selects N_i and T_1 Computes: $TK_{V-C} = TR_i \oplus h(RPW_i \parallel r_i)$ $TN_{V-C} = h(TK_{V-C} \parallel PID_i)$ $TV_{V-C} = (ID_i \parallel N_i) \oplus TN_{V-C}$ $V_1 = h(ID_i \parallel N_i \parallel TV_{V-C})$ $\xrightarrow{M_1=\{PID_i,TV_{V-C},V_1,T_1\}}$		
	Checks $\lvert T_1 - T_c \rvert \leq \Delta T$ Matches TK_{V-C} according to PID_i Selects N_j, T_2 Computes: $TN_{R-C} = h(TK_{R-C} \parallel PID_j)$ $TV_{R-C} = N_j \oplus TN_{R-C}$ $V_2 = h(ID_j \parallel N_j \parallel TV_{R-C})$ $\xrightarrow{M_2=\{PID_i,TV_{V-C},V_1,PID_j,TV_{R-C},V_2,T_2\}}$	
		Checks $\lvert T_2 - T_c \rvert \leq \Delta T$ Matches TK_{V-C} according to PID_i and compute: $TN_{V-C} = h(TK_{V-C} \parallel PID_i)$ Computes: $(N_i \parallel ID_i) = TN_{V-C} \oplus TV_{V-C}$ $V_1' \overset{?}{=} h(ID_i \parallel N_i \parallel TV_{V-C})$ Gets ID_j from table according to PID_j Matches TK_{R-C} according to PID_j and compute: $TN_{R-C} = h(TK_{R-C} \parallel PID_j)$ Computes: $N_j = TN_{R-C} \oplus TV_{R-C}$ $V_2' \overset{?}{=} h(ID_j \parallel N_j \parallel TV_{R-C})$ Selects T_3 $TV_{F-R} = (ID_i \parallel N_i) \oplus h(ID_j \parallel N_j)$ $V_3 = h(ID_j \parallel N_j \parallel TV_{F-R})$ $\xleftarrow{M_3=\{TV_{F-R},V_3,T_3\}}$
	Checks $\lvert T_3 - T_c \rvert \leq \Delta T$ Computes: $(ID_i \parallel N_i) = h(ID_j \parallel N_j) \oplus TV_{F-R}$ $SK = h(ID_i \parallel ID_j \parallel N_i \parallel N_j)$ $V_3' \overset{?}{=} h(ID_j \parallel N_j \parallel TV_{F-R})$ Selects T_4 $TV_{R-V} = (ID_j \parallel N_j) \oplus h(ID_i \parallel N_i)$ $V_4 = h(ID_i \parallel N_i \parallel TV_{R-V})$ $\xleftarrow{M_4=\{TV_{R-V},V_4,T_4\}}$	
Checks $\lvert T_4 - T_c \rvert \leq \Delta T$ Computes: $(ID_j \parallel N_j) = h(ID_i \parallel N_i) \oplus TV_{R-V}$ $SK = h(ID_i \parallel ID_j \parallel N_i \parallel N_j)$ $V_4' \overset{?}{=} h(ID_i \parallel N_i \parallel TV_{R-V})$		

Figure 7. Login and authentication phase.

(1) V_i first enters its own ID_i, PSW_i, and BIO_i, and the SC computes $\sigma_i' = Rep(BIO_i, \tau_i)$, $RPW_i = h(PSW_i \parallel \sigma_i')$, and $P_i^* = h(ID_i \parallel RPW_i \parallel r_i)$ and compares $P_i^* \overset{?}{=} P_i$. If they are equal, then the login is successful. Otherwise, the login fails. After successfully logging in to the SC, V_i selects a random number N_i and timestamp T_1 and computes $TK_{V-C} = TR_i \oplus h(RPW_i \parallel r_i)$, $TN_{V-C} = h(TK_{V-C} \parallel PID_i)$, $TV_{V-C} = (ID_i \parallel N_i) \oplus TN_{V-C}$, and $V_1 = h(ID_i \parallel N_i \parallel TV_{V-C})$. Finally, V_i sends the message $M_1 = \{PID_i, TV_{V-C}, V_1, T_1\}$ to RSU_j.

(2) After RSU_j receives M_1 from V_i, it first checks the freshness of the timestamp T_1. If the limit is exceeded, then the authentication is suspended. Otherwise, the authentication continues. Then, RSU_j indexes TK_{V-C} stored in SGX according to PID_i. If it cannot be indexed, then it indicates that V_i is communicating with RSU_j for the first time. Here, RSU_j sends a data request to the CS, requesting the CS to send the private value of V_i to itself and FS_m, as shown in Figure 8, and then continue to realize the authentication process. If it can be indexed, V_i is not communicating with RSU_j for the first time, and the authentication process continues. Later, RSU_j selects N_j, and T_2 computes $TN_{R-C} = h(TK_{R-C} \parallel PID_j)$, $TV_{R-C} = N_j \oplus TN_{R-C}$, and $V_2 = h(ID_j \parallel N_j \parallel TV_{R-C})$. Finally, RSU_j sends the message $M_2 = \{PID_i, TV_{V-C}, V_1, PID_j, TV_{R-C}, V_2, T_2\}$ to FS_m.

(3) After FS_m receives M_2 from RSU_j, it checks the freshness of T_2 and then sends PID_i to the security interface of SGX. SGX matches the secret value TK_{V-C} according to

PID_i, computes $TN_{V-C} = h(TK_{V-C} \parallel PID_i)$, and outputs the secret value TN_{V-C} from the secure interface after the computation is completed. Then, FS_m computes $(N_i \parallel ID_i) = TN_{V-C} \oplus TV_{V-C}$ and $V_1' = h(ID_i \parallel N_i \parallel TV_{V-C})$ and compares $V_1' \overset{?}{=} V_1$. If they are equal, this indicates that V_i is legal. Otherwise, the authentication is suspended. FS_m finds the identity ID_j according to PID_j and then sends PID_j to the security interface of SGX. SGX matches the secret value TK_{R-C} according to PID_j, computes $TN_{R-C} = h(TK_{R-C} \parallel PID_j)$, and outputs the secret value TN_{R-C} from the secure interface. Then, FS_m computes $N_j = TN_{R-C} \oplus TV_{R-C}$, $V_2' = h(ID_j \parallel N_j \parallel TV_{R-C})$ and compares $V_2' \overset{?}{=} V_2$. If they are equal, RSU_j is legal. Otherwise, the authentication is suspended. After authenticating V_i and RSU_j, FS_m selects timestamp T_3, computes $TV_{F-R} = (ID_i \parallel N_i) \oplus h(ID_j \parallel N_j)$, $V_3 = (ID_j \parallel N_j \parallel TV_{F-R})$, and finally sends the message $M_3 = \{TV_{F-R}, V_3, T_3\}$ to RSU_j.

(4) After RSU_j receives the message M_3 from FS_m, it checks the freshness of T_3 and computes $(ID_i \parallel N_i) = h(ID_j \parallel N_j) \oplus TV_{F-R}$, $SK = h(ID_i \parallel ID_j \parallel N_i \parallel N_j)$, and $V_3' = h(ID_j \parallel N_j \parallel TV_{F-R})$. Then, FS_m compares $V_3' \overset{?}{=} V_3$. If they are equal, FS_m is legal. Otherwise, the authentication fails. After the authentication is successful, RSU_j selects the timestamp T_4 and computes $TV_{R-V} = (ID_j \parallel N_j) \oplus h(ID_i \parallel N_i)$ and $V_4 = h(ID_i \parallel N_i \parallel TV_{R-V})$. Finally, the message $M_6 = \{TV_{R-V}, V_4, T_4\}$ is sent to V_i.

(5) After V_i receives M_4 from RSU_j, it first checks the freshness of T_4 and then computes $(ID_j \parallel N_j) = h(ID_i \parallel N_i) \oplus TV_{R-V}$, $SK = h(ID_i \parallel ID_j \parallel N_i \parallel N_j)$, and $V_4' = h(ID_i \parallel N_i \parallel TV_{R-V})$. Then, V_i compares $V_4' \overset{?}{=} V_4$. If they are equal, RSU_j is legal. Otherwise, the authentication fails.

3.2.3. Data Transmission Phase

When V_i communicates with RSU_j under FS_m for the first time, the private value of V_i is not stored in RSU_j or FS_m. Thus, RSU_j requests private data from the CS. The entities in this phase include RSU_j, FS_m, and the CS. According to Case 1, when RSU_j receives the message M_1 sent by V_i and does not retrieve the privacy value of V_i through PID_i, RSU_j requests data from the CS, as shown in Figure 8. The specific steps are as follows:

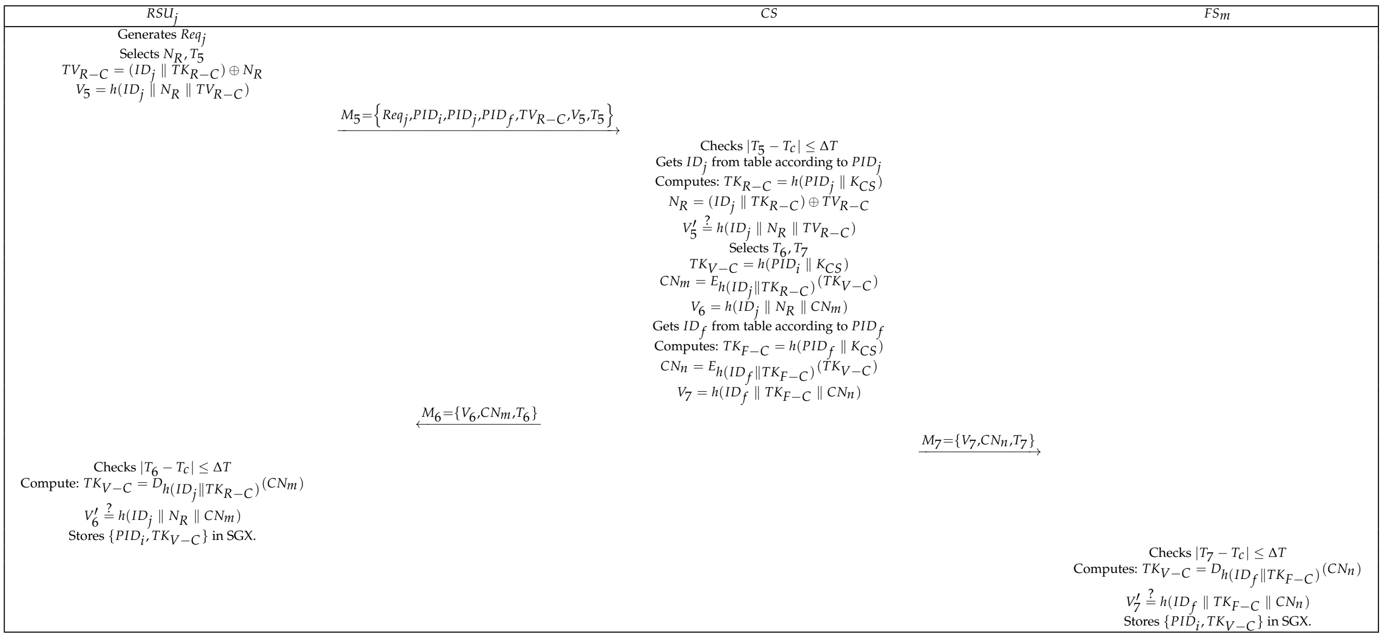
(1) RSU_j generates a data request Req_j, selects N_R and T_5, and computes $TV_{R-C} = (ID_j \parallel TK_{R-C}) \oplus N_R$ and $V_5 = h(ID_j \parallel N_R \parallel TV_{R-C})$. Finally, the message $M_5 = \{Req_j, PID_i, PID_j, PID_f, TV_{R-C}, V_5, T_5\}$ is sent to the CS.

(2) After the CS receives M_5 from RSU_j, it checks the freshness of T_5. After finding ID_j according to PID_j, the CS computes $TK_{R-C} = h(PID_j \parallel K_{CS})$, $N_R = (ID_j \parallel TK_{R-C}) \oplus TV_{R-C}$, and $V_5' = h(ID_j \parallel N_R \parallel TV_{R-C})$ and compares $V_5' \overset{?}{=} V_5$. If they are equal, RSU_j is legal. Otherwise, the authentication is suspended. Then, CS selects T_6 and T_7 and computes $TK_{V-C} = h(PID_i \parallel K_{CS})$, $CN_m = E_{h(ID_j \parallel TK_{R-C})}(TK_{V-C})$, and $V_6 = h(ID_j \parallel N_R \parallel CN_m)$. The CS finds ID_f according to PID_f and computes $TK_{F-C} = h(PID_f \parallel K_{CS})$, $CN_n = E_{h(ID_f \parallel TK_{F-C})}(TK_{V-C})$, and $V_7 = h(ID_f \parallel TK_{F-C} \parallel CN_n)$. Finally, the message $M_6 = \{V_6, CN_m, T_6\}$ is sent to RSU_j, and $M_7 = \{V_7, CN_n, T_7\}$ is sent to FS_m.

(3) After RSU_j receives M_6 from the CS, it checks the freshness of T_6, computes $TK_{V-C} = D_{h(ID_j \parallel TK_{R-C})}(CN_m)$ and $V_6' = h(ID_j \parallel N_R \parallel CN_m)$, and compares $V_6' \overset{?}{=} V_6$. If they are equal, then CS is legal. Otherwise, the authentication is suspended. Finally, RSU_j stores $\{PID_i, TK_{V-C}\}$ in SGX.

(4) After FS_m receives the message M_7 from the CS, it checks the freshness of T_7, computes $TK_{V-C} = D_{h(ID_f \parallel TK_{F-C})}(CN_n)$ and $V_7' \overset{?}{=} h(ID_f \parallel TK_{F-C} \parallel CN_n)$, and compares $V_7' \overset{?}{=} V_7$. If they are equal, then the CS is legal. Otherwise, the authentication is suspended. Finally, FS_m stores $\{PID_i, TK_{V-C}\}$ in SGX.

After completing the above data request, V_i, RSU_j, and FS_m continue the authentication process.



RSU_j
CS
FS_m
Generates Req_j
Selects N_R, T_5
$TV_{R-C} = (ID_j \parallel TK_{R-C}) \oplus N_R$
$V_5 = h(ID_j \parallel N_R \parallel TV_{R-C})$
$M_5=\{Req_j, PID_i, PID_j, PID_f, TV_{R-C}, V_5, T_5\}$
Checks $|T_5 - T_c| \leq \Delta T$
Gets ID_j from table according to PID_j
Computes: $TK_{R-C} = h(PID_j \parallel K_{CS})$
$N_R = (ID_j \parallel TK_{R-C}) \oplus TV_{R-C}$
$V_5' \overset{?}{=} h(ID_j \parallel N_R \parallel TV_{R-C})$
Selects T_6, T_7
$TK_{V-C} = h(PID_i \parallel K_{CS})$
$CNm = E_{h(ID_j \parallel TK_{R-C})}(TK_{V-C})$
$V_6 = h(ID_j \parallel N_R \parallel CNm)$
Gets ID_f from table according to PID_f
Computes: $TK_{F-C} = h(PID_f \parallel K_{CS})$
$CNn = E_{h(ID_f \parallel TK_{F-C})}(TK_{V-C})$
$V_7 = h(ID_f \parallel TK_{F-C} \parallel CNn)$
$M_6=\{V_6, CNm, T_6\}$
$M_7=\{V_7, CNn, T_7\}$
Checks $|T_6 - T_c| \leq \Delta T$
Compute: $TK_{V-C} = D_{h(ID_j \parallel TK_{R-C})}(CNm)$
$V_6' \overset{?}{=} h(ID_j \parallel N_R \parallel CNm)$
Stores $\{PID_i, TK_{V-C}\}$ in SGX.
Checks $|T_7 - T_c| \leq \Delta T$
Computes: $TK_{V-C} = D_{h(ID_f \parallel TK_{F-C})}(CNn)$
$V_7' \overset{?}{=} h(ID_f \parallel TK_{F-C} \parallel CNn)$
Stores $\{PID_i, TK_{V-C}\}$ in SGX.

Figure 8. Data transmission phase.

4. Security Analysis

4.1. Formal Security Analysis

The ROR model, proposed by Canetti et al. [32], calculates the probability that the attacker (A) can break the SK through multiple query operations to prove the security of the protocol. Here, we compute the probability of cracking the SK and prove that our protocol is secure.

4.1.1. Adversary Model

We used the Dolev–Yao and Canetti–Krawczyk models to define the capabilities of A [33,34]. The specific capabilities are as follows:

(1) A can intercept, interrupt, forge, and replay the information transmitted on the common channel.
(2) A can act as malicious insiders of the fog nodes and CS to obtain internal information.
(3) A can guess the identities or passwords of vehicles by violent cracking.
(4) A can obtain the values in the vehicle's smart card.
(5) A can obtain the private key of the CS and a random number of four entities.

Here, we use Π_V^m, Π_{RSU}^n, and Π_{FS}^z to represent the m-th V_i, n-th RSU_j, and z-th FS_m instances, respectively. Here, we assume that the query capabilities of A are $Y = \{\Pi_V^m$, Π_{RSU}^n, and $\Pi_{FS}^z\}$:

(1) $Execute(Y)$: A intercepts messages $\{M_1, M_2, M_3, M_4\}$ transmitted on the common channel;
(2) $Send(Y, M)$: A sends message M to entity Y and receives a response;
(3) $Hash(string)$: A enters a character string of any length and returns the corresponding hash value;
(4) $Corrupt(Y)$: A can obtain the private value of an entity;
(5) $Test(Y)$: A flips a coin C. If $C = 1$, then A can obtain SK. If $C = 0$, A can obtain any string of the same length as the SK.

4.1.2. Security Requirements

The secure AKA protocol should meet the following security requirements:

1. *User anonymity and untraceability*: A can neither obtain the real identity of the communication entity nor trace the session key of the communication process through the data transmitted on the public channel.
2. *Resistance of common attacks*: The secure AKA protocol should be able to resist the following attacks:

(1) *Privileged insider attacks*: As an A, the insiders of the communication entity spy on the private data stored in the database to disguise as a legal entity or obtain the session key;
(2) *Impersonation attacks*: A intercepts and decrypts the data transmitted on the public channel, disguises as a legal entity, communicates with other entities, and establishes a session key;
(3) *Known temporary information disclosure attacks*: A calculates the session key in the communication process by obtaining the random number of an entity and the data transmitted on the public channel;
(4) *Man-in-the-middle attacks*: A intercepts the data transmitted on the public channel, tampers with the data, and establishes the session key with the legal entity without the knowledge of both parties of the communication entity;
(5) *Offline password guessing attacks*: A intercepts the verification value containing the password stored on the public channel or in the smart device, repeatedly guesses the password, calculates the verification value in the offline state, and compares it with the intercepted verification value until the two values are equal;
(6) *Replay attacks*: A repeatedly sends the message transmitted on the public channel to the communication entity so as to deceive the entity and interfere with normal communication;
(7) *Stolen smart card attacks*: After A steals the smart card and obtains the parameters about the entity identity before using the parameters to launch camouflage attacks or malicious acts.

Theorem 1. *Suppose that A can execute the above queries and the probability that A can break the proposed protocol P in polynomial time is $adv_{\mathcal{A}}^{\mathcal{P}}(\xi) \leq q_{send}/2^{l-1} + 3q_{hash}^2/2^l + 2max\{C' \cdot q_{send}^{s'}, q_{send}/2^l\}$. Here, q_{send} refers to the number of queries executed, q_{hash} refers to the number of times the hash is executed, l refers to the bit length of the biological information, and C' and s' refer to two constants.*

Proof. We define seven games GM_0–GM_6 to simulate the attack process of A. In the proof, $Succ_A^{GM_i}(\xi)$ represents the probability that A can win multiple rounds of the game. The process of A simulating the query is shown in Table 3. The proof steps are as follows:

GM_0: In the ROR model, the simulation of GM_0 is consistent with a real attack. Therefore, we have

$$Adv_A^P = |2Pr[Succ_A^{GM_0}] - 1|. \tag{1}$$

GM_1: GM_1 and GM_0 are different from the GM_1 add Execute() operation. In GM_1, A can intercept $\{M_1, M_2, M_3, M_4\}$ transmitted on the common channel. When GM_1 ends, A executes a Test() query to compute the SK, where $SK = h(ID_i \parallel ID_j \parallel N_i \parallel N_j)$. As $\{ID_i, ID_j, N_i, N_j\}$ is confidential to A, the probability of GM_1 is equal to GM_0. The probability of GM_1 is

$$Pr[Succ_A^{GM_1}] = Pr[Succ_A^{GM_0}]. \tag{2}$$

GM_2: The difference between GM_2 and GM_1 is that GM_2 adds the Send() operation. According to Zipf's law [35], the probability of GM_2 is expressed as

$$|Pr[Succ_{\mathcal{A}}^{GM_2}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_1}(\xi)]| \leq q_{send}/2^l. \tag{3}$$

GM_3: GM_3 adds the Hash() operation and reduces the Send() operation. According to the birthday paradox, the probability of GM_3 is

$$|Pr[Succ_{\mathcal{A}}^{GM_3}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_2}(\xi)]| \leq q_{hash}^2/2^{l+1}. \tag{4}$$

GM_4: In GM_4, A obtains temporary information to verify that it is resistant to known temporary information disclosure attacks. A can obtain a random number for one of the two parties: Π_V^m and Π_{RSU}^n. Suppose A obtains a random number N_i. As ID_i, ID_j, and N_j are unknown, the SK cannot be computed. Similarly, if the random number N_j is leaked, then the SK cannot be computed by A. Therefore, the probability of GM_4 is expressed as

$$|Pr[Succ_{\mathcal{A}}^{GM_4}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_3}(\xi)]| \leq q_{hash}^2/2^{l+1}. \tag{5}$$

GM_5: In this game, A executes the Corrupt(Π_V^m) query to obtain the parameters $\{PID_i, r_i, P_i, TR_i, \tau_i\}$ in the smart card. Legitimate users typically use low-entropy passwords. A may attempt to extract the password PSW_i by executing an offline password guessing attack using the parameters $\{PID_i, r_i, P_i, TR_i, \tau_i\}$. However, in our protocol, A cannot obtain PSW_i without the biometric information τ_i and secret credential RPW_i. The probability of A guessing one bit of biological information is $1/2^l$. According to Zipf's law [35], when $q^{send} \leq 10^6$, the probability that A can guess a password is greater than 0.5. These results prove that the proposed protocol is resistant to offline password guessing attacks. Therefore, we can derive

$$|Pr[Succ_A^{GM_5}] - Pr[Succ_A^{GM_4})]| \leq max\{C' \cdot q_{send}^{s'}, q_{send}/2^l\}. \tag{6}$$

GM_6: This game verifies whether the proposed protocol is resistant to impersonation attacks. The difference between GM_6 and GM_5 is that A uses $h(ID_i \parallel ID_j \parallel N_i \parallel N_j)$ for the query operation, and the probability of successfully obtaining SK is

$$|Pr[Succ_{\mathcal{A}}^{GM_6}(\xi)] - Pr[Succ_{\mathcal{A}}^{GM_5}(\xi)]| \leq q_{hash}^2/2^{l+1}. \tag{7}$$

The probability of the success and failure of GM_6 is $1/2$. Therefore, the probability that A can guess SK is

$$Pr[Succ_{\mathcal{A}}^{GM_6}(\xi)] = 1/2. \tag{8}$$

Using these formulas, we obtain

$$\begin{aligned} 1/2Adv_A^P &= |Pr[Succ_A^{GM_0}] - 1/2| \\ &= |Pr[Succ_A^{GM_0}] - Pr[Succ_A^{GM_6}]| \\ &= |Pr[Succ_A^{GM_1}] - Pr[Succ_A^{GM_6}]| \\ &\leq \sum_{i=0}^{5} |Pr[Succ_A^{GM_{i+1}}] - Pr[Succ_A^{GM_i}]| \\ &= q_{send}/2^l + 3q_{hash}^2/2^{l+1} + max\{C' \cdot q_{send}^{s'}, q_{send}/2^l\} \end{aligned} \tag{9}$$

Therefore, we can obtain

$$Adv_A^P \leq q_{send}/2^{l-1} + 3q_{hash}^2/2^l + 2max\{C' \cdot q_{send}^{s'}, q_{send}/2^l\}. \tag{10}$$

□

Table 3. The process of *Send*, *Execute*, *Corrupt*, and *Test* queries.

Query	Description
$Send(Y, M)$	On a query $Send(\Pi_V^m, start)$, we assume Π_V^m is a normal state. Π_V^m selects N_i, T_1 and computes $TK_{V-C} = TR_i \oplus h(RPW_i \parallel r_i)$, $TN_{V-C} = h(TK_{V-C} \parallel PID_i)$, $TV_{V-C} = (ID_i \parallel N_i) \oplus TN_{V-C}$, and $V_1 = h(ID_i \parallel N_i \parallel TV_{V-C})$. Then, $Send(\Pi_V^m, start)$ returns $M_1 = \{PID_i, TV_{V-C}, V_1, T_1\}$.
	On a query $Send(\Pi_{FS}^z, (PID_i, TV_{V-C}, V_1, PID_j, TV_{R-C}, V_2, T_2))$, Π_{FS}^z computes $TN_{V-C}, (N_i \parallel ID_i)$, and checks V_1. If the verification holds, continue to calculate TN_{R-C}, N_j and check V_2. If it is equal, select T_3 and compute TV_{F-R}, V_3. Then, $Send(\Pi_{FS}^z, M_2)$ returns $M_3 = \{TV_{F-R}, V_3, T_3\}$.
	On a query $Send(\Pi_{RSU}^n, (TV_{F-R}, V_3, T_3))$, Π_{RSU}^n computes $(ID_i \parallel N_i), SK$, and V_3 and checks V_3. If V_3 holds, Π_{RSU}^n selects T_4 and computes TV_{R-V}, V_4. Then, $Send(\Pi_{RSU}^n, M_3))$ returns $M_4 = \{TV_{R-V}, V_4, T_4\}$.
	On a query $Send(\Pi_V^m, TV_{R-V}, V_4, T_4)$, Π_V^m computes $(ID_j \parallel N_j), SK$, and V_4 and checks V_4. If it is not equal, then the query process is terminated. Otherwise, Π_V^m accepts and terminates.
$Execute(Y)$	On an *Execute* query, we continue with the *Send* query simulation as follows: $(PID_i, TV_{V-C}, V_1, T_1) \longleftarrow Send(\Pi_V^m, start)$, $(PID_i, TV_{V-C}, V_1, PID_j, TV_{R-C}, V_2, T_2) \longleftarrow Send(\Pi_{RSU}^n, (PID_i, TV_{V-C}, V_1, T_1))$, $(TV_{F-R}, V_3, T_3) \longleftarrow Send(\Pi_{FS}^z, (PID_i, TV_{V-C}, V_1, PID_j, TV_{R-C}, V_2, T_2))$, $(TV_{R-V}, V_4, T_4) \longleftarrow Send(\Pi_{RSU}^n, (TV_{F-R}, V_3, T_3))$. The query returns $(PID_i, TV_{V-C}, V_1, T_1), (PID_i, TV_{V-C}, V_1, PID_j, TV_{R-C}, V_2, T_2), (TV_{F-R}, V_3, T_3)$, and (TV_{R-V}, V_4, T_4).
$Corrupt(Y)$	On a $Corrupt(\Pi_V^m)$ query, if Π_V^m is accepted, then it returns the private information $\{PID_i, r_i, P_i, TR_i, \tau_i\}$ of vehicle V.
$Test(Y)$	On a *Test* query, to flip a coin C, if $C = 1$, A can obtain SK. If $C = 0$, A can obtain any string of the same length as SK.

4.2. Informal Security Analysis

4.2.1. Mutual Authentication

The proposed protocol realizes mutual authentication using $\{V_1, V_2, V_3, V_4\}$. FS_m uses V_1 to verify the legitimacy of V_i and V_2 to verify the legitimacy of RSU_j. RSU_j uses V_3 to verify the legitimacy of FS_m, and V_i uses V_4 to verify the legitimacy of RSU_j. Therefore, the proposed protocol can achieve mutual authentication.

4.2.2. Replay Attacks

The timestamps $\{T_1, T_2, T_3, T_4\}$ are used by the protocol to resist replay attacks. Here, we consider T_1 as an example. When RSU_j receives message M_1 of V_i, it first checks the freshness of T_1. If T_1 is valid, then the authentication continues. Otherwise, authentication is suspended. Suppose A intercepts M_1 and repeatedly sends it to RSU_j. When RSU_j checks the freshness of T_1, T_1 exceeds this time, and the authentication process stops. Therefore, the proposed protocol can resist replay attacks.

4.2.3. Privileged Insider Attacks

Suppose that A can obtain the value from a party's database. Here, we consider FS_m as an example. Based on this assumption, A can obtain the values $\{PID_f, TK_{F-C}, PID_j, ID_i\}$ stored in the database. However, because the protocol uses a secure hardware SGX, the values $\{TK_{R-C}, TK_{V-C}\}$ stored in SGX are not available. Therefore, A cannot obtain the value $\{ID_i, ID_j, N_i, N_j\}$ required to compute the SK, where $SK = h(ID_i \parallel ID_j \parallel N_i \parallel N_j)$. Therefore, the proposed protocol can resist privileged insider attacks.

4.2.4. Man-in-the-Middle Attacks

Suppose that A can intercept $\{M_1, M_2, M_3, M_4\}$ in the common channel. Here, we consider an intercept M_1 between V_i and FS_m as an example. Because A does not know the values of $\{ID_i, N_i, RPW_i\}$ or $\{r_i, TR_i\}$ in the smart card, A cannot compute the values of $\{ID_i, N_i, TV_{V-C}\}$ required by V_1, where $V_1 = h(ID_i \parallel N_i \parallel TV_{V-C})$. Thus, the legitimacy of V_i cannot be verified in FS_m. The same applies for A attempting to intercept $\{M_2, M_3, M_4\}$. Therefore, the proposed protocol can resist man-in-the-middle attacks.

4.2.5. User Anonymity and Untraceability

In our protocol, the identities of V_i, RSU_j, and FS_m are not transmitted to the common channel, but pseudo-identities $\{PID_i, PID_j, PID_f\}$ are transmitted. A cannot know the identities of the three, thus realizing anonymity. Because random numbers N_i and N_j

used in $\{M_1, M_2, M_3, M_4\}$ are variable in every session, A cannot track V_i, RSU_j, or FS_m. Therefore, our protocol provides anonymity and untraceability.

5. Comparisons and Discussions

In this section, we compare the proposed protocol with the AKA protocols proposed by Ma et al. [9], Wazid et al. [29], Eftekhari et al. [10], and Wu et al. [11] in terms of security and performance.

5.1. Security Comparisons

Table 4 presents the comparison results in terms of security. Here, ✓ indicates that the protocol can resist the attack, × indicates that the protocol is vulnerable to the attack, and − demonstrates that it is not mentioned whether the protocol can resist it. As shown in the table, the protocol of Ma et al. [9] is vulnerable to privileged insider attacks, known specific temporary information disclosure attacks, and stolen smart card attacks. The protocol proposed by Wazid et al. [29] is vulnerable to impersonation attacks. The other protocols and the proposed protocol are secure.

Table 4. Comparisons of security.

Security Properties	[9]	[29]	[10]	[11]	Ours
Privileged insider attacks	×	✓	−	✓	✓
Impersonation attacks	✓	×	✓	✓	✓
Known temporary information disclosure attacks	×	−	✓	✓	✓
Stolen smart card attacks	×	✓	−	−	✓
User anonymity	×	✓	✓	✓	✓
Man-in-the-middle attacks	✓	−	−	✓	✓
Untraceability	×	✓	✓	✓	✓
Offline password guessing attacks	−	✓	−	✓	✓
Replay attacks	✓	✓	✓	✓	✓

5.2. Performance Comparison

In this part, we compare the proposed protocol with the current AKA protocols for computational and communication costs. Because the authentication phase of the proposed protocol has two cases, the calculations of the computational and communication costs are also divided into two cases.

To calculate the computational cost, we considered two cases of the proposed protocol as examples. In Case 1, RSU_j and FS_m have no privacy value of V_i. Therefore, the CS must transmit the privacy value to RSU_j and FS_m to realize the entire authentication process. This situation requires the participation of the CS. Therefore, when calculating the computational cost, in addition to calculating the computational cost of the three parties involved in the authentication phase, we also needed to calculate the computational cost of the CS. In Case 2, RSU_j and FS_m have the private value of V_i and do not require the participation of the CS; only the computational costs of V_i, RSU_j, and FS_m need to be calculated.

When comparing the computational cost, we estimated the computational time of each entity in the protocol through a simulation experiment. We used MI 8 to simulate the vehicle, a Lenovo laptop to simulate the RSU and fog node, and a Lenovo desktop computer to simulate the cloud server. The equipment configuration of the three devices is shown in Table 5. The simulation experiment used the average execution time of the three devices 10 times as the running time, as shown in Table 6. Here, the $\oplus$ and $\|$ operations were negligibly small, and the execution time of fuzzy extraction was similar to that of the hash function, according to [36]. It is shown in [19] that the average running time of the system with SGX only increases by 20 μs, sufficiently showing the low computation cost of SGX. Hence, we ignored the computational cost of SGX in the following comparisons.

The results of the comparison of the computational cost are listed in Table 7. It is evident from Table 7 that for V_i and FS_m, excluding our protocol, the other protocols performed point multiplication, so the computational cost of our protocol was the lowest. For RSU_j, only our protocol contained the RSU. For the CS, the computational cost of Wazid et al. [29] was smaller than that of our protocol, whereas the others were higher than that of our protocol. As the CS had strong computing power, it did not affect the protocol's performance. Therefore, the computational cost of the proposed protocol was relatively small.

Table 5. Experimental environment.

	MI 8	Lenovo Desktop Computer	Lenovo Laptop
Operating System	Android system	Windows 10	Windows 10
CPU	Qualcomm Snapdragon 845	Intel(R) Core(TM) i5-9500 CPU @ 3.00 GHz	Intel(R) Core(TM) i7-6700HQ CPU @ 2.60 GHz
Memory	6 GB	8 GB RAM	16 GB RAM

Table 6. Experimental results.

Operations	V_i (ms)	RSU_j/FS_m (ms)	CS (ms)
ECC scalar multiplication	20	18	12
ECC point addition	0.1556	0.0731	0.0500
Symmetric key encryption and decryption	0.2263	0.1648	0.1384
Hash function	0.0042	0.0030	0.0024

Table 7. Computational cost comparison[1].

Protocol	V_i (ms)	RSU_j (ms)	FS_m (ms)	CS (ms)
Ma et al. [9]	$3T_{sm} + 4T_{ha} \approx 60.017$	-	$4T_{sm} + 4T_{ha} \approx 42.012$	$10T_{sm} + 11T_{ha} \approx 120.026$
Wazid et al. [29]	$3T_{sm} + 2T_{fe} + 22T_{ha} \approx 60.100$	-	$2T_{sm} + T_{pa} + 14T_{ha} \approx 36.115$	$3T_{ha} \approx 0.007$
Eftekhari et al. [10]	$3T_{sm} + T_{pa} + 11T_{ha} \approx 60.202$	-	$3T_{sm} + T_{pa} + 12T_{ha} \approx 54.109$	$3T_{sm} + 2T_{pa} + 15T_{ha} \approx 36.136$
Wu et al. [11]	$2T_{sm} + T_{fe} + 8T_{ha} \approx 60.038$	-	$4T_{sm} + 5T_{ha} \approx 72.015$	$4T_{sm} + 13T_{ha} \approx 48.031$
Ours, Case 1	$T_{fe} + 8T_{ha} \approx 0.038$	$T_{en} + 9T_{ha} \approx 0.092$	$T_{en} + 7T_{ha} \approx 0.186$	$2T_{en} + 6T_{ha} \approx 0.291$
Ours, Case 2	$T_{fe} + 8T_{ha} \approx 0.038$	$7T_{ha} \approx 0.021$	$6T_{ha} \approx 0.018$	-

[1] Here, T_{sm} indicates the running time of ECC scalar multiplication, T_{pa} indicates running time of ECC point addition, T_{fe} indicates running time of the fuzzy extraction operation, T_{en} indicates the running time of encryption and decryption, and T_{ha} indicates the running time of the hash function.

When comparing the communication cost, the length of the timestamp was regarded as 32 bits, the length of the identity and random number was 160 bits, that of the hash function and symmetric encryption and decryption was 256 bits, and that of the ECC point was 320 bits. The proposed protocol was considered an example to illustrate the calculation method. The communication cost was calculated based on the two cases of the protocol. For Case 1, we needed to calculate the communication cost in the authentication phase and the communication cost in the data transmission phase. In Case 1, the protocol transmitted seven messages on the common channel, including $M_1 = \{PID_i, TV_{V-C}, V_1, T_1\}$, $M_2 = \{PID_i, TV_{V-C}, V_1, PID_j, TV_{R-C}, V_2, T_2\}$, $M_3 = \{TV_{F-R}, V_3, T_3\}$, $M_4 = \{TV_{R-V}, V_4, T_4\}$, $M_5 = \{Req_j, PID_i, PID_j, PID_f, TV_{R-C}, V_5, T_5\}$,$M_6 = \{V_6, CN_m, T_6\}$,and $M_7 = \{V_7, CN_n, T_7\}$. Here, $\{PID_i, TV_{V-C}, PID_j, TV_{R-C}, TV_{F-R}, TV_{R-V}, Req_f, PID_f, TV_{R-C}\}$ are random numbers, $\{V_1, V_2, V_3, V_4, V_5, V_6\}$ are hash values, $\{T_1, T_2, T_3, T_4, T_5, T_6, T_7\}$ are timestamps, and $\{CN_m, CN_n\}$ are encrypted values. Through our calculations, the communication cost of Case 1 was 5152 bits. In Case 2, we only needed to calculate the communication cost of the four messages $\{M_1, M_2, M_3, M_4\}$ in the authentication phase. Through calculations, the communication cost of Case 2 was 2688 bits. From the above calculation method, the communication costs of the protocols of Ma et al. [9], Wazid et al. [29], Eftekhari et al. [10], and Wu et al. [11] were 4512, 3488, 4416, and 4448 bits, respectively. The comparison results are presented in Table 8. Therefore, evidently in Case 1, because our protocol has seven rounds of messages, the communication cost of our protocol was higher than that of the other protocols. In Case 2, the communication cost of our protocol was lower than that of the other protocols, which significantly reduced the communication cost of the protocol.

Table 8. Communication cost comparison.

Protocols	Rounds	Communication Cost
Ma et al. [9]	4	4512 bits
Wazid et al. [29]	3	3488 bits
Eftekhari et al. [10]	4	4416 bits
Wu et al. [11]	4	4448 bits
Ours, Case 1	7	5152 bits
Ours, Case 2	4	2688 bits

5.3. Discussions

Now, we discuss our protocol and those of Eftekhari et al. [10] and Wu et al. [11] in terms of architecture, computational cost, and communication cost.

In the first item (architecture), both protocols [10,11] used the traditional three-layer architecture (i.e., vehicle–fog node–CS). As mentioned in the Introduction, this architecture fails to realize the function of fog nodes sharing the computational burden of the CS, because the fog nodes actually replace the RSUs, and computation is still performed on the CS. To reduce the computational burden of the CS, we extended the traditional architecture to propose the first four-layer architecture, namely vehicle–RSU–fog node–CS. In this architecture, when a vehicle enters the road and wants to communicate with the RSU, the communication modes are divided into the two cases. Case 1 requires the participation of each entity, while Case 2 only requires the participation of the vehicle, RSU and fog node without the CS. Therefore, this architecture effectively realizes the function of the fog node sharing the computational burden of the CS.

In the second item (computational cost), both protocols [10,11] performed ECC operations. Our protocol performs hash, fuzzy extraction, and symmetric encryption and decryption operations in Case 1, and it only performs hash and fuzzy extraction operations in Case 2. Though the three protocols are secure, our protocol has a lower computational cost.

In the final item (communication cost), the communication cost of the proposed protocol (Case 1) was slightly higher than that of both protocols [10,11] because it requires an additional three rounds to judge whether the vehicle communicates with the RSU for the first time. However, Case 1 only occurred once for the same vehicle and RSU. In Case 2, our protocol had a lower communication cost than Eftekhari et al. [10] and Wu et al.'s [11] protocols.

6. Conclusions

In this paper, we first introduced the IoV and fog computing and reviewed the related research results. After that, we proposed an authentication protocol based on SGX and fog computing in the IoV. In this protocol, we first proposed a four-layer architecture that significantly reduced the computational burden of the CS. To resist several well-known attacks, we applied SGX to our protocol. Finally, we proved the security of the proposed protocol through the ROR model and informal analysis. We also compared its performance with those of recent protocols. The results show that the proposed protocol had better security and a lower computational cost. Future studies will continue to conduct further research based on the architecture used. We hope that this research will provide ideas and help researchers.

Author Contributions: Conceptualization, T.-Y.W.; innovation, X.G.; methodology, T.-Y.W. and X.G.; software, Y.-C.C.; formal analysis, S.K.; investigation, C.-M.C.; writing—original draft preparation, T.-Y.W., X.G., Y.-C.C., S.K. and C.-M.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: This study did not involve humans.

Data Availability Statement: The data are contained within the article.

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

Abbreviations

The following abbreviations are used in this manuscript:

IoT	Internet of Things
IoV	Internet of Vehicles
SGX	Software guard extensions
PRM	Preserved random memory
EPC	Enclave page cache
ROR	Real-or-Random
V2V	Vehicle-to-vehicle
V2P	Vehicles-to-pedestrians
V2I	Vehicle-to-infrastructure
AKA	Authenticated key agreement
RSU	Roadside unit
PKI	Public key infrastructure
ECC	Elliptic curve cryptography
AKM	Authentication key management
PUF	Physical unclonable function

References

1. Khan, M.A.; Salah, K. IoT security: Review, blockchain solutions, and open challenges. *Future Gener. Comput. Syst.* **2018**, *82*, 395–411. [CrossRef]
2. Chegini, H.; Naha, R.K.; Mahanti, A.; Thulasiraman, P. Process automation in an IoT–fog–cloud ecosystem: A survey and taxonomy. *IoT* **2021**, *2*, 92–118. [CrossRef]
3. Yang, F.; Wang, S.; Li, J.; Liu, Z.; Sun, Q. An overview of internet of vehicles. *China Commun.* **2014**, *11*, 1–15. [CrossRef]
4. Contreras-Castillo, J.; Zeadally, S.; Guerrero-Ibañez, J.A. Internet of vehicles: Architecture, protocols, and security. *IEEE Internet Things J.* **2017**, *5*, 3701–3709. [CrossRef]
5. Zhou, H.; Xu, W.; Chen, J.; Wang, W. Evolutionary V2X technologies toward the Internet of vehicles: Challenges and opportunities. *Proc. IEEE* **2020**, *108*, 308–323. [CrossRef]
6. Stojmenovic, I.; Wen, S.; Huang, X.; Luan, H. An overview of fog computing and its security issues. *Concurr. Comput. Pract. Exp.* **2016**, *28*, 2991–3005. [CrossRef]
7. Chen, S.; Zhang, T.; Shi, W. Fog computing. *IEEE Internet Comput.* **2017**, *21*, 4–6. [CrossRef]
8. Dastjerdi, A.V.; Gupta, H.; Calheiros, R.N.; Ghosh, S.K.; Buyya, R. Fog computing: Principles, architectures, and applications. In *Internet of things*; Elsevier: Amsterdam, The Netherlands, 2016; pp. 61–75. [CrossRef]
9. Ma, M.; He, D.; Wang, H.; Kumar, N.; Choo, K.K.R. An efficient and provably secure authenticated key agreement protocol for fog-based vehicular ad-hoc networks. *IEEE Internet Things J.* **2019**, *6*, 8065–8075. [CrossRef]
10. Eftekhari, S.A.; Nikooghadam, M.; Rafighi, M. Security-enhanced three-party pairwise secret key agreement protocol for fog-based vehicular ad-hoc communications. *Veh. Commun.* **2021**, *28*, 100306. [CrossRef]
11. Wu, T.Y.; Lee, Z.; Yang, L.; Luo, J.N.; Tso, R. Provably secure authentication key exchange scheme using fog nodes in vehicular ad hoc networks. *J. Supercomput.* **2021**, *77*, 6992–7020. [CrossRef]
12. Wu, T.Y.; Guo, X.; Yang, L.; Meng, Q.; Chen, C.M. A Lightweight Authenticated Key Agreement Protocol Using Fog Nodes in Social Internet of Vehicles. *Mob. Inf. Syst.* **2021**, *2021*, 3277113. [CrossRef]
13. Ying, B.; Nayak, A. Anonymous and lightweight authentication for secure vehicular networks. *IEEE Trans. Veh. Technol.* **2017**, *66*, 10626–10636. [CrossRef]
14. Mohit, P.; Amin, R.; Biswas, G. Design of authentication protocol for wireless sensor network-based smart vehicular system. *Veh. Commun.* **2017**, *9*, 64–71. [CrossRef]
15. Yu, S.; Lee, J.; Lee, K.; Park, K.; Park, Y. Secure authentication protocol for wireless sensor networks in vehicular communications. *Sensors* **2018**, *18*, 3191. [CrossRef] [PubMed]
16. Li, J.; Lu, H.; Guizani, M. ACPN: A novel authentication framework with conditional privacy-preservation and non-repudiation for VANETs. *IEEE Trans. Parallel Distrib. Syst.* **2014**, *26*, 938–948. [CrossRef]

17. Liu, X.; Guo, Z.; Ma, J.; Song, Y. A Secure Authentication Scheme for Wireless Sensor Networks Based on DAC and Intel SGX. *IEEE Internet Things J.* **2021**, *9*, 3533–3547. [CrossRef]
18. Condé, R.C.; Maziero, C.A.; Will, N.C. Using Intel SGX to protect authentication credentials in an untrusted operating system. In Proceedings of the 2018 IEEE Symposium on Computers and Communications (ISCC), Natal, Brazil, 25–28 June 2018; pp. 158–163.
19. Wang, J.; Hao, S.; Li, Y.; Fan, C.; Wang, J.; Han, L.; Hong, Z.; Hu, H. Challenges towards protecting vnf with sgx. In Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization, Tempe, AZ, USA, 21 March 2018; pp. 39–42. [CrossRef]
20. Chaudhry, S.A. Combating identity de-synchronization: An improved lightweight symmetric key based authentication scheme for IoV. *J. Netw. Intell.* **2021**, *6*, 12.
21. Xiong, H.; Chen, J.; Mei, Q.; Zhao, Y. Conditional privacy-preserving authentication protocol with dynamic membership updating for VANETs. *IEEE Trans. Dependable Secur. Comput.* **2022**, *19*, 2089–2104. [CrossRef]
22. Raya, M.; Hubaux, J.P. Securing vehicular ad hoc networks. *J. Comput. Secur.* **2007**, *15*, 39–68. [CrossRef]
23. Huang, J.L.; Yeh, L.Y.; Chien, H.Y. ABAKA: An anonymous batch authenticated and key agreement scheme for value-added services in vehicular ad hoc networks. *IEEE Trans. Veh. Technol.* **2010**, *60*, 248–262. [CrossRef]
24. Sadri, M.J.; Rajabzadeh Asaar, M. A lightweight anonymous two-factor authentication protocol for wireless sensor networks in Internet of Vehicles. *Int. J. Commun. Syst.* **2020**, *33*, e4511. [CrossRef]
25. Jiang, Q.; Zhang, X.; Zhang, N.; Tian, Y.; Ma, X.; Ma, J. Three-factor authentication protocol using physical unclonable function for IoV. *Comput. Commun.* **2021**, *173*, 45–55. [CrossRef]
26. Kumar, S.; Banka, H.; Kaushik, B.; Sharma, S. A review and analysis of secure and lightweight ECC-based RFID authentication protocol for Internet of Vehicles. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4354. [CrossRef]
27. Wu, T.Y.; Meng, Q.; Yang, L.; Guo, X.; Kumari, S. A provably secure lightweight authentication protocol in mobile edge computing environments. *J. Supercomput.* **2022**, 1–22. [CrossRef]
28. Huang, X.; Xiong, H.; Chen, J.; Yang, M. Efficient Revocable Storage Attribute-based Encryption with Arithmetic Span Programs in Cloud-assisted Internet of Things. *IEEE Trans. Cloud Comput.* **2021**. [CrossRef]
29. Wazid, M.; Bagga, P.; Das, A.K.; Shetty, S.; Rodrigues, J.J.; Park, Y. AKM-IoV: Authenticated key management protocol in fog computing-based Internet of vehicles deployment. *IEEE Internet Things J.* **2019**, *6*, 8804–8817. [CrossRef]
30. Han, M.; Liu, S.; Ma, S.; Wan, A. Anonymous-authentication scheme based on fog computing for VANET. *PLoS ONE* **2020**, *15*, e0228319. [CrossRef]
31. Soleymani, S.A.; Goudarzi, S.; Anisi, M.H.; Zareei, M.; Abdullah, A.H.; Kama, N. A security and privacy scheme based on node and message authentication and trust in fog-enabled VANET. *Veh. Commun.* **2021**, *29*, 100335. [CrossRef]
32. Canetti, R.; Goldreich, O.; Halevi, S. The random oracle methodology, revisited. *J. ACM (JACM)* **2004**, *51*, 557–594. [CrossRef]
33. Dolev, D.; Yao, A. On the security of public key protocols. *IEEE Trans. Inf. Theory* **1983**, *29*, 198–208. [CrossRef]
34. Canetti, R.; Krawczyk, H. Analysis of key-exchange protocols and their use for building secure channels. In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Innsbruck, Austria, 6–10 May 2001; Springer: Berlin/Heidelberg, Germany, 2001; pp. 453–474.
35. Wang, D.; Cheng, H.; Wang, P.; Huang, X.; Jian, G. Zipf's law in passwords. *IEEE Trans. Inf. Forensics Secur.* **2017**, *12*, 2776–2791. [CrossRef]
36. He, D.; Kumar, N.; Lee, J.H.; Sherratt, R.S. Enhanced three-factor security protocol for consumer USB mass storage devices. *IEEE Trans. Consum. Electron.* **2014**, *60*, 30–37.