Heterogeneous Signcryption Scheme with Group Equality Test for Satellite-enabled IoVs

Yingzhe Hou, Yue Cao, *Senior Member*, *IEEE*, Hu Xiong, *Senior Member*, *IEEE*, Yulin Hu, *Senior Member*, *IEEE*, Max Hashem Eiza

***Abstract*—With the growing popularization of the Internet of Vehicles (IoVs), the combination of satellite navigation system and IoVs is also in a state of continuous improvement. In this paper, we present a heterogeneous signcryption scheme with group equality test for IoVs (HSC-GET), which avoids the adversaries existing in the insecure channels to intercept, alter or delete messages from satellite to vehicles. The satellite is arranged in an identity-based cryptographic (IBC) system to ensure safe and fast transmission of instruction, while the vehicles are arranged in certificateless cryptosystem (CLC) to concern the security of the equipment. In addition, the group granularity authorization is integrated to ensure the cloud server can only execute the equality test on ciphertext generated by the same group of vehicles. Through rigorous performance and security analyses, we observe that our proposed construction reduces the equality test overhead by about 63.96%, 81.23%, 80.84%, and 54.98% in comparison to other competitive protocols. Furthermore, the confidentiality, integrity and authenticity of messages are guaranteed.**

***Index Terms*—Heterogeneous, Group Equality Test, Signcryption, Internet of Vehicles (IoVs).**

I. Introduction

THE Internet of Vehicles (IoVs) is regarded as the superset of Vehicular Ad-hoc Networks (VANETs) [1]. Numerous of previous protocols focus on the interaction of ground traffic information in the IoVs, to ensure the safe transmission of information between vehicles and infrastructures equipped on roadside [2], [3]. Due to the economic costs and technical factors, many areas with complex terrain are unable to provide ground network coverage through cellular base stations. Therefore, satellite communication is integrated into terrestrial 5G/6G mobile communication to complement coverage loss, which has the advantages of wide coverage, large communication capacity, and high flexibility. The satellite and ground-integrated IoVs concern the information transmission from a more comprehensive perspective, rather than within the ground view only, so as to broaden the applicability of IoVs [4].

Despite a lot of benefits are brought by IoVs, it is critical to guarantee the confidentiality of message transmission due to the insecure nature of the transmission channel [5]. Subsequently, the concept of encryption is introduced to preserve the confidentiality of messages [6]. A satellite navigation system with high-precision positioning and real-time communication can help vehicles answer the questions of "where am I, where am I going and how to reach there". For security navigation purpose, the satellite can encrypt the instruction messages and deliver the ciphertexts to vehicles. Then, the vehicles can back up these instruction ciphertexts to the cloud server for further utilization. However, this approach leads to a new problem of incompatibility between data utilization and security because it is hard to search the ciphertext. To address this problem, the concept of public key encryption supporting equality test (PKEET) is presented. Through this scheme, the search function is conducted via the cloud server, to determine whether two ciphertexts conclude the same message and improve the ciphertext search efficiency. Following this, plenty of schemes based on PKEET are presented [7]–[9].

Nevertheless, the aforementioned PKEET schemes are not suitable for scenarios with groups of users. As the Fig. 1 shown, there exist four vehicles V_A, V_B, V_C and V_D in IoVs, V_A and V_B belonging to group G_1. Similarly, V_C and V_D belonging to group G_2. Ordinarily, each vehicle uploads its trapdoor to cloud server such as TD_A, TD_B, TD_C and TD_D. If there exist n group users, that is to say, the tester requires to store n trapdoors generated by group users. Thus, the cloud server will inevitably execute the test operation on ciphertext from different groups. However, the storage and management of n trapdoors results in a lot of storage overhead. Therefore, a group granularity authorization for PKEET scene is required to provide. Inspired by this, Ling *et al.* [10] proposed a group public key encryption supporting equality test scheme (G-PKEET), which integrates the group granularity authorization. The trapdoor uploading method of group equality test in IoVs is elaborated in Fig. 2. In this manner, the cloud server just stores two group trapdoors from G_1 and G_2, and reduces the communication cost greatly.

Inevitably, only utilizing the above schemes is vulnerable to altering, intercepting or deleting the transmitted messages, due to lack of the signature primitive. In order to address this challenge, a signcryption protocol supporting equality test (PKS-DET) is formulated in [11]. In their construction, the function of signature and encryption can be implemented within one algorithm, and thus the communication and computing costs are reduced simultaneously. However, the scheme in [11] is only targeted at the traditional public key infrastructure (PKI) system. When the devices are equipped on heterogeneous

Y. Hou, Y. Cao are with the School of Cyber Science and Engineering, Wuhan University, Wuhan, 430000, China. (e-mail: hyzpax1@gmail.com, yue.cao@whu.edu.cn.).

H. Xiong is with the School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, 610054, China. (e-mail: xionghu.uestc@gmail.com.).

Y. Hu is with the School of Electronic Information, Wuhan University, Wuhan, 430072, China.(e-mail: yulin.hu@whu.edu.cn.).

M. H. Eiza is with the School of Computer Science and Mathematics Liverpool John Moores University (LJMU) Liverpool, United Kingdom.(e-mail: M.HashemEiza@ljmu.ac.uk.).

Corresponding author: yue.cao@whu.edu.cn.

cryptosystems, it still needs more research effort to solve. Envisioning for satellite-enabled IoVs, the satellite is deployed in the identity-based cryptography (IBC) to ensure the rapid transmission of instruction messages. Besides, the ground vehicles are allocated in the certificateless cryptography (CLC) to guarantee a higher security of vehicle. Therefore, how to construct an appropriate solution for information exchange, and support the efficient group retrieval function is a crucial issue.

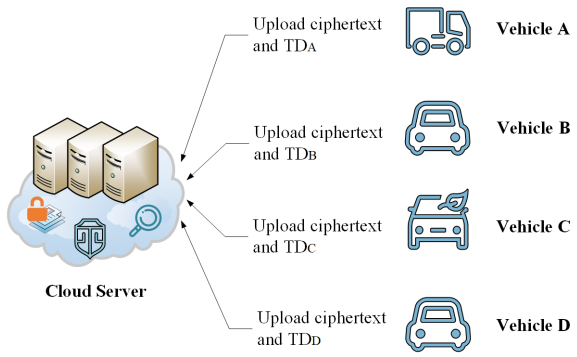


Fig. 1: The trapdoor uploaded way of equality test

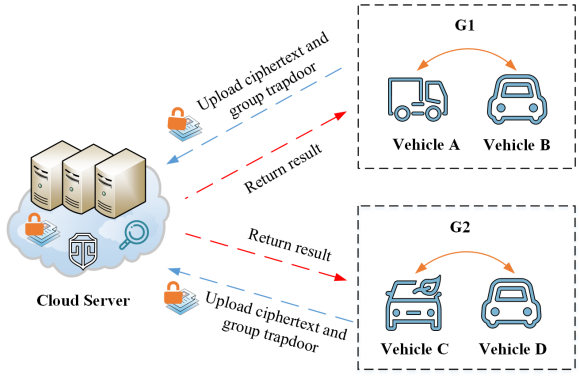


Fig. 2: The trapdoor uploaded way of group equality test

Considering the aforementioned challenges, a heterogeneous signcryption protocol supporting group equality test for IoVs (HSC-GET) is proposed. The purpose is to achieve the information interaction between heterogeneous devices under the umbrella of satellite-generated IoVs. Moreover, it also provides the ciphertext searchable with a group granularity authorization. The HSC-GET scheme can reach the security of indistinguishability against adaptive chosen-ciphertext attacks (IND-CCA), one-way under chosen-ciphertext attacks (OW-CCA), and existential unforgeability against adaptive chosen messages attacks (EUF-CMA). Furthermore, the concrete contributions are demonstrated as follows:

- Compared to literature of applying equality test only for single group communication, the group granularity authorization is integrated, it can guarantee the cloud server to conduct the equality test on ciphertext generated from the same group, which reduces the storage overhead of trapdoor and the computing cost greatly. Besides, the proposed HSC-GET can resist the offline message recovery attack (OMRA), that is to say, during the actual polynomial message space, the cloud server cannot recover the plaintext message from a specified ciphertext.
- In addition, compared to literature of applying equality test only in heterogeneous networks, the complementary feature between heterogeneous IoVs and signcryption is considered in proposed scheme. According to the different security requirements of different devices, the satellite and vehicles are deployed on IBC and CLC system, respectively. The proposed HSC-GET provides a solution to communication from IBC to CLC system. The confidentiality, unforgeability, and integrity of information are all ensured in heterogeneous system simultaneously.

The related work is shown in section II. The relevant basic preliminaries are given in section III. During section IV, we elaborate the specific HSC-GET construction. During section V, we prove the security of HSC-GET via five theorems. During section VI, we analyze the performance comparison of schemes. During section VII, we summarize the conclusion of this paper.

II. Related works

A. Signcryption

The first primitive of signcryption is introduced by Zheng [12]. This manner can realize all the security targets of signature and encryption during a concrete operation. In literature [12], the authors also constructed a concrete signcryption protocol, and the security of this construction is proven by a discrete logarithm problem. However, the signcryption security notion is failed to formalize. An *et al.* [13] studied the related properties of signcryption systematically. Then, a series of schemes based on signcryption have been proposed [14]–[16]. Nonetheless, the certificate management problem exists in signcryption scheme under PKI mechanism. Therefore, the scheme of identity-based signcryption (IBSC) is proposed in [17], in which the identity information can be seen as the public key. Then, there have more details of IBSC are defined in [18]–[20]. The first certificateless signcryption (CLSC) protocol is formulated [21]. In this protocol, the certificateless mechanism is introduced to avoid the inherent problem in PKI and key escrow problem in IBC. Following this idea, the research based on CLSC is increasing gradually [22], [23]. With the in-depth understanding of IoVs, different devices may be deployed in different cryptographic systems, so how to build a way of information transmission for heterogeneous devices is very important. Li *et al.* [24] introduced two heterogeneous signcryption methods, to provide a two-way between PKI and IBC devices transmission.

B. Equality test

Yang *et al.* [25] constructed the first public key encryption protocol supporting equality test (PKEET), their scheme allows examine the ciphertext from different public keys. Nonetheless, the authorization mechanism is not discussed in [25]. Then, the fine-grained authorization policy is proposed based on PKEET (FG-PKEET) [26]. Following this, Tang

[27] formulated a two-proxy mechanism collaborating with each other to perform the equality test. Besides, the all-or-nothing PKEET is also introduced to specify who has the ability to conduct the equality test [28]. An identity-based encryption protocol supporting the equality test operation (IBEET) is introduced [29]. Through this manner, the certificate management problem is resolved. However, the key escrow defect cannot be ignored in IBC system. To solve this weakness, Qu *et al.* [30] presented a certificateless public key encryption protocol with equality test (CL-PKEET), which addresses the inherent defect during above schemes. Combined with the signcryption primitive, Wang *et al.* [11] suggested the public key signcryption protocol with designated equality test on ciphertexts. Xiong *et al.* [31] integrated the identity-based signcryption scheme with equality test. In order to adapt to heterogeneous environment, a series of heterogeneous signcryption protocols supporting equality test are researched recently [32]–[34].

C. Resistance Against OMRA

There exist two main reasons for OMRA during the schemes based on equality test [35]. The first reason is independent testing, this condition allows the tester executes the equality test independently. Targeted to this challenge, two servers are employed to resist OMRA since that two testers require to collaborate with each other to perform equality test, and this cracking the function of independence [27], [36]–[38]. The second reason is that the tester has the ability to produce ciphertext for any guessed message. For addressing this problem, Wu *et al.* [39] defined a novel security model of IBEET, their scheme enables the attacker to perform the equality test on all ciphertexts. Nonetheless, the attacker does not have the function to output ciphertexts. Then, Ling *et al.* [40] proposed a group ID-based encryption with equality test (G-IBEET), which resists the insider attack via the group mechanism.

III. Preliminary

We describe the basic mathematical knowledge, the protocol definition, the security and system model in this section. The symbols involved in this scheme and their specific meanings are elaborated in Table I.

A. Bilinear Map

Given two cyclic groups G_1 and G_2. Define q, P are the prime order and the generator. The bilinear map $e : G_1 \times G_1 \rightarrow G_2$ is set with the properties as below:

- Bilinearity: $\forall b, d \in Z_p^*$, $\forall B, D \in G_1$, $e(bB, dD) = e(B, D)^{bd}$.
- Non-degeneracy: $\exists B, D \in G_1$ meets $e(B, D) \neq 1$.

B. Mathematical Problem

Bilinear Diffie-Hellman Problem (BDHP): Inputting an item (g, g^a, g^b, g^c), the target of BDHP is to determine the equation $V \stackrel{?}{=} e(g, g)^{abc}$ holds or not, in which $a, b, c \in Z_p^*$, $V \in G_2$.

TABLE I: Notions and symbols

Notations	Explanation
G_1, G_2	Two multiplication group
g, U	Two generator of G_1
$H_i (0 \leq i \leq 4)$	Five hash functions
k	The security parameter
s	The system secret key
PP	The system parameter
P_{pub}	The system public key
ID_s	The identity of satellite
sk_s	The private key of satellite
ID_r	The identity of vehicle
D_r	The partial private key of vehicle
sk_r	The private key of vehicle
PK_r	The public key of vehicle
gsk	The group secret key
gpk_r	The group public key
C	The signcryption ciphertext
gtd	The group trapdoor
T_{bp}	The bilinear pairing operation
T_{hg}	The hash to G_1 operation
T_{e1}	The exponentiation operation in G_1
T_{e2}	The exponentiation operation in G_2
T_{cr}	The cost of certificate management

Bilinear Diffie-Hellman Assumption (BDHA): Suppose this assumption establishes, it refers that there is no probability polynomial time (PPT) adversary $\mathcal{A}$ can break the BDHP with a non-negligible advantage.

Computational Diffie-Hellman Problem (CDHP): Inputting an item $(g, g^a, g^b) \in G_1$, the target of CDHP is to compute the value of g^{ab}, where $a, b \in Z_p^*$.

Computational Diffie-Hellman Assumption (BDHA): Suppose this assumption establishes, there is no PPT adversary $\mathcal{A}$ can break the CDHP with a non-negligible advantage.

C. Scheme Definition

- Setup: Upon obtaining k, this step is performed by KGC to output s and P_{pub}. Then, it publics the public parameter PP.
- IBC-KG: Upon obtaining the sender's identity ID_s, KGC outputs the private key sk_s to the required sender.
- CLC-KG: The following steps are mainly included in the CLC system.
  - Partial private key: Upon obtaining the receiver's identity ID_r, KGC generates the partial private key D_r to the required receiver.
  - Private Key: Upon receiving D_r, the receiver chooses a secret key x, then calculates its private key sk_r.
  - Public Key: Upon receiving ID_r, the receiver generates its own public key PK_r.
- KG-Group: This operation is executed by TA to output the group secret key gsk for group receiver.
- Join-Group: This operation is executed by TA to generate the group public key gpk_r for group receiver.
- Signcryption: Upon receiving the plaintext M, the group public key gpk_r, the sender's private key sk_s, and a group receiver's public key PK_r, the sender performs this step to output C.
- Unsigncryption: Upon obtaining C, a group public key gpk_r, the sender's public key ID_s, and the receiver's

private key sk_r, this operation is run by a receiver to output the plaintext M.
- Group-TD: Upon receiving the group secret key gsk, TA generates the group trapdoor gtd.
- Test: Upon receiving two ciphertexts from two receivers C_A, C_B and the group trapdoor gtd, this operation is executed by the cloud server and returned 1 or 0.

D. Security Model

During this section, the security model of HSC-GET is elaborated via five games.

Game 1: $\mathcal{A}_1$ and a challenger $\mathcal{C}$ perform the operations as below.

Setup: $\mathcal{C}$ executes the Setup step and outputs the public parameter PP to $\mathcal{A}_1$. In addition, $\mathcal{C}$ performs PKI-KG to produce sender's key pair (sk_s^*, ID_s^*) to $\mathcal{A}_1$.

Phase 1: The queries are issued by $\mathcal{A}_1$ as below.

- Public key query: Upon obtaining an identity of receiver ID_r, the public key generation algorithm is conducted by $\mathcal{C}$ and the outcome PK_r is returned to $\mathcal{A}_1$.
- Partial private key query: Given ID_r as input, $\mathcal{C}$ delivers D_r to $\mathcal{A}_1$.
- Private key query: Given ID_r as input, $\mathcal{C}$ performs the private key generation step and delivers sk_r to $\mathcal{A}_1$.
- Replace public key query: Upon obtaining this query, $\mathcal{C}$ replaces the current public key PK_r with PK_r'.
- Unsigncryption query: Upon obtaining the query, the outcome of Unsigncryption (C, sk_r, ID_s^*) is returned to $\mathcal{A}_1$.

Challenge: $\mathcal{A}_1$ picks two messages M_0 and M_1 and the challenge identity of receiver ID_r^* to $\mathcal{C}$, where two messages have the same length. Then, $\mathcal{C}$ chooses $\rho \in \{0,1\}$ and sends the outcome of $C^* = \text{Signcryption}(M_\rho, sk_s^*, PK_r^*)$ to $\mathcal{A}_1$.

Phase 2: The similar queries are executed by $\mathcal{A}_1$ as *Phase 1*. Nonetheless, $\mathcal{A}_1$ has the following restrictions:

- $\mathcal{A}_1$ cannot perform the Partial private key query, Private key query on ID_r^*.
- If the public key has been replaced, $\mathcal{A}_1$ cannot execute the Private key query on corresponding ID_r.
- The Unsigncryption query on (C^*, sk_r^*, ID_s^*) also cannot be issued by $\mathcal{A}_1$.

Guess: $\mathcal{A}_1$ generates $\rho' \in \{0,1\}$. Suppose $\rho' = \rho$, this game wins.

Definition 1: The HSC-GET scheme can achieve the IND-CCA security, if there is no adversary $\mathcal{A}_1$ with a non-negligible advantage during the above **Game 1**.

Game 2: $\mathcal{A}_2$ and a challenger $\mathcal{C}$ perform the operations as below.

Setup: $\mathcal{C}$ runs the Setup step and produces the public parameter PP, the master secret key s to $\mathcal{A}_2$. In addition, $\mathcal{C}$ performs PKI-KG to produce sender's key pair (sk_s^*, ID_s^*) to $\mathcal{A}_2$.

Phase 1: The following queries are issued by $\mathcal{A}_2$.

- Public key query: Upon obtaining the identity of receiver ID_r, the public key generation step is conducted by $\mathcal{C}$ and the outcome PK_r is returned to $\mathcal{A}_2$.
- Private key query: Given ID_r as input, $\mathcal{C}$ conducts the private key generation step and regards sk_r as a response.
- Unsigncryption query: Upon obtaining a ciphertext C, the outcome of Unsigncryption (C, sk_r, ID_s^*) is returned to $\mathcal{A}_2$.

Challenge: $\mathcal{A}_2$ picks two messages M_0 and M_1 and the challenge identity of receiver ID_r^* to $\mathcal{C}$, where two messages have the same length. Then, $\mathcal{C}$ chooses $\rho \in \{0,1\}$ and sends the outcome of $C^* = \text{Signcryption}(M_\rho, sk_s^*, PK_r^*)$ to $\mathcal{A}_2$.

Phase 2: The similar queries are executed by $\mathcal{A}_2$ as *Phase 1*. Nonetheless, $\mathcal{A}_2$ cannot execute the Private key query on ID_r^*. Besides, the Unsigncryption query on (C^*, sk_r^*, ID_s^*) also cannot be issued by $\mathcal{A}_2$.

Guess: $\mathcal{A}_2$ generates $\rho' \in \{0,1\}$. Suppose $\rho' = \rho$, this game wins.

Definition 2: The HSC-GET scheme can achieve the IND-CCA security, if there is no adversary $\mathcal{A}_2$ with a non-negligible advantage during the above **Game 2**.

Game 3: $\mathcal{C}$ and the adversary $\mathcal{A}_3$ execute the following operations.

Setup: $\mathcal{C}$ conducts the Setup step and outputs the public parameter PP to $\mathcal{A}_3$. In addition, $\mathcal{C}$ performs PKI-KG to produce sender's key pair (sk_s^*, ID_s^*) to $\mathcal{A}_3$.

Phase 1: The following queries are issued by $\mathcal{A}_3$.

- Public key query: Upon obtaining an identity of receiver ID_r, the public key algorithm is conducted by $\mathcal{C}$ to return the outcome PK_r.
- Partial private key query: Given ID_r as input, $\mathcal{C}$ executes the partial private key generation step and delivers D_r as a response.
- Private key query: Given ID_r as input, $\mathcal{C}$ returns sk_r as a response.
- Replace public key query: Upon obtaining this query, $\mathcal{C}$ replaces the current PK_r with PK_r'.
- Unsigncryption query: Upon obtaining the issue, the outcome of Unsigncryption (C, sk_r, ID_s^*) is returned to $\mathcal{A}_3$.
- Trapdoor query: Upon obtaining this query, $\mathcal{C}$ runs the Group-TD step and returns gtd to $\mathcal{A}_3$.

Challenge: $\mathcal{A}_3$ sends the message M^* and the challenge identity of receiver ID_r^* to $\mathcal{C}$. Then, $\mathcal{C}$ returns the outcome of $C^* = \text{Signcryption}(M^*, sk_s^*, PK_r^*)$ to $\mathcal{A}_3$.

Phase 2: The similar queries are executed by $\mathcal{A}_3$ as *Phase 1*. Nonetheless, $\mathcal{A}_3$ has the following restrictions:

- $\mathcal{A}_3$ cannot perform the Partial private key query, Private key query on ID_r^*.
- If the public key has been replaced, $\mathcal{A}_3$ cannot execute the Private key query on corresponding ID_r.
- The Unsigncryption query on (C^*, sk_r^*, ID_s^*) also cannot be issued by $\mathcal{A}_3$.

Guess: $\mathcal{A}_3$ generates M'. Suppose $M' = M^*$, $\mathcal{A}_3$ wins this game.

Definition 3: The HSC-GET scheme can achieve the OW-CCA security, if there is no adversary $\mathcal{A}_3$ with a non-negligible advantage during the above **Game 3**.

Game 4: $\mathcal{C}$ and the adversary $\mathcal{A}_4$ execute the operations as below.

Setup: $\mathcal{C}$ runs the Setup step and produces the public parameter PP , the master secret key s to $\mathcal{A}_4$. In addition, $\mathcal{C}$ performs PKI-KG to produce sender's key pair (sk_s^*, ID_s^*) to $\mathcal{A}_4$.

Phase 1: The queries are issued by $\mathcal{A}_4$ as below.

- Public key query: Upon obtaining the identity of receiver ID_r, the public key generation algorithm is conducted by $\mathcal{C}$ to return the outcome PK_r.
- Private key query: Given ID_r as input, $\mathcal{C}$ returns sk_r to $\mathcal{A}_4$.
- Unsigncryption query: Upon obtaining this query, the outcome of Unsigncryption (C, sk_r, ID_s^*) is returned to $\mathcal{A}_4$.
- Trapdoor query: Upon obtaining this query, $\mathcal{C}$ runs the Group-TD step and returns gtd to $\mathcal{A}_4$.

Challenge: $\mathcal{A}_4$ sends the message M^* and the challenge receiver's identity ID_r^* to $\mathcal{C}$. Then, the outcome of $C^* =$ Signcryption(M^*, sk_s^*, PK_r^*) is delivered to $\mathcal{A}_4$.

Phase 2: $\mathcal{A}_4$ conducts the similar queries as *Phase 1*. Nonetheless, $\mathcal{A}_4$ cannot execute the Private key query on ID_r^*. Besides, the Unsigncryption query on (C^*, sk_r^*, ID_s^*) also cannot be issued by $\mathcal{A}_4$.

Guess: $\mathcal{A}_4$ generates M'. Suppose $M' = M^*$, $\mathcal{A}_4$ wins this game.

Definition 4: The HSC-GET scheme can achieve the OW-CCA security, if there is no adversary $\mathcal{A}_4$ with a non-negligible advantage during the above **Game 4**.

Game 5: $\mathcal{C}$ and the adversary $\mathcal{A}_5$ execute the operations as follows.

Setup: $\mathcal{C}$ conducts the Setup step and delivers the system parameters PP to $\mathcal{A}_5$. Then, the IBC-KG algorithm is conducted by $\mathcal{C}$, the receiver's key pair (sk_r^*, PK_r^*) is delivered to $\mathcal{A}_5$.

Queries: The following queries are issued by $\mathcal{A}_5$.

- Private key query: Given ID_s as input, $\mathcal{C}$ conducts the IBC-KG operation to output sk_s to $\mathcal{A}_5$.
- Signcryption query: Upon obtaining this query, $\mathcal{C}$ delivers the outcome of Signcryption (M, sk_s, PK_r^*) to $\mathcal{A}_5$.

Forgery: $\mathcal{A}_5$ generates a valid challenge ciphertext $(M^*, sk_s^*, PK_r^*, C^*)$ to $\mathcal{C}$. Nonetheless, $\mathcal{A}_5$ cannot perform the Signcryption query on (M^*, sk_s^*, PK_r^*).

Definition 5: The EUF-CMA security is achieved in HSC-GET, if there is no adversary $\mathcal{A}_5$ with a non-negligible advantage during the above **Game 5**.

E. System Model

In the proposed scheme, as the Fig. 3 elaborated, there exist four entities during this system model. The specific functions are shown as follows:

- Satellite: It is allocated on the IBC system, then it signcrypts the message to required vehicles.
- Vehicle: It is allocated on the CLC system, then it recovers the cipheretxt to plaintext message.
- Trusted authorization (TA): It is in charge of generating the group public key and group trapdoor for group vehicles.
- Cloud server: It is responsible for executing the equality test on different ciphertexts from group vehicles.

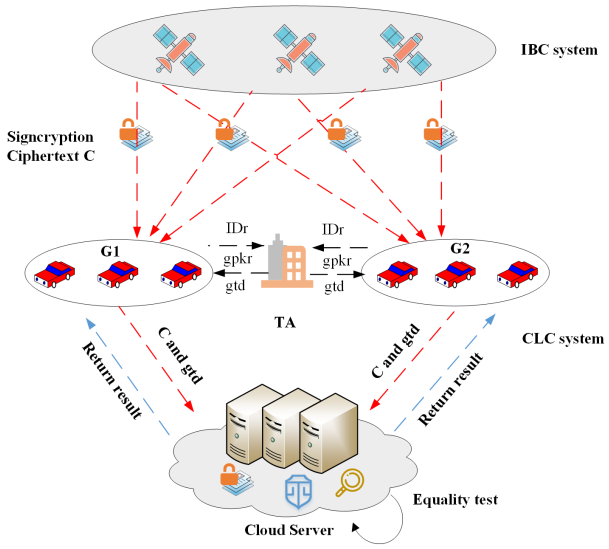


Fig. 3: The system model of HSC-GET

IV. Construction

The construction of proposed HSC-GET is illustrated as below:

1) Setup: With the security parameter k as input, KGC generates the multiplicative groups G_1 and G_2, in which p is the prime order, g and U are two generators of G_1. The bilinear map is defined $G_1 \times G_1 \to G_2$. In addition, it selects five hash functions: $H_0 : \{0,1\}^* \to G_1$, $H_1 : \{0,1\}^* \to G_1$, $H_2 : \{0,1\}^n \to G_1$, $H_3 : G_2 \to \{0,1\}^{l_1+l_2}$, $H_4 : \{0,1\}^n \times G_1^3 \times \{0,1\}^{l_1+l_2} \times Z_p^* \to Z_p^*$, in which l_1 and l_2 represents the length of message and Z_p^*. Then, KGC selects s as the master secret key and computes the system public key $P_{pub} = g^s$. Finally, it publics the parameter $PP = \{e, p, G_1, G_2, g, U, P_{pub}, H_0, H_1, H_2, H_3, H_4\}$.
2) IBC-KG: Upon receiving the identity of sender (satellite) ID_s, this operation is generated by KGC, it computes $Q_s = H_0(ID_s)$, and calculates $sk_s = Q_s^s$ as the private key. The corresponding system model is displayed in Fig. 4.

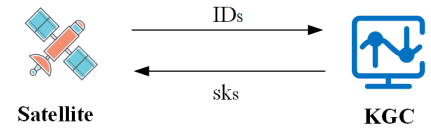


Fig. 4: The interaction model of IBC-KG

3) CLC-KG: Given the identity ID_r of receiver (vehicle) as input, the following operation is performed. The corresponding interaction model is shown in Fig. 5.
   - Partial private key generation: KGC computes $Q_r = H_1(ID_r)$, and calculates the partial private key $D_r = Q_r^s$.

- Private key generation: A receiver chooses $x \in Z_p^*$, then calculates $sk_r = D_r^x$ as the private key.
- Public key generation: The receiver computes $PK_r = P_{pub}^x$ as the public key.

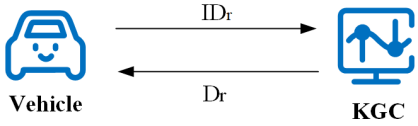


Fig. 5: The interaction model of CLC-KG

4) KG-Group: The step randomly selects $s_1, s_2 \in Z_p^*$ and sets $gsk = (s_1, s_2)$ as the group secret key.
5) Join-Group: Given gsk and ID_r as input, this operation computes $Q_r = H_1(ID_r)$ and defines $gpk_r = (s_1, Q_r^{s_2})$ as the group public key for group user ID_r. The corresponding interaction model is elaborated in Fig. 6.

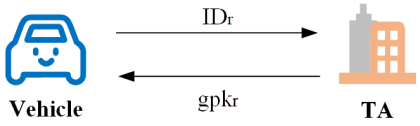


Fig. 6: The interaction model of Join-Group vehicles

6) Signcryption: Inputting M, the group public key gpk_r, the private key of sender sk_s, and the public key of a group receiver PK_r. The corresponding interaction model is elaborated in Fig. 7. This algorithm executes the following operations:
- Select $r_1, r_2 \in Z_p^*$ and calculate $Q_r = H_1(ID_r)$.
- Calculate $C_1 = g^{r_1}$.
- Calculate $C_2 = Q_r^{r_2 s_2} \cdot H_2(M)^{s_1}$.
- Calculate $C_3 = Q_r^{r_2}$.
- Calculate $W = e(PK_r, Q_r)^{r_1}$.
- Calculate $C_4 = H_3(W) \oplus (M||r_2)$.
- Calculate $h = H_4(M||C_1||C_2||C_3||C_4||r_2)$.
- Calculate $C_5 = U^{r_1} \cdot sk_s^h$.
- Output the ciphertext $C = (C_1, C_2, C_3, C_4, C_5)$. Finally, a sender delivers C to the group receiver.

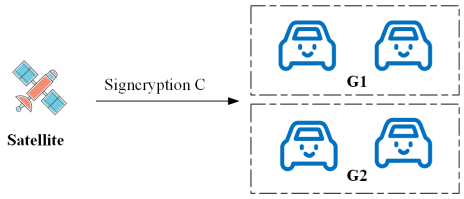


Fig. 7: The interaction model of Signcryption

7) Unsigncryption: Inputting a group public key gpk_r, C, the group receiver's private key sk_r, and the sender's public key ID_s. The corresponding interaction model is elaborated in Fig. 8. A received vehicle executes the following operations:
- Calculate $Q_s = H_0(ID_s)$.
- Calculate $W = e(C_1, sk_r)$.
- Compute $M'||r_2' = C_4 \oplus H_3(W)$.
- Compute $h' = H_4(M'||C_1||C_2||C_3||C_4||r_2')$.
- If the following equations $C_3 \stackrel{?}{=} Q_r^{r_2'}$, $e(C_5, g) \stackrel{?}{=} e(C_1, U)e(Q_s^{h'}, P_{pub})$ both hold, the receiver returns the message M'; Else, it returns $\perp$.

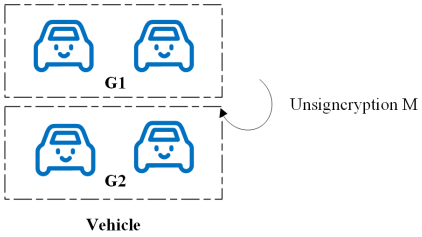


Fig. 8: The interaction model of Unsigncryption

8) Group-TD: Inputting the group secret key gsk, this operation randomly picks $y \in Z_p^*$. Then it calculates $gtd = (td_1, td_2) = (g^y, g^{s_2 y})$ as the group trapdoor. The corresponding interaction model is elaborated in Fig. 9.

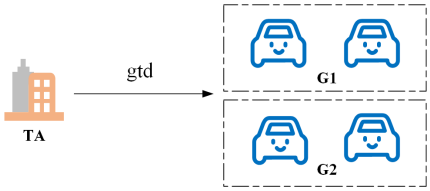


Fig. 9: The interaction model of generating group trapdoor

9) Test: Given two ciphertexts from two receivers $C_A = (C_{A,1}, C_{A,2}, C_{A,3}, C_{A,4}, C_{A,5})$, $C_B = (C_{B,1}, C_{B,2}, C_{B,3}, C_{B,4}, C_{B,5})$ and the group trapdoor $gtd = (td_1, td_2)$. The cloud server checks if the follow equation holds or not. The corresponding interaction model is shown in Fig. 10.

$$e(\frac{C_{A,2}}{C_{B,2}}, td_1) \stackrel{?}{=} e(\frac{C_{A,3}}{C_{B,3}}, td_2).$$

If the above equation holds, return 1; Else, return 0.

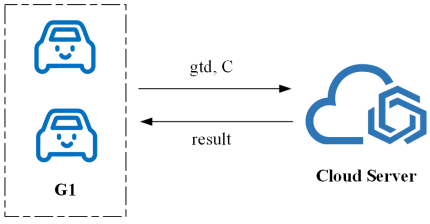


Fig. 10: The interaction model of equality test

Correctness: In the unsigncryption phase, the correctness is demonstrated as below:

$$\begin{aligned} W &= e(C_1, sk_r) \\ &= e(g^{r_1}, D_r^x) \\ &= e(g^{r_1}, Q_r^{sx}) \\ &= e(g^{sx}, Q_r)^{r_1} \\ &= e(PK_r, Q_r)^{r_1} \\ e(C_5, g) &= e(U^{r_1} \cdot sk_s^h, g) \\ &= e(U^{r_1}, g)e(sk_s^h, g) \\ &= e(U, g^{r_1})e(Q_s^h, g^s) \\ &= e(U, g^{r_1})e(Q_s^h, P_{pub}) \\ &= e(g^{r_1}, U)e(Q_s^h, P_{pub}) \\ &= e(C_1, U)e(Q_s^h, P_{pub}) \end{aligned}$$

In the test phase. If $M_A = M_B$, the following equation will hold. If $M_A \neq M_B$, it will not hold.

$$\begin{aligned} e(\frac{C_{A,2}}{C_{B,2}}, td_1) &= e(\frac{Q_A^{r_{A,2}s_2} \cdot H_2(M_A)^{s_1}}{Q_B^{r_{B,2}s_2} \cdot H_2(M_B)^{s_1}}, td_1) \\ &= e(\frac{Q_A^{r_{A,2}s_2} \cdot H_2(M_A)^{s_1}}{Q_B^{r_{B,2}s_2} \cdot H_2(M_B)^{s_1}}, g^y) \\ &= e(\frac{Q_A^{r_{A,2}} \cdot H_2(M_A)^{s_1}}{Q_B^{r_{B,2}} \cdot H_2(M_B)^{s_1}}, g^{s_2 y}) \\ &= e(\frac{C_{A,3} \cdot H_2(M_A)^{s_1}}{C_{B,3} \cdot H_2(M_B)^{s_1}}, td_2) \\ &= e(\frac{C_{A,3}}{C_{B,3}}, td_2) \end{aligned}$$

V. Security Analysis

In the security analysis section, we elaborate the concrete proof process, the proposed HSC-GET has the ability to achieve IND-CCA in (**Theorem 1**, **Theorem 2**), OW-CCA in (**Theorem 3**, **Theorem 4**) and EUF-CMA security in **Theorem 5**.

Theorem 1: Assuming an adversary $\mathcal{A}_1$ can break the security of IND-CCA, that is to say, it has a challenger $\mathcal{C}$ can address the BDHP with a non-negligible advantage. $\mathcal{A}_1$ cannot access the master secret key s. Nonetheless, it can replace the public key of receiver.

Proof: Inputting an item (g, g^a, g^b, g^c), the target of $\mathcal{C}$ is to generate $e(g, g)^{abc}$. Then, $\mathcal{A}_1$ and $\mathcal{C}$ jointly perform the operations as below.

Setup: The Setup step is firstly conducted by $\mathcal{C}$. Then, $\mathcal{C}$ sets $P_{pub} = g^a$ and delivers the system parameters to $\mathcal{A}_1$. Besides, the PKI-KG operation is performed to generate the sender's key pair (sk_s^*, ID_s^*) to $\mathcal{A}_1$.

Phase 1: $\mathcal{A}_1$ makes the following queries, $\mathcal{C}$ preserves five lists L_0, L_1, L_2, L_3, L_4.

- H_0 query: When executing the query, $\mathcal{C}$ firstly judges whether this query on ID_s was performed. If it has been executed, the previous result is returned. Else, $\mathcal{C}$ picks $h_0 \in G_1$ and inserts (ID_s, h_0) into L_0. Finally, $\mathcal{C}$ delivers h_0 to $\mathcal{A}_1$.
- H_1 query: When executing the query, $\mathcal{C}$ firstly judges whether this query on ID_r was performed. If it has been executed, the previous result is returned. Else, the following operations are performed.
  - If $ID_r \neq ID_r^*$, $\mathcal{C}$ picks $v \in Z_p^*$ and computes $h_1 = g^v$, then it inserts (ID_r, h_1) into L_1. Finally, $\mathcal{C}$ delivers h_1 to $\mathcal{A}_1$.
  - If $ID_r = ID_r^*$, $\mathcal{C}$ sets $h_1^* = g^b$ and inserts (ID_r^*, h_1^*) into L_1. Finally, $\mathcal{C}$ delivers h_1^* to $\mathcal{A}_1$.
- H_2 query: When executing the query, $\mathcal{C}$ firstly judges whether this query on M was performed. If it has been executed, the previous result is returned. Else, $\mathcal{C}$ picks $h_2 \in G_1$ and inserts (M, h_2) into L_2. Finally, $\mathcal{C}$ delivers h_2 to $\mathcal{A}_1$.
- H_3 query: When executing the query, $\mathcal{C}$ firstly judges whether this query on W was performed. If it has been executed, the previous result is returned. Else, $\mathcal{C}$ picks $h_3 \in \{0,1\}^{l_1+l_2}$ and inserts (W, h_3) into L_3. Then, h_3 is returned to $\mathcal{A}_1$.
- H_4 query: When executing the query, $\mathcal{C}$ firstly judges whether this query on $(M, C_1, C_2, C_3, C_4, r_2)$ was performed. If it has been executed, the previous result is returned. Else, $\mathcal{C}$ picks $h_4 \in Z_p^*$ and inserts $(M, C_1, C_2, C_3, C_4, r_2, h_4)$ into L_4. Finally, $\mathcal{C}$ delivers h_4 to $\mathcal{A}_1$.
- Public key query: Upon executing the query on ID_r, $\mathcal{C}$ firstly judges whether this query on ID_r was performed. If it has been executed, the previous result is returned. Else, the following operations are performed.
  - If $ID_r \neq ID_r^*$, $\mathcal{C}$ picks $x \in Z_p^*$ and computes $PK_r = P_{pub}^x$. Then, it inserts (ID_r, PK_r) into L_{pk} and delivers PK_r to $\mathcal{A}_1$.
  - If $ID_r = ID_r^*$, $\mathcal{C}$ computes $PK_r = g^{ax}$. Finally, $\mathcal{C}$ inserts (ID_r^*, PK_r^*) into L_{pk}. Finally, $\mathcal{C}$ delivers PK_r^* to $\mathcal{A}_1$.
- Partial private key query: Upon executing the query on ID_r, $\mathcal{C}$ firstly judges whether this query was performed. If it has been executed, the previous result is returned. Else, the following operations are performed.
  - If $ID_r \neq ID_r^*$, $\mathcal{C}$ computes $D_r = g^{av}$ and $sk_r = D_r^x = g^{avx}$. Then, it inserts $(ID_r, h_1, D_r, x, sk_r)$ into L_{sk}. Finally, $\mathcal{C}$ delivers D_r to $\mathcal{A}_1$.
  - If $ID_r = ID_r^*$, $\mathcal{C}$ aborts.
- Private key query: Upon executing the query on ID_r, the following operations are performed.
  - If $ID_r \neq ID_r^*$, $\mathcal{C}$ searches $(ID_r, h_1, D_r, x, sk_r)$ from L_{sk}. Finally, $\mathcal{C}$ delivers sk_r to $\mathcal{A}_1$.
  - If $ID_r = ID_r^*$, $\mathcal{C}$ aborts.
- Replace public key query: Given PK_r', the public key PK_r is replaced with PK_r'. Then $\mathcal{C}$ inserts (ID_r', PK_r') into L_{pk}.
- Unsigncryption query: When receiving the ciphertext $C = (C_1, C_2, C_3, C_4, C_5)$, $\mathcal{C}$ firstly executes Public key query and Private key query on ID_r. Besides, $\mathcal{C}$ continues

as below.

- If $ID_r \neq ID_r^*$, $\mathcal{C}$ searches $(ID_r, h_1, D_r, x, sk_r)$ from L_{sk}. Then, $\mathcal{C}$ delivers the outcome of Unsigncryption (C, sk_r, ID_s^*) to $\mathcal{A}_1$.
- If $ID_r = ID_r^*$, $\mathcal{C}$ searches the tuples (ID_s, h_0), (ID_r^*, h_1^*) and (W, h_3) from L_0, L_1 and L_3 correspondingly, $\mathcal{C}$ continues the following steps:
  1) Calculate $C_4 \oplus h_3 = M'||r_2'$.
  2) Compute $h' = H_4(M'||C_1||C_2||C_3||C_4||r_2')$.
  3) Determine whether the following equations $C_3 \stackrel{?}{=} h_1^{*r_2'}$ and $e(C_5, g) \stackrel{?}{=} e(C_1, U)e(h_0^{h'}, P_{pub})$ hold or not. If hold, $\mathcal{C}$ delivers M' to $\mathcal{A}_1$.

  If the tuples do not exist in L_0, L_1 and L_3, $\mathcal{C}$ will deliver $\perp$ to $\mathcal{A}_1$.

Challenge: $\mathcal{A}_1$ delivers two same length messages M_0^*, M_1^* and the identity of challenger ID_r^* to $\mathcal{C}$. $\mathcal{C}$ firstly searches $(ID_r^*, h_1^*, D_r^*, x^*, sk_r^*)$ from L_{sk}, then it executes the following steps:

- If $ID_r \neq ID_r^*$, $\mathcal{C}$ aborts.
- If $ID_r = ID_r^*$, $\mathcal{C}$ picks $\rho \in \{0,1\}, r_1^*, r_2^* \in Z_p^*$, $C_2^*, C_3^* \in G_1$, $h_3^* \in \{0,1\}^{l_1+l_2}$ and does the following procedures:
  - Compute $C_1^* = g^c$.
  - Compute $C_4^* = h_3^* \oplus (M_\rho^*||r_2^*)$.
  - Calculate $h^* = H_4(M_\rho^*||C_1^*||C_2^*||C_3^*||C_4^*||r_2^*)$.
  - Calculate $C_5^* = U^{r_1^*} \cdot sk_s^{*h^*}$.
  - Finally, $\mathcal{C}$ delivers $C^* = (C_1^*, C_2^*, C_3^*, C_4^*, C_5^*)$ to $\mathcal{A}_1$.

Phase 2: The similar queries are executed by $\mathcal{A}_1$ as *Phase 1*. Nonetheless, $\mathcal{A}_1$ cannot perform the Partial private key query, Private key query on ID_r^* and the replaced identity ID_r'. Besides, the Unsigncryption query on (C^*, sk_r^*, ID_s^*) cannot be issued by $\mathcal{A}_1$.

Guess: $\mathcal{A}_1$ guesses the bit $\rho' \in \{0,1\}$. Suppose $\rho' = \rho$, this game wins. Therefore, $\mathcal{C}$ picks an item (W^*, h_3^*) from L_3, then BDHP can be solved as: $W^{*(x^*)^{-1}} = e(g,g)^{abc}$. ■

Theorem 2: Assuming an adversary $\mathcal{A}_2$ can break the security of IND-CCA, there has a challenger $\mathcal{C}$ can address the BDHP with a non-negligible advantage. $\mathcal{A}_2$ can access the master secret key s. Nonetheless, it cannot replace the public key of receiver.

Proof: Inputting an item (g, g^a, g^b, g^c), the target of $\mathcal{C}$ is to generate $e(g,g)^{abc}$. Then, $\mathcal{A}_2$ and $\mathcal{C}$ jointly perform the following steps.

Setup: The Setup step is firstly executed by $\mathcal{C}$. Then, $\mathcal{C}$ sets $P_{pub} = g^s$ and delivers the system parameters to $\mathcal{A}_2$. Besides, the PKI-KG operation is performed to generate the sender's key pair (sk_s^*, ID_s^*) to $\mathcal{A}_2$.

Phase 1: $\mathcal{A}_2$ makes the following queries, $\mathcal{C}$ preserves five lists L_0, L_1, L_2, L_3, L_4.

- H_0 query: When executing the query, $\mathcal{C}$ firstly judges whether this query on ID_s was performed. If it has been executed, the previous result is returned. Else, $\mathcal{C}$ picks $h_0 \in G_1$ and inserts (ID_s, h_0) into L_0. Finally, $\mathcal{C}$ delivers h_0 to $\mathcal{A}_2$.
- H_1 query: When executing the query, $\mathcal{C}$ firstly judges whether this query on ID_r was performed. If it has been executed, the previous result is returned. Else, the following operations are performed.
  - If $ID_r \neq ID_r^*$, $\mathcal{C}$ picks $v \in Z_p^*$ and computes $h_1 = g^v$, then it inserts (ID_r, h_1) into L_1. Finally, $\mathcal{C}$ delivers h_1 to $\mathcal{A}_2$.
  - If $ID_r = ID_r^*$, $\mathcal{C}$ sets $h_1^* = g^b$ and inserts (ID_r^*, h_1^*) into L_1. Finally, $\mathcal{C}$ delivers h_1^* to $\mathcal{A}_2$.
- H_2 query: When executing the query, $\mathcal{C}$ firstly judges whether this query on M was performed. If it has been executed, the previous result is returned. Else, $\mathcal{C}$ picks $h_2 \in G_1$ and inserts (M, h_2) into L_2. Finally, $\mathcal{C}$ delivers h_2 to $\mathcal{A}_2$.
- H_3 query: When executing the query, $\mathcal{C}$ firstly judges whether this query on W was performed. If it has been executed, the previous result is returned. Else, $\mathcal{C}$ picks $h_3 \in \{0,1\}^{l_1+l_2}$ and inserts (W, h_3) into L_3. Finally, $\mathcal{C}$ delivers h_3 to $\mathcal{A}_2$.
- H_4 query: When executing the query, $\mathcal{C}$ firstly judges whether this query on $(M, C_1, C_2, C_3, C_4, r_2)$ was performed. If it has been executed, the previous result is returned. Else, $\mathcal{C}$ picks $h_4 \in Z_p^*$ and inserts $(M, C_1, C_2, C_3, C_4, r_2, h_4)$ into L_4. Finally, $\mathcal{C}$ delivers h_4 to $\mathcal{A}_2$.
- Public key query: Upon executing the query on ID_r, $\mathcal{C}$ firstly judges whether this query on ID_r was performed. If it has been executed, the previous result is returned. Else, the following operations are performed.
  - If $ID_r \neq ID_r^*$, $\mathcal{C}$ picks $x \in Z_p^*$ and computes $PK_r = P_{pub}^x$. Then, it inserts (ID_r, PK_r) into L_{pk} and delivers PK_r to $\mathcal{A}_2$.
  - If $ID_r = ID_r^*$, $\mathcal{C}$ computes $PK_r = g^{as}$. Finally, $\mathcal{C}$ inserts (ID_r^*, PK_r^*) into L_{pk}. Finally, $\mathcal{C}$ delivers PK_r^* to $\mathcal{A}_2$.
- Private key query: Upon executing the query on ID_r, the following operations are performed.
  - If $ID_r \neq ID_r^*$, $\mathcal{C}$ computes $D_r = h_1^s = g^{vs}$ and $sk_r = D_r^x = g^{vsx}$. Then, it inserts $(ID_r, h_1, D_r, x, sk_r)$ into L_{sk}. Finally, $\mathcal{C}$ delivers sk_r to $\mathcal{A}_2$.
  - If $ID_r = ID_r^*$, $\mathcal{C}$ aborts.
- Unsigncryption query: Upon receiving the ciphertext $C = (C_1, C_2, C_3, C_4, C_5)$, $\mathcal{C}$ firstly executes the Public key query and Private key query on ID_r. Besides, $\mathcal{C}$ continues as below.
  - If $ID_r \neq ID_r^*$, $\mathcal{C}$ searches $(ID_r, h_1, D_r, x, sk_r)$ from L_{sk}. Then, $\mathcal{C}$ delivers the outcome of Unsigncryption (C, sk_r, ID_s^*) to $\mathcal{A}_2$.
  - If $ID_r = ID_r^*$, $\mathcal{C}$ searches the tuples (ID_s, h_0), (ID_r^*, h_1^*) and (W, h_3) from L_0, L_1 and L_3 correspondingly, $\mathcal{C}$ continues the following steps:
    1) Calculate $C_4 \oplus h_3 = M'||r_2'$.
    2) Compute $h' = H_4(M'||C_1||C_2||C_3||C_4||r_2')$.
    3) Check whether the following equations $C_3 \stackrel{?}{=} h_1^{*r_2'}$ and $e(C_5, g) \stackrel{?}{=} e(C_1, U)e(h_0^{h'}, P_{pub})$ hold or not. If hold, $\mathcal{C}$ delivers M' to $\mathcal{A}_2$.

  If the tuples do not exist in L_0, L_1 and L_3, $\mathcal{C}$ will deliver $\perp$ to $\mathcal{A}_2$.

Challenge: $\mathcal{A}_2$ delivers two same length messages M_0^*, M_1^* and the identity of challenger ID_r^* to $\mathcal{C}$. $\mathcal{C}$ firstly searches $(ID_r^*, h_1^*, D_r^*, x^*, sk_r^*)$ from L_{sk}, then it executes the following steps:

- If $ID_r \neq ID_r^*$, $\mathcal{C}$ aborts.
- If $ID_r = ID_r^*$, $\mathcal{C}$ picks $\rho \in \{0,1\}, r_1^*, r_2^* \in Z_p^*$, $C_2^*, C_3^* \in G_1$, $h_3^* \in \{0,1\}^{l_1+l_2}$ and does the following procedures:
  - Compute $C_1^* = g^c$.
  - Compute $C_4^* = h_3^* \oplus (M_\rho^* || r_2^*)$.
  - Calculate $h^* = H_4(M_\rho^* || C_1^* || C_2^* || C_3^* || C_4^* || r_2^*)$.
  - Calculate $C_5^* = U^{r_1^*} \cdot sk_s^{*h^*}$.
  - Finally, $\mathcal{C}$ delivers $C^* = (C_1^*, C_2^*, C_3^*, C_4^*, C_5^*)$ to $\mathcal{A}_2$.

Phase 2: The similar queries are executed by $\mathcal{A}_2$ as *Phase 1*. Nonetheless, $\mathcal{A}_2$ cannot execute the Private key query on ID_r^*. Besides, the Unsigncryption query on (C^*, sk_r^*, ID_s^*) cannot be issued by $\mathcal{A}_2$.

Guess: $\mathcal{A}_2$ guesses the bit $\rho' \in \{0,1\}$. Suppose $\rho' = \rho$, this game wins. Therefore, $\mathcal{C}$ picks an item (W^*, h_3^*) from L_3, then BDHP can be solved as: $W^{*(s^*)^{-1}} = e(g,g)^{abc}$. ■

Theorem 3: Assuming an adversary $\mathcal{A}_3$ can break the security of OW-CCA, there has a challenger $\mathcal{C}$ can address the BDHP with a non-negligible advantage.

Proof: Inputting the tuple (g, g^a, g^b, g^c), the target of $\mathcal{C}$ is to generate $e(g,g)^{abc}$. Then, $\mathcal{A}_3$ and $\mathcal{C}$ jointly perform the following steps.

Setup: The Setup step is firstly executed by $\mathcal{C}$. Then, $\mathcal{C}$ sets $P_{pub} = g^a$ and delivers the system parameters to $\mathcal{A}_3$. Besides, the PKI-KG operation is performed to generate the sender's key pair (sk_s^*, ID_s^*) to $\mathcal{A}_3$.

Phase 1: $\mathcal{A}_3$ makes the following queries, $\mathcal{C}$ preserves five lists L_0, L_1, L_2, L_3, L_4.

- H_i query $(0 \leq i \leq 4)$: The hash queries are the same as **Theorem 1**.
- The queries from public key, partial private key, private key, replace public key and unsigncryption are the same as **Theorem 1**.
- Trapdoor query: When executing this query, $\mathcal{C}$ picks $y, s_2 \in Z_p^*$. Then it returns $gtd = (td_1, td_2) = (g^y, g^{s_2 y})$ to $\mathcal{A}_3$.

Challenge: $\mathcal{A}_3$ delivers the message M^* and the identity of challenger ID_r^* to $\mathcal{C}$. $\mathcal{C}$ firstly searches $(ID_r^*, h_1^*, D_r^*, x^*, sk_r^*)$ from L_{sk}, then it continues with the steps as below:

- If $ID_r \neq ID_r^*$, $\mathcal{C}$ aborts.
- If $ID_r = ID_r^*$, $\mathcal{C}$ picks $r_1^*, r_2^* \in Z_p^*$, $C_2^*, C_3^* \in G_1$, $h_3^* \in \{0,1\}^{l_1+l_2}$ and does the following procedures:
  - Compute $C_1^* = g^c$.
  - Compute $C_4^* = h_3^* \oplus (M^* || r_2^*)$.
  - Calculate $h^* = H_4(M^* || C_1^* || C_2^* || C_3^* || C_4^* || r_2^*)$.
  - Calculate $C_5^* = U^{r_1^*} \cdot sk_s^{*h^*}$.
  - Finally, $\mathcal{C}$ delivers $C^* = (C_1^*, C_2^*, C_3^*, C_4^*, C_5^*)$ to $\mathcal{A}_3$.

Phase 2: The similar queries are executed by $\mathcal{A}_3$ as *Phase 1*. Nonetheless, $\mathcal{A}_3$ cannot perform the Partial private key query, Private key query on ID_r^* and the replaced identity ID_r'. Besides, the Unsigncryption query on (C^*, sk_r^*, ID_s^*) also cannot be issued by $\mathcal{A}_3$.

Guess: $\mathcal{A}_3$ guesses M'. Suppose $M' = M^*$, this game wins. Thus, $\mathcal{C}$ picks an item (W^*, h_3^*) from L_3, and BDHP can be solved as: $W^{*(x^*)^{-1}} = e(g,g)^{abc}$. ■

Theorem 4: Assuming an adversary $\mathcal{A}_4$ can break the security of OW-CCA, there has a challenger $\mathcal{C}$ can address the BDHP with a non-negligible advantage.

Proof: Inputting the tuple (g, g^a, g^b, g^c), the target of $\mathcal{C}$ is to generate $e(g,g)^{abc}$. Then, $\mathcal{A}_4$ and $\mathcal{C}$ jointly perform the following steps.

Setup: The Setup step is firstly performed by $\mathcal{C}$. $\mathcal{C}$ sets $P_{pub} = g^s$ and delivers the system parameters to $\mathcal{A}_4$. Besides, the PKI-KG operation is performed to generate the sender's key pair (sk_s^*, ID_s^*) to $\mathcal{A}_4$.

Phase 1: $\mathcal{A}_4$ makes the following queries, $\mathcal{C}$ preserves five lists L_0, L_1, L_2, L_3, L_4.

- H_i query $(0 \leq i \leq 4)$: The hash queries are the same as **Theorem 2**.
- The queries from public key, private key and unsigncryption are the same as **Theorem 1**.
- Trapdoor query: When executing this query, $\mathcal{C}$ picks $y, s_2 \in Z_p^*$. Then it returns $gtd = (td_1, td_2) = (g^y, g^{s_2 y})$ to $\mathcal{A}_4$.

Challenge: $\mathcal{A}_4$ delivers the message M^* and the identity of challenger ID_r^* to $\mathcal{C}$. $\mathcal{C}$ firstly searches $(ID_r^*, h_1^*, D_r^*, x^*, sk_r^*)$ from L_{sk}, then it executes the following steps:

- If $ID_r \neq ID_r^*$, $\mathcal{C}$ aborts.
- If $ID_r = ID_r^*$, $\mathcal{C}$ picks $\rho \in \{0,1\}, r_1^*, r_2^* \in Z_p^*$, $C_2^*, C_3^* \in G_1$, $h_3^* \in \{0,1\}^{l_1+l_2}$ and does the following procedures:
  - Compute $C_1^* = g^c$.
  - Compute $C_4^* = h_3^* \oplus (M^* || r_2^*)$.
  - Calculate $h^* = H_4(M^* || C_1^* || C_2^* || C_3^* || C_4^* || r_2^*)$.
  - Calculate $C_5^* = U^{r_1^*} \cdot sk_s^{*h^*}$.
  - Finally, $\mathcal{C}$ delivers $C^* = (C_1^*, C_2^*, C_3^*, C_4^*, C_5^*)$ to $\mathcal{A}_4$.

Phase 2: The similar queries are executed by $\mathcal{A}_4$ as *Phase 1*. Nonetheless, $\mathcal{A}_4$ cannot perform the Private key query on ID_r^*. Besides, the Unsigncryption query on (C^*, sk_r^*, ID_s^*) cannot be issued by $\mathcal{A}_4$.

Guess: $\mathcal{A}_4$ guesses M'. Suppose $M' = M^*$, this game wins. Thus, $\mathcal{C}$ picks an item (W^*, h_3^*) from L_3, and BDHP can be solved as: $W^{*(s^*)^{-1}} = e(g,g)^{abc}$. ■

Theorem 5: Assuming an adversary $\mathcal{A}_5$ can break the security of EUF-CMA, there has a challenger $\mathcal{C}$ can address the CDHP with a non-negligible advantage.

Proof: Inputting the tuple (g, g^a, g^b), the target of $\mathcal{C}$ is to generate g^{ab}. Then, $\mathcal{A}_5$ and $\mathcal{C}$ jointly perform the following steps.

Setup: The Setup step is firstly performed by $\mathcal{C}$. $\mathcal{C}$ sets $P_{pub} = g^a$ and delivers the system parameters to $\mathcal{A}_5$. Besides, the IBC-KG operation is performed to generate the receiver's key pair (sk_r^*, PK_r^*) to $\mathcal{A}_5$.

Queries: $\mathcal{A}_5$ makes the following queries from $\mathcal{C}$.

- H_0 query: When executing this query, $\mathcal{C}$ firstly checks whether this query on ID_s was performed. If it has been executed, the previous result is returned. Else, the following operations are performed.

- If $ID_s \neq ID_s^*$, $\mathcal{C}$ picks $\xi \in Z_p^*$ and computes $h_0 = g^{\xi}$, then it inserts (ID_s, h_0) into L_0. Finally, $\mathcal{C}$ delivers h_0 to $\mathcal{A}_5$.
- If $ID_s = ID_s^*$, $\mathcal{C}$ sets $h_0^* = g^b$ and inserts (ID_s^*, h_0^*) into L_0. Finally, $\mathcal{C}$ delivers h_0^* to $\mathcal{A}_5$.

- H_1 query: $\mathcal{C}$ firstly checks whether this query on ID_r was performed. If it has been executed, the previous result is returned. Else, $\mathcal{C}$ picks $h_1 \in G_1$ and inserts (ID_r, h_1) into L_1. Finally, $\mathcal{C}$ delivers h_1 to $\mathcal{A}_5$.
- H_2 query: When executing this query, $\mathcal{C}$ firstly checks whether this query on M was performed. If it has been executed, the previous result is returned. Else, $\mathcal{C}$ picks $h_2 \in G_1$ and inserts (M, h_2) into L_2. Finally, $\mathcal{C}$ delivers h_2 to $\mathcal{A}_5$.
- H_3 query: When executing this query, $\mathcal{C}$ firstly checks whether this query on W was performed. If it has been executed, the previous result is returned. Else, $\mathcal{C}$ picks $h_3 \in \{0,1\}^{l_1+l_2}$ and inserts (W, h_3) into L_3. Finally, $\mathcal{C}$ delivers h_3 to $\mathcal{A}_5$.
- H_4 query: When executing this query, $\mathcal{C}$ firstly checks whether this query on $(M, C_1, C_2, C_3, C_4, r_2)$ was performed. If it has been executed, the previous result is returned. Else, $\mathcal{C}$ picks $h_4 \in Z_p^*$ and inserts $(M, C_1, C_2, C_3, C_4, r_2, h_4)$ into L_4. Finally, $\mathcal{C}$ delivers h_4 to $\mathcal{A}_5$.
- Private key query: When executing this query on ID_s, the following operations are performed.
  - If $ID_s \neq ID_s^*$, $\mathcal{C}$ computes $sk_s = h_0^a = g^{a\xi}$. Finally, $\mathcal{C}$ delivers sk_s to $\mathcal{A}_5$.
  - If $ID_s = ID_s^*$, $\mathcal{C}$ aborts.
- Signcryption query: When receiving the message M and the sender's identity ID_s, it will execute the operations as below.
  - If $ID_s \neq ID_s^*$, $\mathcal{C}$ searches (ID_s, h_0) from L_0. Then, $\mathcal{C}$ computes the private key $sk_s = h_0^s = g^{a\xi}$ and delivers the outcome of Signcryption (M, sk_s, PK_r^*) to $\mathcal{A}_5$.
  - If $ID_s = ID_s^*$, $\mathcal{C}$ picks $r_1', r_2', x_s \in Z_p^*$, $C_2, C_3 \in G_1$, $C_4 \in \{0,1\}^{l_1+l_2}$, then $\mathcal{C}$ sets $Q_s = g^{x_s}$ and continues the following steps.
    1) Calculate $C_1 = g^{r_1'}$.
    2) Compute $h' = H_4(M||C_1||C_2||C_3||C_4||r_2')$.
    3) Compute $C_5 = u^{r_1'} \cdot P_{pub}^{x_s h'}$.
    4) Finally, $\mathcal{C}$ delivers the valid $C = (C_1, C_2, C_3, C_4, C_5)$ to $\mathcal{A}_5$.

  Correctness:

$$\begin{aligned} e(C_5, g) &= e(u^{r_1'} \cdot P_{pub}^{x_s h'}, g) \\ &= e(u^{r_1'}, g) e(P_{pub}, g^{x_s h'}) \\ &= e(g^{r_1'}, U) e(g^{x_s h'}, P_{pub}) \\ &= e(C_1, U) e(Q_s^{h'}, P_{pub}) \end{aligned}$$

Forgery: Assume that $\mathcal{A}_5$ can successfully forge the ciphertext. Then it returns the forged ciphertext $C^* = (C_1^*, C_2^*, C_3^*, C_4^*, C_5^*)$ and h^* on ID_s^* to $\mathcal{C}$. $\mathcal{C}$ picks $t \in Z_p^*$ and sets $U = g^t$. We have $P_{pub} = g^a, Q_s^* = g^b$ and thus

$$C_5^* = C_1^{*t} \cdot g^{abh}$$

$$g^{ab} = (\frac{C_5^*}{C_1^*})^{h^{*-1}}$$

where the CDHP can be solved as: $g^{ab} = (\frac{C_5^*}{C_1^*})^{h^{*-1}}$.

Resistance to collusion attack: During the "Signcryption" step, the sender randomly picks $r_1, r_2 \in Z^*_{\ p}$ to generate the ciphertext. It is precisely due to the existence of random numbers that the generated ciphertext is constantly changing. In addition, we can observe that $C_1 = g^{r_1}$. If the collusion attackers intend to recover the random number r_1 from C_1, this is contrary to principle of the Discrete Logarithm Problem (DLP), which is unbreakable in polynomial time. Therefore, the proposed approach has the ability to resist the collusion attacks.

Resistance to denial-of-service attack: In this paper, the construction of the proposed scheme is based on signcryption mechanism, which can realize the functions of signature and encryption in the same step. The signature operation can identify and verify the ciphertext that exists in the system. For example, in the section "Construction", we can observe that during the "Unsigncryption" step, the corresponding verification formulas are: $C_3 \stackrel{?}{=} Q_r^{r_2'}$, $e(C_5, g) \stackrel{?}{=} e(C_1, U) e(Q_s^{h'}, P_{pub})$. Only the valid ciphertext information can be received, while the invalid ciphertext information will be discarded. Therefore, when the system has a large amount of information at the same time, our scheme will filter the information to some extent, thus partly protecting against denial-of-service attacks.

VI. Performance Analyze

In this section, the security function and performance of the proposed scheme with literatures [10], [11], [31], [32] are elaborated. With regard to the comparison of computation efficiency, t he e xperiment e nvironment i s e xecuted o n Windows 10 PC configured w ith i 7-11700F C PU @ 2.50GHz and 16 GB memory. As for pairing-based schemes, for achieving the security level of 1024-bits RSA, a supersingular curve $E/F_p : y^2 = x^3 + x$ over a finite fi eld F_p is ut ilized, which defines a Tate pairing $e : G_1 \times G_1 \to G_2$. Here, the embedding degree is 2, the 160-bit Solinas prime is $q = 2^{159} + 2^{17} + 1$, the 512-bit prime is p. The cryptographic operations' running time are elaborated in Table II. It should be noted that the schemes based on PKI have the certificate management problem. We set T_{cr} to represent the cost of certificate management. Generally, T_{cr} is about 1.2s. Besides, we have $|Z^*| = 20$ bytes and $|G_1| = |G_2| = 128$ bytes.

TABLE II: The cryptographic operation time

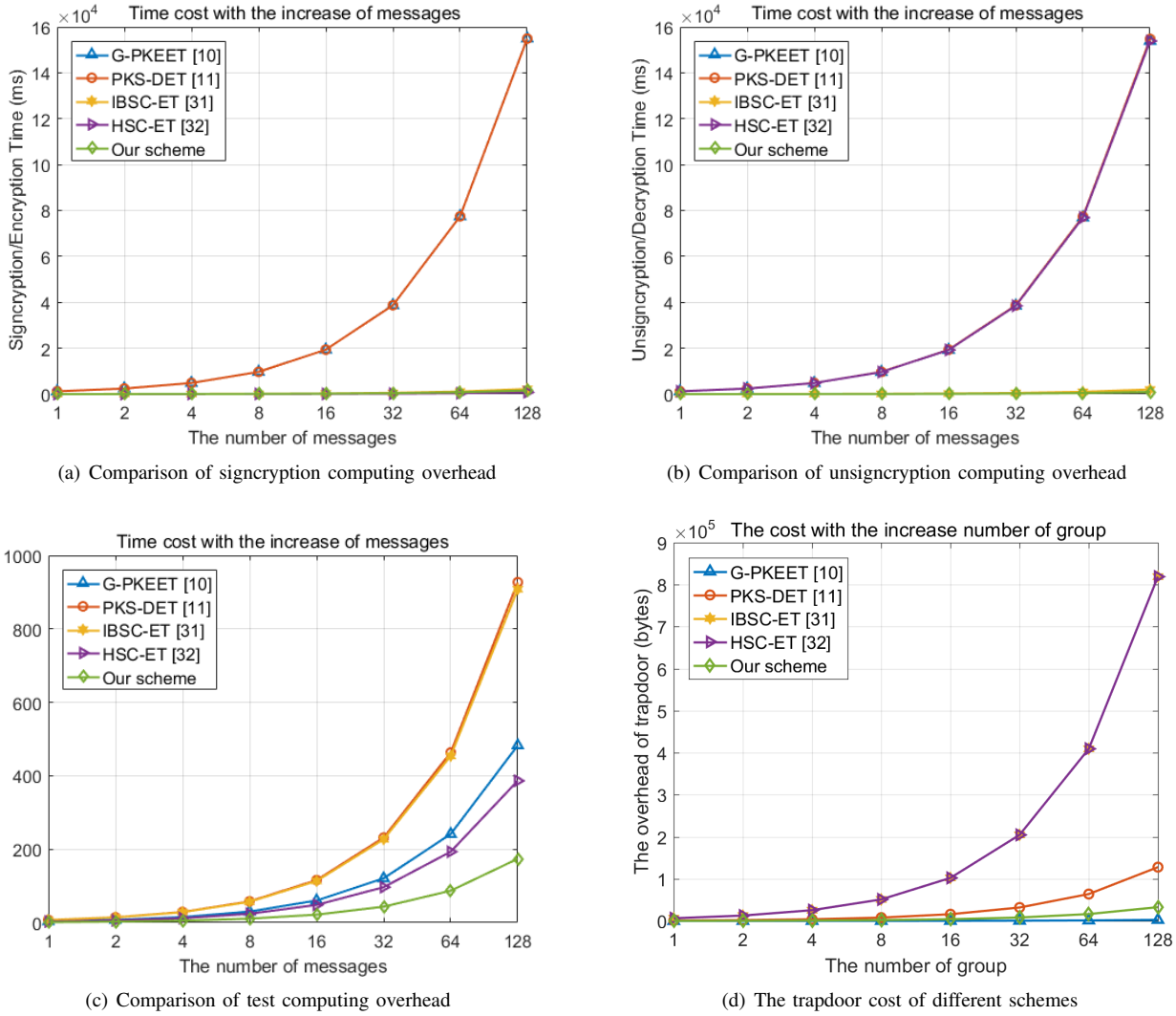
Operation	T_{bp}	T_{hg}	T_{e1}	T_{e2}
Time(ms)	0.6795	2.1870	1.2059	0.0751

A. Comparison of Computational Overhead

The computation overhead of competitive schemes is evaluated as follows, the comparison result is listed in Table

TABLE III: Comparison the computation overhead of competitive schemes

Protocol	Signcryption or Encryption	Unsigncryption or Decryption	Equality Test
G-PKEET [10]	$5T_{e1}$+$2T_{hg}$+T_{cr}	$2T_{e1}$+T_{cr}	$2T_{bp}$+$2T_{e1}$
PKS-DET [11]	T_{bp}+$4T_{e1}$+T_{e2}+$2T_{hg}$ +T_{cr}	$3T_{bp}$+$2T_{e1}$+T_{e2}+$2T_{hg}$+T_{cr}	$4T_{bp}$+$2T_{e2}$+$2T_{hg}$
IBSC-ET [31]	$2T_{bp}$+$6T_{e1}$+$2T_{e2}$+$4T_{hg}$	$5T_{bp}$+$3T_{e1}$+$4T_{hg}$	$4T_{bp}$+$2T_{hg}$
HSC-ET [32]	$5T_{e1}$+$2T_{e2}$	$3T_{bp}$+T_{e2} +T_{cr}	$4T_{bp}$+$4T_{e2}$
Our scheme	T_{bp}+$6T_{e1}$+T_{e2}+$2T_{hg}$	$4T_{bp}$+$2T_{e1}$+T_{hg}	$2T_{bp}$



(a) Comparison of signcryption computing overhead

(b) Comparison of unsigncryption computing overhead

(c) Comparison of test computing overhead

(d) The trapdoor cost of different schemes

Fig. 11: The computation and trapdoor cost of competitive protocols.

TABLE IV: Comparison the communication cost of competitive protocols

Protocol	The length of public key	The length of ciphertext	The length of trapdoor
G-PKEET [10]	$\|G_1\|$	$5\|G_1\|$+$\|Z_p^*\|$	$\|Z_p^*\|$
PKS-DET [11]	$2\|G_1\|$	$4\|G_1\|$+$\|Z_p^*\|$	$\|Z_p^*\|$
IBSC-ET [31]	$*$	$5\|G_1\|$+$\|Z_p^*\|$	$\|G_1\|$
HSC-ET [32]	$*$	$4\|G_1\|$+$2\|Z_p^*\|$	$\|G_1\|$
Our scheme	$\|G_1\|$	$5\|G_1\|$+$\|Z_p^*\|$	$2\|G_1\|$

* Legends: $*$: The length of identity, we set it as $|G_1|$.

```
** Event #1  t=3.358384818979  Aloha.host[15] (Host, id=18)  on selfmsg send/endTx (omnetpp::cMessage, id=16)
INFO:generating packet pk-18-#0
** Event #2  t=3.35838673475  Aloha.server (Server, id=2)  on pk-18-#0 (omnetpp::cPacket, id=21)
INFO:started receiving
** Event #3  t=3.614695498591  Aloha.host[15] (Host, id=18)  on selfmsg send/endTx (omnetpp::cMessage, id=16)
** Event #4  t=3.614697414362  Aloha.server (Server, id=2)  on selfmsg end-reception (omnetpp::cMessage, id=0)
INFO:reception finished
```

(a) The transmission simulation in the scheme [10].

```
** Event #1  t=3.358384818979  Aloha.host[15] (Host, id=18)  on selfmsg send/endTx (omnetpp::cMessage, id=16)
INFO:generating packet pk-18-#0
** Event #2  t=3.35838673475  Aloha.server (Server, id=2)  on pk-18-#0 (omnetpp::cPacket, id=21)
INFO:started receiving
** Event #3  t=3.564986760727  Aloha.host[15] (Host, id=18)  on selfmsg send/endTx (omnetpp::cMessage, id=16)
** Event #4  t=3.564988676498  Aloha.server (Server, id=2)  on selfmsg end-reception (omnetpp::cMessage, id=0)
INFO:reception finished
```

(b) The transmission simulation in the scheme [11].

```
** Event #1  t=3.358384818979  Aloha.host[15] (Host, id=18)  on selfmsg send/endTx (omnetpp::cMessage, id=16)
INFO:generating packet pk-18-#0
** Event #2  t=3.35838673475  Aloha.server (Server, id=2)  on pk-18-#0 (omnetpp::cPacket, id=21)
INFO:started receiving
** Event #3  t=3.614695498591  Aloha.host[15] (Host, id=18)  on selfmsg send/endTx (omnetpp::cMessage, id=16)
** Event #4  t=3.614697414362  Aloha.server (Server, id=2)  on selfmsg end-reception (omnetpp::cMessage, id=0)
INFO:reception finished
```

(c) The transmission simulation in the scheme [31].

```
** Event #1  t=3.358384818979  Aloha.host[15] (Host, id=18)  on selfmsg send/endTx (omnetpp::cMessage, id=16)
INFO:generating packet pk-18-#0
** Event #2  t=3.35838673475  Aloha.server (Server, id=2)  on pk-18-#0 (omnetpp::cPacket, id=21)
INFO:started receiving
** Event #3  t=3.572753751018  Aloha.host[15] (Host, id=18)  on selfmsg send/endTx (omnetpp::cMessage, id=16)
** Event #4  t=3.572755666789  Aloha.server (Server, id=2)  on selfmsg end-reception (omnetpp::cMessage, id=0)
INFO:reception finished
```

(d) The transmission simulation in the scheme [32].

```
** Event #1  t=3.358384818979  Aloha.host[15] (Host, id=18)  on selfmsg send/endTx (omnetpp::cMessage, id=16)
INFO:generating packet pk-18-#0
** Event #2  t=3.35838673475  Aloha.server (Server, id=2)  on pk-18-#0 (omnetpp::cPacket, id=21)
INFO:started receiving
** Event #3  t=3.614695498591  Aloha.host[15] (Host, id=18)  on selfmsg send/endTx (omnetpp::cMessage, id=16)
** Event #4  t=3.614697414362  Aloha.server (Server, id=2)  on selfmsg end-reception (omnetpp::cMessage, id=0)
INFO:reception finished
```

(e) The transmission simulation in our scheme.

Fig. 12: Comparison of ciphertext communication overhead in OMNet++

TABLE V: Comparison the functionality of competitive schemes

Protocol	Confidentiality	Unforgeability	Equality Test	Group	Heterogeneous	Security assumptions
G-PKEET [10]	✓	×	✓	✓	×	CDHA
PKS-DET [11]	✓	✓	✓	×	×	CDHA& BDHA
IBSC-ET [31]	✓	✓	✓	×	×	CDHA& BDHA
HSC-ET [32]	✓	✓	✓	×	✓	BDHIA&CDHIA
Our scheme	✓	✓	✓	✓	✓	CDHA& BDHA

* Legends: ✓: this scheme supports this function, ×: this scheme cannot support this function, BDHIA: Bilinear Diffie-Hellman Inversion Assumption, CDHIA: Computational Diffie-Hellman Inversion Assumption.

III. For example, when signcrypting a message during the protocol [31], six exponentiations in G_1, two exponentiations in G_2, two pairing operations, and four hash to G_1 are needed by the sender, thus the signcryption computation overhead of [31] is $2T_{bp}$+$6T_{e1}$+$2T_{e2}$+$4T_{hg}$=$17.4926ms$. In Fig. 11(a), the signcryption or encryption cost with regard to increase number of messages is evaluated. It is important to find that the schemes in [10], [11] are constructed with PKI cryptography, the scheme in [32] is delivered the message from PKI to IBC system, therefore, in the actual transmission process, the aforementioned certificate management operation takes up a lot of overhead, which is more expensive than our HSC-GET. From Fig. 11(a), we can observe that our proposed scheme has a lower overhead than literatures in [10], [11], [31], [32].

Similarly, when unsigncrypting a ciphertext during the protocol [31], five p airing o perations, t hree exponentiations in G_1, and four hash to G_1 are needed by the receiver, and thus the unsigncryption computation overhead of [31] is $5T_{bp}$+$3T_{e1}$+$4T_{hg}$=$15.7632ms$. In Fig. 11(b), the unsigncryption or decryption cost with regard to increase number of messages is evaluated. Our scheme is more efficient than that in [10], [11], [31], [32]. When executing the equality test during the protocol [31], four pairing operations, and two hash to G_1 are needed by the cloud server, and thus the test cost of [31] is $4T_{bp}$+$2T_{hg}$=$7.092ms$. In Fig. 11(c), the cost with regard to the number of test times is evaluated. Through the analysis, our scheme reduces the test overhead about 63.96%, 81.23%, 80.84%, and 54.98% compared to [10], [11], [31] and [32], respectively. To sum up, the total computing overhead of the proposed scheme is lower than competitive protocols not only on the vehicle side (including Signcryption/Encryption

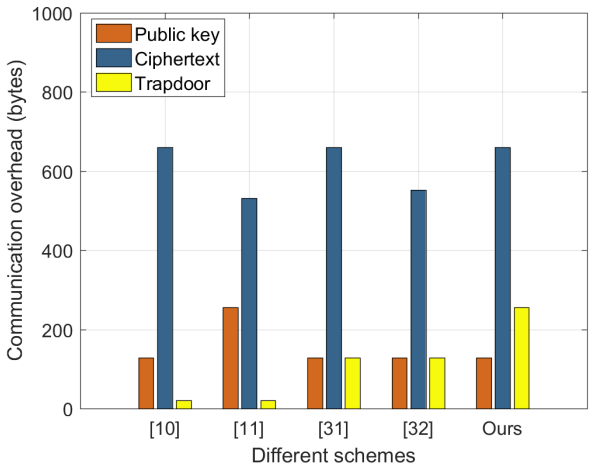


Fig. 13: The communication overhead of different schemes.

and Unsigncryption/Decryption part), but also on the cloud side (including the test part).

B. Comparison of Communication Overhead

The communication overhead of competitive protocols is evaluated as follows, the comparison result is listed in Table IV. For instance, in IBSC-ET scheme [31], the public key length is $*$, and we set the length of $*$ is $|G_1| = 128$ bytes. In addition, the resulting ciphertext length for IBSC-ET is $5|G_1|+|Z_p^*| = 660$ bytes, the trapdoor size is defined $|G_1| = 128$ bytes. Besides, we set one group of vehicles including 50 vehicles, the trapdoor overhead with the increasing number of groups is shown in Fig. 11(d). Obviously, the trapdoor storage cost of our construction is lower than that in scheme [11], [31], [32]. When there is only one vehicle in the system, the communication overhead of different schemes is demonstrated in Fig. 13. From Fig. 13, our construction has the same overhead as [10], [31], [32] and is lower than [11] in terms of public key length. As for the ciphertext length, our scheme has the same length as [10] and [31]. Towards the trapdoor size, our HSC-GET is slightly more expensive than other works, this is forgivable due to the computation overhead during our test phase is more efficient t han other competitive schemes. Thus, the overhead between computation and communication achieves a compromise effect. In order to verify the communication overhead, we simulate the ciphertext communication process between vehicles with limited resources in the OMNet++ experiment. The specific results are shown in the Fig. 12, which demonstrates that the experimental result is consistent with the theoretical analysis.

C. Comparison Analysis of Functionality

The functionality of competitive schemes is evaluated as follows, the comparison result is listed in Table V. As the table shown, all of the works can guarantee the confidentiality of messages. With regard to the authenticity, the schemes in [11], [31] and [32] have the ability to preserve. All the schemes involved can perform the equality test. Nonetheless, only our scheme and G-PKEET [10] provide the group granularity authorization. The cloud server utilizes the group trapdoor instead of the inherent trapdoor from all receivers, which reduces the storage overhead of trapdoors greatly. As for the communication between devices deployed on heterogeneous cryptographic mechanisms, the HSC-ET scheme in [32] and our protocol provide the concrete solution for addressing this problem. All in all, our protocol can meet all the functions listed in the Table V, which is not the condition in the other protocols.

VII. Conclusion

In this paper, we constructed a heterogeneous signcryption scheme supporting group equality test for IoVs (HSC-GET), to provide a way for information transmission from IBC to CLC system. The rigorous security proof elaborates that the proposed scheme can ensure the confidentiality, unforgeability and integrity of messages simultaneously. Besides, the construction of proposed approach is based on cyclic group, and the adoption of big prime number enables the scheme more secure. In addition, the function of group equality test allows each group of vehicles to upload one group trapdoor to the cloud server, instead of N trapdoors from N vehicles in the traditional test scheme. According to the performance analysis, we can find that the computational and communication overhead are greatly reduced. Therefore, our construction is more suitable for IoVs.

References

[1] JiuJun Cheng, JunLu Cheng, MengChu Zhou, FuQiang Liu, ShangCe Gao, and Cong Liu. Routing in internet of vehicles: A review. *IEEE Transactions on Intelligent Transportation Systems*, 16(5):2339–2352, 2015.

[2] Xiangwang Hou, Zhiyuan Ren, Jingjing Wang, Wenchi Cheng, Yong Ren, Kwang-Cheng Chen, and Hailin Zhang. Reliable computation offloading for edge-computing-enabled software-defined iov. *IEEE Internet of Things Journal*, 7(8):7097–7111, 2020.

[3] Jing Wang, Libing Wu, Huaqun Wang, Kim-Kwang Raymond Choo, Lianhai Wang, and Debiao He. A secure and efficient multiserver authentication and key agreement protocol for internet of vehicles. *IEEE Internet of Things Journal*, 9(23):24398–24416, 2022.

[4] Minghui LiWang, Shijie Dai, Zhibin Gao, Xiaojiang Du, Mohsen Guizani, and Huaiyu Dai. A computation offloading incentive mechanism with delay and cost constraints under 5g satellite-ground iov architecture. *IEEE Wireless Communications*, 26(4):124–132, 2019.

[5] Juan Contreras-Castillo, Sherali Zeadally, and Juan Antonio Guerrero-Ibañez. Internet of vehicles: architecture, protocols, and security. *IEEE internet of things Journal*, 5(5):3701–3709, 2017.

[6] Chaosheng Feng, Keping Yu, Moayad Aloqaily, Mamoun Alazab, Zhihan Lv, and Shahid Mumtaz. Attribute-based encryption with parallel outsourced decryption for edge intelligent iov. *IEEE Transactions on Vehicular Technology*, 69(11):13784–13795, 2020.

[7] Hyung Tae Lee, San Ling, Jae Hong Seo, Huaxiong Wang, and Taek-Young Youn. Public key encryption with equality test in the standard model. *Information Sciences*, 516:89–108, 2020.

[8] Hyung Tae Lee, San Ling, Jae Hong Seo, and Huaxiong Wang. Semi-generic construction of public key encryption and identity-based encryption with equality test. *Information Sciences*, 373:419–440, 2016.

[9] Ganesh Gopal Deverajan, V Muthukumaran, Ching-Hsien Hsu, Marimuthu Karuppiah, Yeh-Ching Chung, and Ying-Huei Chen. Public key encryption with equality test for industrial internet of things system in cloud computing. *Transactions on Emerging Telecommunications Technologies*, 33(4):e4202, 2022.

[10] Yunhao Ling, Sha Ma, Qiong Huang, Ximing Li, and Yunzhi Ling. Group public key encryption with equality test against offline message recovery attack. *Information Sciences*, 510:16–32, 2020.

[11] Yujue Wang, HweeHwa Pang, Robert H Deng, Yong Ding, Qianhong Wu, and Bo Qin. Securing messaging services through efficient signcryption with designated equality test. *Information Sciences*, 490:146–165, 2019.

[12] Yuliang Zheng. Digital signcryption or how to achieve cost (signature & encryption)ł cost (signature)+ cost (encryption). In *Advances in CryptologyCRYPTO'97: 17th Annual International Cryptology Conference Santa Barbara, California, USA August 17–21, 1997 Proceedings 17*, pages 165–179. Springer, 1997.

[13] Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In *Advances in CryptologyEUROCRYPT 2002: International Conference on the Theory and Applications of Cryptographic Techniques Amsterdam, The Netherlands, April 28–May 2, 2002 Proceedings 21*, pages 83–107. Springer, 2002.

[14] Guangxia Xu, Jingnan Dong, Chuang Ma, Jun Liu, and Uchani Gutierrez Omar Cliff. A certificateless signcryption mechanism based on blockchain for edge computing. *IEEE Internet of Things Journal*, 2022.

[15] Nabeil Eltayieb, Rashad Elhabob, Alzubair Hassan, and Fagen Li. A blockchain-based attribute-based signcryption scheme to secure data sharing in the cloud. *Journal of Systems Architecture*, 102:101653, 2020.

[16] Ikram Ali, Tandoh Lawrence, Anyembe Andrew Omala, and Fagen Li. An efficient hybrid signcryption scheme with conditional privacy-preservation for heterogeneous vehicular communication in vanets. *IEEE Transactions on Vehicular Technology*, 69(10):11266–11280, 2020.

[17] John Malone-Lee. Identity-based signcryption. *Cryptology ePrint Archive*, 2002.

[18] Liqun Chen and John Malone-Lee. Improved identity-based signcryption. In *Public Key Cryptography*, volume 3386, pages 362–379. Springer, 2005.

[19] Benoit Libert and Jean-Jacques Quisquater. A new identity based signcryption scheme from pairings. In *Proceedings 2003 IEEE Information Theory Workshop (Cat. No. 03EX674)*, pages 155–158. IEEE, 2003.

[20] Yanan Zhao, Yunpeng Wang, Yuhao Liang, Haiyang Yu, and Yilong Ren. Identity-based broadcast signcryption scheme for vehicular platoon communication. *IEEE Transactions on Industrial Informatics*, 2022.

[21] Manuel Barbosa and Pooya Farshim. Certificateless signcryption. In *Proceedings of the 2008 ACM symposium on Information, computer and communications security*, pages 369–372, 2008.

[22] Zhenhua Liu, Yupu Hu, Xiangsong Zhang, and Hua Ma. Certificateless signcryption scheme in the standard model. *Information Sciences*, 180(3):452–464, 2010.

[23] Songqin Miao, Futai Zhang, Sujuan Li, and Yi Mu. On security of a certificateless signcryption scheme. *Information Sciences*, 232:475–481, 2013.

[24] Fagen Li, Hui Zhang, and Tsuyoshi Takagi. Efficient signcryption for heterogeneous systems. *IEEE Systems Journal*, 7(3):420–429, 2013.

[25] Guomin Yang, Chik How Tan, Qiong Huang, and Duncan S Wong. Probabilistic public key encryption with equality test. In *Topics in Cryptology-CT-RSA 2010: The Cryptographers Track at the RSA Conference 2010, San Francisco, CA, USA, March 1-5, 2010. Proceedings*, pages 119–131. Springer, 2010.

[26] Qiang Tang. Towards public key encryption scheme supporting equality test with fine-grained authorization. In *Information Security and Privacy: 16th Australasian Conference, ACISP 2011, Melbourne, Australia, July 11-13, 2011. Proceedings 16*, pages 389–406. Springer, 2011.

[27] Qiang Tang. Public key encryption schemes supporting equality test with authorisation of different granularity. *International journal of applied cryptography*, 2(4):304–321, 2012.

[28] Qiang Tang. Public key encryption supporting plaintext equality test and user-specified authorization. *Security and Communication Networks*, 5(12):1351–1362, 2012.

[29] Sha Ma. Identity-based encryption with outsourced equality test in cloud computing. *Information Sciences*, 328:389–402, 2016.

[30] Haipeng Qu, Zhen Yan, Xi-Jun Lin, Qi Zhang, and Lin Sun. Certificateless public key encryption with equality test. *Information Sciences*, 462:76–92, 2018.

[31] Hu Xiong, Yingzhe Hou, Xin Huang, and Yanan Zhao. Secure message classification services through identity-based signcryption with equality test towards the internet of vehicles. *Vehicular Communications*, 26:100264, 2020.

[32] Hu Xiong, Yanan Zhao, Yingzhe Hou, Xin Huang, Chuanjie Jin, Lili Wang, and Saru Kumari. Heterogeneous signcryption with equality test for iiot environment. *IEEE Internet of Things Journal*, 8(21):16142–16152, 2020.

[33] Hu Xiong, Yingzhe Hou, Xin Huang, Yanan Zhao, and Chien-Ming Chen. Heterogeneous signcryption scheme from ibc to pki with equality test for wbans. *IEEE Systems Journal*, 16(2):2391–2400, 2021.

[34] Xiangyu Pan, Yuqiao Jin, Ziqing Wang, and Fagen Li. A pairing-free heterogeneous signcryption scheme for unmanned aerial vehicles. *IEEE Internet of Things Journal*, 9(19):19426–19437, 2022.

[35] Hu Xiong, Hanxiao Wang, Weizhi Meng, and Kuo-Hui YehSenior Member. Attribute-based data sharing scheme with flexible search functionality for cloud assisted autonomous transportation system. *IEEE Transactions on Industrial Informatics*, 2023.

[36] Yunhao Ling, Sha Ma, Qiong Huang, and Ximing Li. A general two-server framework for ciphertext-checkable encryption against offline message recovery attack. In *Cloud Computing and Security: 4th International Conference, ICCCS 2018, Haikou, China, June 8–10, 2018, Revised Selected Papers, Part III*, pages 370–382. Springer, 2018.

[37] Sha Ma and Yunhao Ling. A general two-server cryptosystem supporting complex queries. In *Information Security Applications: 18th International Conference, WISA 2017, Jeju Island, Korea, August 24-26, 2017, Revised Selected Papers 18*, pages 249–260. Springer, 2018.

[38] Libing Wu, Yubo Zhang, and Debiao He. Dual server identity-based encryption with equality test for cloud computing. *J. Comput. Res. Dev.*, 54(10):2232–2243, 2017.

[39] Tong Wu, Sha Ma, Yi Mu, and Shengke Zeng. Id-based encryption with equality test against insider attack. In *Information Security and Privacy: 22nd Australasian Conference, ACISP 2017, Auckland, New Zealand, July 3–5, 2017, Proceedings, Part I 22*, pages 168–183. Springer, 2017.

[40] Yunhao Ling, Sha Ma, Qiong Huang, Ru Xiang, and Ximing Li. Group id-based encryption with equality test. In *Information Security and Privacy: 24th Australasian Conference, ACISP 2019, Christchurch, New Zealand, July 3–5, 2019, Proceedings 24*, pages 39–57. Springer, 2019.