Vehicular Communications 49 (2024) 100811

Contents lists available at ScienceDirect

Vehicular Communications

journal homepage: www.elsevier.com/locate/vehcom

An efficient heterogeneous online/offline anonymous certificateless signcryption with proxy re-encryption for Internet of Vehicles [☆]

Negalign Wake Hundera [a,b], Muhammad Umar Aftab [f], Dagmawit Mesfin [c], Fatene Dioubi [a,*], Huiying Xu [a], Xinzhong Zhu [a,d,e,*]

[a] School of Computer Science and Technology, Zhejiang Normal University, Jinhua, 321004, China
[b] Zhejiang Institute of Optoelectronics, Jinhua, China
[c] Department of Electrical Engineering, University of Notre Dame, Indiana, 46556, USA
[d] AI Research Institute of Beijing Geekplus Technology Co., Ltd., Beijing, 100101, China
[e] Research Institute of Hangzhou Artificial Intelligence, Zhejiang Normal University, Hangzhou, 311231, China
[f] Department of Computer Science, National University of Computer and Emerging Sciences, Islamabad, Chiniot-Faisalabad Campus, Chiniot 35400, Pakistan

ARTICLE INFO

Keywords:
Internet of Vehicles
Certificateless cryptosystem
Internet of Things
Identity-based cryptosystem
Online/offline signcryption

ABSTRACT

In the Internet of Vehicles (IoV) domain, the collection of traffic data is executed by intelligent devices and stored within a cloud-assisted IoV system. However, ensuring confidentiality and authorized access to data are the main problems of data storage. To address these problems, this paper proposes an efficient heterogeneous online/offline certificateless signcryption with a proxy re-encryption scheme (HOOCLS-PRE). This scheme enables IoV nodes in a certificateless cryptosystem (CLC) environment to store encrypted IoV-related data in the cloud. When an authorized user from an identity-based cryptosystem (IBC) wishes to access the data, the IoV node delegates the task of re-encrypting the data to the cloud, and only an authorized user can decrypt the data and verify its integrity and authenticity. The cloud cannot obtain any plaintext details about the data. In the proposed scheme, the signcryption process is split into offline and online phases. Most heavy computations are conducted without knowledge of the message during the offline phase. Only light computations are performed in the online phase when a message is available. The scheme protects the privacy and anonymity of vehicles by preventing adversaries from linking vehicle identities and locations. Moreover, a formal security proof is provided in the random oracle model. Finally, the performance analysis reveals that HOOCLS-PRE outperforms existing relevant schemes. Hence, HOOCLS-PRE is ideal for cloud-assisted IoV environments.

1. Introduction

The Internet of Things (IoT) is a network of physical objects equipped with sensors that collect and exchange data. It can be used in different areas, such as smart transportation, smart grids, and smart health [1–3]. The IoV is an IoT application that is dedicated to the interconnection of vehicles, sensors, and infrastructure to enhance transportation systems and improve quality of life [4,5]. In the IoV, vehicles are equipped with integrated sensors that collect data on parameters such as vehicle direction, driving speed, and route information. These data are then exchanged with other vehicles, roadside infrastructure, authorized users, and cloud computing to enable various services and applications [6,7]. The typical IoV scenario in Fig. 1 shows that a self-organizing network is formed when vehicles and roadside infrastructure communicate with each other. To collect and process information, each vehicle is outfitted with an On-Board Unit (OBU). These OBUs can interact with nearby OBUs in other vehicles or with Roadside Units (RSUs) located along the road. The rapid expansion of wireless sensor networks highlights the need for the development of a comprehensive IoV network that allows vehicles, roadside infrastructure, and the environment to interact in real time. This enables an IoV node to transmit an enormous amount of sensitive data across networks, and cloud computing

---

[☆] This work was supported by the National Natural Science Foundation of China (62376252, 61976196, U22A20102); Key Project of Natural Science Foundation of Zhejiang Province (LZ22F030003) and the Zhejiang Normal University Postdoctoral Research Fund (Grant No. ZC304022985).

[*] Corresponding authors at: School of Computer Science and Technology, Zhejiang Normal University, Jinhua, 321004, China.
E-mail addresses: nigaccna21@zjnu.edu.cn (N.W. Hundera), Umar.aftab@nu.edu.pk (M.U. Aftab), dgeresu@nd.ed (D. Mesfin), dioubifatene@zjnu.edu.cn (F. Dioubi), xhy@zjnu.edu.cn (H. Xu), zxz@zjnu.edu.cn (X. Zhu).

https://doi.org/10.1016/j.vehcom.2024.100811
Received 15 March 2024; Received in revised form 15 April 2024; Accepted 26 May 2024

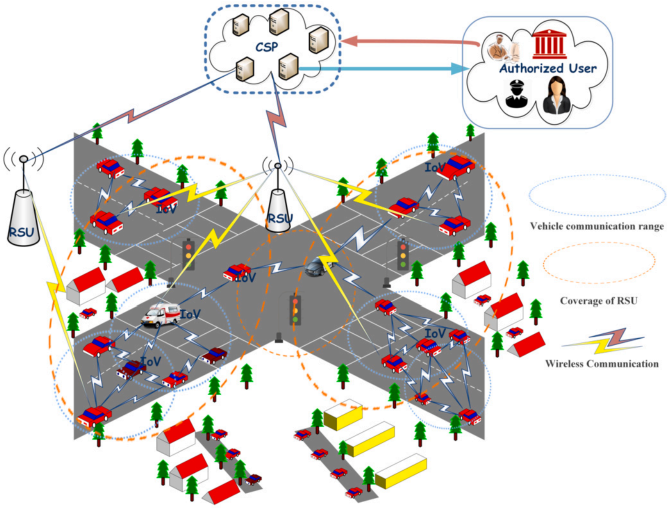

Fig. 1. Standard IoV scenario.

supplies storage services for the data gathered by IoV nodes, allowing the IoV node to access the data via the Internet at any time and from anywhere. Additionally, other users who have been authorized by the IoV node can share the data. Because an IoV node cannot completely rely on a cloud service provider (CSP), it is imperative that the data be encrypted prior to storage. When sharing encrypted data with authorized users, it should be re-encrypted without decryption, ensuring that only they can access the original content.

Public key cryptography offers a promising solution to address these security challenges [8]. A pair of cryptographic keys was employed to establish secure communication and maintain data confidentiality. However, traditional methods are inefficient. For instance, the data owner may need to perform time-consuming tasks such as downloading, decrypting, and re-encrypting data with the recipient's public key [9]. Additionally, sharing the private key with authorized users can pose security risks, as it grants access to the data [10]. Therefore, there is a need for more efficient and secure techniques to share encrypted data in the cloud to enhance the privacy and security of IoV systems. One such technique is proxy re-encryption (PRE) [11]. This technique enables a semi-trusted proxy to alter a ciphertext encrypted under one person's public key into a ciphertext that can be decrypted by another without revealing any information. This feature eliminates the need for data owners to manage and distribute keys for each recipient and to protect the privacy and anonymity of vehicles in the IoV system by preventing direct links between the sender and receiver during communication. Moreover, in this study, the IoV and authorized users belong to different cryptographic infrastructures in a certain area. The three main public key cryptography infrastructures are public key infrastructure (PKI), identity-based cryptography (IBC) and certificateless cryptography (CLC). A PKI uses a certificate authority (CA) to link a user's public key with their identity but faces certificate management challenges such as revocation and verification [12]. IBC, in which public keys are user identities such as email addresses or phone numbers, involves a private key generator (PKG) that generates secret keys, leading to a key escrow issue [13]. CLC uses a key generation center (KGC) for master and partial private keys, and users create their secret keys while avoiding key escrow and certificate management problems [14]. Therefore, CLC is the best choice for the IoV node because it avoids key escrow and public key certificate management problems, whereas IBC, which is free from public key certificate management issues, is ideal for authorized user.

1.1. *Motivation and contribution*

The motivation of this study is to maintain secure communication between an IoV and an authorized user operating within different cryptographic environments. Because the IoV node has limited processing and storage capacity, the scheme employs online and offline approaches to reduce the computational and communication load on the IoV node. The HOOCLS-PRE method is used to ensure secure communication within a cloud-assisted IoV environment. The contributions of this study are as follows:

1. First, an efficient heterogeneous online/offline certificateless signcryption with a proxy re-encryption scheme (HOOCLS-PRE) is proposed. In the proposed scheme, the IoV node operates in the CLC environment, which avoids the certificate management issues of PKI and the key escrow problems of IBC, while the authorized user operates in the IBC environment, which avoids PKI certificate management issues.
2. The proposed scheme splits signcryption into offline and online phases. In the offline phase, most heavy computations are performed without knowledge of the message. During the online phase, when a message is available, only light computations are performed.
3. The proposed HOOCLS-PRE scheme achieves confidentiality, integrity, authentication, and nonrepudiation and protects the privacy and anonymity of vehicles by enabling communication through a proxy, preventing adversaries from linking vehicle identities and locations. Its security has been proven in terms of indistinguishability against adaptive chosen ciphertext attacks (IND-CCA2), existential unforgeability against adaptive chosen message attacks (EUF-CMA) and anonymity under adaptive chosen ciphertext attack (ANON-CCA2) under DBDH and CDH problems in the random oracle model.
4. An extensive evaluation was performed to establish that the proposed scheme outperforms existing schemes in terms of computational cost and communication burden.

1.2. *Related work*

In cloud-assisted IoV systems, vehicles collect crucial data from the surrounding area and exchange them with other vehicles, roadside infrastructure, authorized users, and CSPs [15]. These data can be assessed either on-site or via a cloud server, with subsequent measures taken based on specific requirements. However, the extensive interconnected networks and numerous users in the IoV pose great security and privacy risks [16–18]. To address these challenges, encryption and digital signature cryptographic tools are used. However, traditional methods of signing and encrypting messages can be computationally expensive and result in high communication overhead. To overcome these challenges, the concept of signcryption was introduced by Zheng in 1997 [19]. Signcryption is a cost-effective cryptographic approach that combines digital signatures with public key encryption in one step [19,20]. That is, signcryption can provide nonrepudiation, integrity, confidentiality, and authentication at a lower cost and is useful in a variety of settings, including smart cards, web information systems, and mobile communications, due to its performance advantages. In 2007, Baek et al. [21] first established the security of signcryption using a random oracle model, demonstrating that signcryption can accomplish both encryption and digital signature security. Following this research, many signcryption techniques have been proposed [22,23]. However, all the above schemes use PKI, which involves certificate management, storage, revocation and time, posing significant challenges in developing a stable IoV network [24].

In 1984, Shamir [25] proposed an IBC approach to avoid PKI certificate management issues. In IBC, the user's public key is obtained from their identity data, while PKG produces the secret key. Malone-Lee [26] combined the concept of IBC with signcryption to propose an ID-based signcryption scheme (IBSC). Since then, several efficient IBSC and IBS schemes have been proposed [27–29]. However, these schemes have key escrow problems; compromising the PKG can threaten or destroy system security. To overcome this issue, CLSC was introduced [30], in which the complete private key is split into two parts: the user's partial private key is generated by the KGC, and the user then generates their secret value. This approach resolves the issues associated with PKI and IBC. Since then, many CLSC schemes have been developed to enhance efficiency and reduce computational costs [31–33].

In the cloud environment, traditional methods of sharing encrypted data face challenges in enabling many people to access the same data without a trustworthy third party. For instance, the data owner may need to share a private key with authorized users, which can pose security risks, or may need to perform time-consuming tasks such as downloading, decrypting, and re-encrypting data with the recipient's public key [34,35]. To address these challenges, proxy re-encryption (PRE) techniques have been proposed. PRE is a cryptographic primitive that allows a semitrusted proxy to re-encrypt ciphertext encrypted with one public key into another without revealing any information [36]. PRE can be used for numerous applications, such as managing digital rights, setting up decentralized storage systems, and sending emails. Because of its advantages, numerous schemes for performing PRE have been proposed, from IBPRE schemes [37–40] to IBSC-PRE schemes [41–45]. Liu et al. [46] and Qi et al. [47] designed pairing-free CLIBPSC schemes, but designing such schemes is challenging. When implemented on resource-limited devices, Liu et al. [46] is exposed to public key replacement attacks. In 2017, [49] proposed a CLPSC technique using ECC. Later, Ahene et al. [48] developed an efficient CLSPRE scheme, and recently, Obiri et al. [49] proposed a CLSPRE method to secure crop-related data in the cloud. However, these schemes incur extensive computational and communication overheads and, [50] are vulnerable to public key replacement attacks in the presence of a Type 1 adversary. Moreover, all the aforementioned schemes are homogeneous and unsuitable for heterogeneous communications. Due to the dynamic nature and complexity of the communication environment of IoV systems, different communication terminals may have different security requirements in different cryptography environments. Li et al. [51] proposed two heterogeneous signcryption methods to ensure message exchange between PKI and IBC. Li et al. [52] similarly developed a heterogeneous signcryption scheme that enables communication between CLC to PKI. Subsequently, Omala et al. [53] designed a heterogeneous signcryption method for communication among CLC and IBC settings. However, all of the above schemes lack PRE techniques, hence, the absence of data access control, and do not have the ANON-CCA2 security property. To address these issues, a novel efficient HOOCLS-PRE for a cloud-assisted IoV environment is proposed.

1.3. Organization

The remainder of this paper is organized as follows: The preliminary work is introduced in Section 2. The formal model of HOOCLS-PRE is presented in Section 3. An efficient HOOCLS-PRE scheme is proposed in Section 4. Security and performance analyses are provided in Sections 5 and 6, respectively. Finally, the conclusions are presented in Section 7.

2. Preliminary work

This section provides the notation, bilinear maps, and hardness assumptions.

2.1. Notation

Table 1 outlines the acronyms employed throughout this paper.

2.2. Bilinear maps

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups with the same prime order q. $\mathbb{G}_1$ is an additive group, and $\mathbb{G}_2$ is a multiplicative group. Let P be a generator of $\mathbb{G}_1$. A bilinear pairing is a map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ that satisfies the following requirements:

1. Bilinearity: For all $P, Q \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q^*$, $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$.
2. Nondegeneracy: There are $P, Q \in \mathbb{G}_1$ such that $\hat{e}(P, Q) \neq 1$, where 1 is the $\mathbb{G}_2$ identity element.
3. Computability: $\hat{e}(P, Q)$ is efficiently calculated for all $P, Q \in \mathbb{G}_1$.

2.3. Hardness assumptions

The modified Weil and Tate pairings offer acceptable maps of this type [54]. The security of HOOCLS-PRE relies on the hardness of the subsequent problems.

The parameters $\mathbb{G}_1$, $\mathbb{G}_2$, q, P and $\hat{e}$, are given, similar to the above definitions.

Definition 1. *Decisional Bilinear Diffie-Hellman Problem (DBDHP):* Given a tuple $(P, aP, bP, cP) \in \mathbb{G}_1$ for some $a, b, c \in \mathbb{Z}_q^*$ and $h \in \mathbb{G}_2$, the *DBDHP* is used to determine whether $h = \hat{e}(P, P)^{abc}$.

Definition 2. *Computational Diffie-Hellman Problem (CDHP):* Given $(P, aP, bP) \in \mathbb{G}_1$ for some $a, b \in \mathbb{Z}_q^*$, the *CDHP* in $\mathbb{G}_1$ is used to calculate abP.

3. Formal model of HOOCLS-PRE

This section depicts the network model, the framework, and the security schemes for HOOCLS-PRE.

3.1. Network model

An overview of the network model for HOOCLS-PRE is presented in Fig. 2. The authorized user, the IoV node, the cloud service provider (CSP), and the KGC are the four different types of entities that make up the paradigm. The KGC registers the IoV node and authorized user and generates partial private keys for the CLC and private keys for the IBC users. The IoV nodes can upload data to the CSP in an encrypted format. The CSP, known for its superior processing and storage capabilities, is regarded as a cloud system designed to enhance the reliability of the IoV system. It maintains outsourced encrypted data and acts as a proxy for re-encryption without knowing the data's contents. The IoV nodes can retrieve and verify the integrity of their data. The IoV nodes transmit re-encryption commands to the CSP when an authorized user seeks access to uploaded ciphertext. The CSP re-encrypts the ciphertext and sends it to the authorized user, who decrypts and verifies its authenticity. We assume that the KGC is always trustworthy and cannot be corrupted and that the CSP is honest and curious.

3.2. Framework

The generic HOOCLS-PRE scheme comprises the following twelve algorithms, and it involves the IoV node, identified by ID_A; the delegate (Bob), identified by ID_B; and the authorized user, identified by ID_C.

1. *Setup:* Run by the KGC. A security parameter λ is taken as the input and outputs the master secret key s and the system parameters *params*, which include the master public key P_{pub}. To maintain simplicity, *params* is excluded from the descriptions of the other algorithms.
2. *PPKGen:* This is executed by the *KGC*. It takes s and a user's identity ID_i, where $i \in \{0,1\}^*$ as inputs. It returns the partial private key d_i.

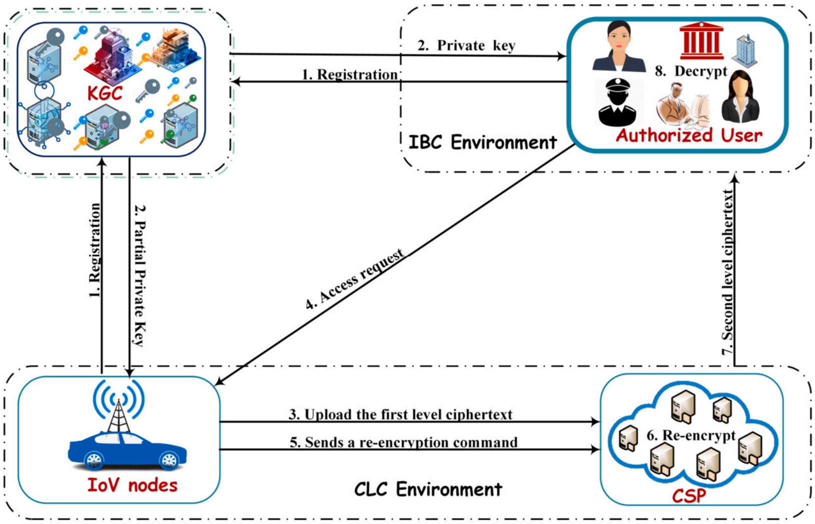

Fig. 2. The HOOCLS-PRE network model.

3. *SV:* This generates a secret value. The user's ID_i is the input, and it outputs a secret value x_i.
4. *SKGen:* Users perform this algorithm. It takes d_i and x_i as inputs and returns a full private key sk_i.
5. *PKGen:* Users perform this algorithm. It takes x_i as inputs, and the output is the public key pk_i.
6. *IB-KE:* It is a key extraction algorithm executed by the *KGC*. It takes a master secret key s and an identity ID_i as inputs and outputs a private key Sk_{ID_i}.
7. *Off-SC:* This is performed by the IoV node. It takes as input the private and public keys of the IoV node S_{k_A}, P_{k_A} and P_{k_B} of the delegate, and returns an offline ciphertext δ.
8. *On-SC:* This is run by the IoV node. The message m, the private and public keys of the IoV node S_{k_A}, P_{k_A} and offline ciphertext δ are used as inputs, and the first-level ciphertext σ_{AB} is output.
9. *DSC*: This is run by the delegate. It takes the first-level ciphertext σ_{AB}, the public key P_{k_A} of the IoV node, and S_{k_B} of the delegate as inputs. It outputs m or $\perp$ if σ_{AB} is not valid between the IoV node and the delegate.
10. *PRKGen:* This is run by the delegate. It takes S_{k_B} of the delegate and Q_{ID_C} of the authorized user as input and outputs a proxy key K_{BC}.
11. *Re-Enc:* This is run by CSP. It takes the first-level ciphertext σ_{AB} and K_{BC} as input and returns a valid second-level ciphertext σ_{AC} for communication between the IoV node and an authorized user.
12. *Dec:* This is a decryption algorithm run by an authorized user that takes as input the second-level ciphertext σ_{AC}, Sk_{ID_C} of the authorized user, P_{k_A} of the IoV node and P_{k_B} of the delegate, respectively. It outputs m or $\perp$ if σ_{AC} is not valid between the IoV node and an authorized user.

The above techniques should meet the consistency condition of HOOCLS-PRE; if $\delta = \textit{Off-SC}(S_{k_A}, P_{k_A}, P_{k_B})$ and $\sigma_{\mathrm{AB}} = \textit{On-SC}(\delta, m, S_{k_A}, P_{k_A})$, then $m = \textit{DSC}(\sigma_{\mathrm{AB}}, P_{k_A}, S_{k_B})$. Additionally, if $K_{\mathrm{BC}} = \textit{PRKGen}(S_{k_B}, Q_{ID_C})$ and $\sigma_{\mathrm{AC}} = \textit{Re-Enc}(\sigma_{\mathrm{AB}}, K_{\mathrm{BC}})$, then $m = \textit{Dec}(\sigma_{\mathrm{AC}}, Sk_{ID_C}, P_{k_A}, P_{k_B})$. Note that the *PPKGen*, *SV*, *SKGen*, *PKGen*, *Off-SC*, *On-SC*, *PRKGen* and *Re-Enc* algorithms are for CLC users, whereas the *IB-KE* and *Dec* algorithms are for IBC users.

3.3. Security schemes

The proposed HOOCLS-PRE scheme ensures confidentiality *(IND-CCA2)*, unforgeability *(EUF-CMA)* and anonymity *(ANON-CCA2)*. The concepts in [52,55] were modified with minor adjustments for HOOCLS-PRE.

Table 1
Acronym and Description.

Acronym	Description
x_i	Secret value of users
d_i	Partial private key for CLC users
sk_i	Private key for CLC users
pk_i	Public key for CLC users
d_{ID_i}	Private key for IBC users
Q_{ID_i}	Public key for IBC users
K_{BC}	Proxy key
CSP	Cloud service provider
ID_A	Identity of the IoV node
ID_B	Identity of the delegate (Bob)
ID_C	Identity of the authorized user
m	Message
P_{pub}	Master public key
s	Master secret key
$\hat{e}$	A bilinear map
$\mathbb{G}_1$	Cyclic additive group
$\mathbb{G}_2$	Cyclic multiplicative group
λ	Security parameter
δ	Offline Ciphertext
σ_{AB}	First-level Ciphertext
σ_{AC}	Second-level Ciphertext

3.3.1. Confidentiality

To assess confidentiality, the game between an adversary $\mathcal{A}$ and a challenger C is examined.
IND-CCA2: C interacts with $\mathcal{A}$.
Initial: C performs the setup with λ and sends *params* to $\mathcal{A}$.
Phase 1: $\mathcal{A}$ conducts a polynomially limited quires.

1. *Partial private key inquiries:* $\mathcal{A}$ chooses $ID_i \in \{0,1\}^*$ and sends ID_i to C. C runs the *PPKGen* and returns d_i to $\mathcal{A}$ as a partial private key.
2. *Private key inquiries:* $\mathcal{A}$ chooses $ID_i \in \{0,1\}^*$. C first computes *SV* and *PPKGen*; then, it performs *SKGen* and sends the full private key sk_i to $\mathcal{A}$.
3. *Public key inquiries:* $\mathcal{A}$ chooses $ID_i \in \{0,1\}^*$. C computes *PKGen* and returns pk_i to $\mathcal{A}$.
4. *Public replacement query:* $\mathcal{A}$ can replace pk_i with a chosen value.
5. *Key extraction inquiries:* $\mathcal{A}$ chooses $ID_i \in \{0,1\}^*$. C computes *IB-KE* and returns the private key Sk_{ID_C} to $\mathcal{A}$.

6. *Proxy key inquiries:* $\mathcal{A}$ selects two identities, ID_i and ID_j. C first runs the *SKGen* and *IB-KE* algorithms to obtain S_{k_i} for the delegator and Q_{ID_j} for the authorized user, respectively. C then runs *PRKGen* and sends a proxy key K_{ij} to $\mathcal{A}$.
7. *Signcrypt inquiries*: $\mathcal{A}$ selects m and two identities, ID_i and ID_j. C first runs the *SKGen* and *PKGen* algorithms to obtain S_{k_i} for the IoV node and P_{k_j} for the delegate. Then, C runs *SC* (m, S_{k_i}, P_{k_j}) and transmits σ_{ij} to $\mathcal{A}$.
8. *De-signcrypt inquiries*: $\mathcal{A}$ selects a ciphertext σ_{ij} and two identities, ID_i and ID_j. C first runs the *SKGen* and *PKGen* algorithms to obtain S_{k_j} for the delegate node and P_{k_i} for the IoV node. Then, C runs *DSC* $(\sigma_{\mathrm{ij}}, S_{k_j}, P_{k_i})$ and transmits the result to $\mathcal{A}$. $\perp$ is returned if σ_{ij} is not a valid ciphertext.
9. *Re-encryption inquiries:* $\mathcal{A}$ picks a ciphertext σ_{ij} and three identities ID_i, ID_j and ID_u. C first runs *SKGen* and *IB-KE* to obtain S_{k_j} for the delegate and Q_{ID_u} for the authorized user and then runs *PRKGen* (Sk_{ID_j}, Q_{ID_u}) to get the proxy key K_{ju}. Finally, C runs *Re-Enc* $(K_{\mathrm{ju}}, \sigma_{\mathrm{ij}})$ and transmits the result σ_{iu} to $\mathcal{A}$.
10. *Decryption queries:* $\mathcal{A}$ picks a ciphertext σ_{iu} and three identities ID_i, ID_j and ID_u. C first runs the *IB-KE* and *PKGen* algorithms to yield Sk_{ID_u} for the authorized user, pk_i for the IoV node, and pk_j for the delegate. Then, C runs *Dec* $(\sigma_{\mathrm{iu}}, Sk_{ID_u}, P_{k_i}, P_{k_i})$ and sends the result to $\mathcal{A}$. If σ_{iu} is invalid, $\perp$ is returned.

Challenge: $\mathcal{A}$ determines when *Phase 1* concludes. $\mathcal{A}$ chooses two messages of equal length, m_0 and m_1, and two identities, ID_A and ID_B, that it wants to challenge. However, the subsequent three constraints are applied.

1. $\mathcal{A}$ has not performed a private key inquiry on ID_B.
2. $\mathcal{A}$ has not inquired about partial private key or public key replacement for ID_B.
3. $\mathcal{A}$ has not executed a proxy key inquiry for (ID_B, ID_u) or a private key query for ID_u.

Then, C picks a random bit $\eta \in \{0, 1\}$ and calculates $\delta = \mathit{Off\text{-}SC}(S_{k_A}, P_{k_A}, P_{k_B})$ and $\sigma_{\mathrm{AB}} = \mathit{On\text{-}SC}(\delta, m_\eta, S_{k_A}, P_{k_A})$. Finally, C sends σ_{AB} to $\mathcal{A}$.
Phase 2: $\mathcal{A}$ performs polynomially limited requests, as in *Phase 1*, with the subsequent restrictions.

1. $\mathcal{A}$ has not performed a private key inquiry for ID_B.
2. $\mathcal{A}$ has not inquired about partial private key or public key replacement for ID_B.
3. $\mathcal{A}$ has not executed proxy key inquiries for (ID_B, ID_u) or private key queries for ID_u.
4. $\mathcal{A}$ cannot make a de-signcryption query for the triple $(\sigma_{\mathrm{AB}}, ID_A, ID_B)$. However, P_{k_A} is replaced after the challenge phase.
5. $\mathcal{A}$ cannot make re-encryption inquiries for the tuple $(\sigma_{\mathrm{AB}}, ID_A, ID_B, ID_u)$ or private key inquiries for the identity ID_u.
6. $\mathcal{A}$ has not run either re-encryption or decryption inquiries on the tuple $(\sigma_{\mathrm{AB}}, ID_A, ID_B, ID_u)$, which return σ_{Au} and $(\sigma_{\mathrm{Au}}, ID_A, ID_B, ID_u)$, respectively, for any identity ID_u.

Guess: $\mathcal{A}$ creates α^*, and if $\alpha^* = \alpha$, then $\mathcal{A}$ wins the game.
$\mathcal{A}$'s advantage is defined as follows:
$\mathrm{Adv}(\mathcal{A}) = |2 \Pr[\alpha^* = \alpha] - 1|$, where $\Pr[\alpha^* = \alpha]$ indicates the probability that $\alpha^* = \alpha$.

Definition 1. The HOOCLS-PRE scheme is $(\varepsilon, t, q_{ppk}, q_{sk}, q_{pk}, q_{pkr}, q_{ke}, q_{kp}, q_{sc}, q_{dsc}, q_{rec}, q_{dec},)-$ *IND-CCA2* secure if no polynomial time adversary $\mathcal{A}$ with runtime t has at least an advantage of ε after most q_{ppk} partial private key inquiries, q_{sk} private key inquiries, q_{pk} public key inquiries, q_{pkr} public key replacement inquiries, q_{ke} key extraction inquiries, q_{kp} proxy key inquiries, q_{sc} signcrypt inquiries, q_{dsc} de-signcrypt inquiries, q_{rec} re-encryption inquiries and q_{dec} decryption inquiries in *IND-CCA2*. See *Section 5* for the security proof.

3.3.2. Unforgeability

Here, because the senders are in a CLC environment, two types of adversaries must be considered for unforgeability: Type I ($\mathcal{F}_{\mathrm{I}}$) and Type II ($\mathcal{F}_{\mathrm{II}}$) [14,52]. $\mathcal{F}_{\mathrm{I}}$ can forge or replace public keys but is unable to access the KGC master key. $\mathcal{F}_{\mathrm{II}}$, a KGC, knows the master secret key but is unable to alter the user's public keys. The security model of HOOCLS-PRE against unforgeability uses two adversary games, EUF-CMA-I and EUF-CMA-II, involving $\mathcal{F}_{\mathrm{I}}$ and $\mathcal{F}_{\mathrm{II}}$ adversaries that play against the challenger (C).
EUF-CMA-I: Here, C plays against $\mathcal{F}_{\mathrm{I}}$.
Initialize: C executes the setup with λ and sends *params* to $\mathcal{F}_{\mathrm{I}}$.
Attack: $\mathcal{F}_{\mathrm{I}}$ executes q_{ppk} inquiries, q_{sk} inquiries, q_{pk} inquiries, q_{ke} inquiries and proxy key inquiries, as in the *IND-CCA2* game. In a signcrypt inquiry, $\mathcal{F}_{\mathrm{I}}$ sends ID_A, ID_B, ID_u and σ_{AB} to C. C first runs *PKGen* to generate the proxy key K_{Bu}. Then, C runs $\delta = \mathit{Off\text{-}SC}(S_{k_A}, P_{k_A}, P_{k_B})$, $\sigma_{\mathrm{AB}} = \mathit{On\text{-}SC}(\delta, m_\eta, S_{k_A}, P_{k_A})$ and $\sigma_{\mathrm{Au}} = \mathit{Re\text{-}Enc}(\sigma_{\mathrm{AB}}, K_{\mathrm{Bu}})$. Finally, C sends σ_{Au} to $\mathcal{F}_{\mathrm{I}}$.
Forgery: $\mathcal{F}_{\mathrm{I}}$ generates a tuple $(ID_A, ID_B, \sigma_{\mathrm{AB}})$ and is successful if the following conditions are met:

1. *DSC* $(\sigma_{\mathrm{AB}}, ID_A, ID_B, S_{k_B}) = m$
2. $\mathcal{F}_{\mathrm{I}}$ is prohibited from extracting private key inquiries on ID_A.
3. $\mathcal{F}_{\mathrm{I}}$ cannot extract proxy key inquiries with $(ID_A,\ ID_u)$ or key extraction inquiries with ID_u.
4. $\mathcal{F}_{\mathrm{I}}$ cannot request partial private key inquiries or public key replacement inquiries on ID_A.
5. $\mathcal{F}_{\mathrm{I}}$ has not made a signcrypt inquiry on (m, ID_A, ID_u) resulting in a ciphertext σ_{AB}, where decrypting with Sk_{ID_u} is considered a potential forgery. Here, ID_u may differ from ID_B.

The $\mathcal{F}_{\mathrm{I}}$ advantage is the probability of success.
EUF-CMA-II: Here, C plays against $\mathcal{F}_2$.
Initialize: C runs the setup with λ and sends *params* to $\mathcal{F}_{\mathrm{II}}$.
Attack: $\mathcal{F}_{\mathrm{II}}$ can make a limited number of adaptive inquiries akin to those in the *IND-CCA2* game, except for *partial private key* and *key extraction inquiries*, because $\mathcal{F}_{\mathrm{II}}$ knows the master private key s.
Forgery: $\mathcal{F}_{\mathrm{II}}$ generates a tuple $(ID_A, ID_B, \sigma_{\mathrm{AB}})$ and is successful if the subsequent conditions are met:

1. *DSC* $(\sigma_{\mathrm{AB}}, ID_A, ID_B, S_{k_B}) = m$
2. $\mathcal{F}_{\mathrm{II}}$ is prohibited from extracting private key inquiries on ID_A.
3. $\mathcal{F}_{\mathrm{II}}$ cannot extract proxy key inquiries with $(ID_A,\ ID_u)$ or key extraction inquiries with ID_u.
4. $\mathcal{F}_{\mathrm{II}}$ has not made a signcrypt inquiry on (m, ID_A, ID_u) resulting in a ciphertext σ_{AB}, where decrypting with Sk_{ID_u} is considered a potential forgery. Here, ID_u may differ from ID_B.

The advantages of $\mathcal{F}_{\mathrm{II}}$ represent success probability.

Definition 2. The HOOCLS-PRE is EUF-CMA secure if no adversary $\mathcal{F}_{\mathrm{I}}$ or $\mathcal{F}_{\mathrm{II}}$ can win the EUF-CMA-I and EUF-CMA-II games with a nonnegligible advantage. HOOCLS-PRE is regarded as EUF-CMA secure when it satisfies both EUF-CMA-I and EUF-CMA-II security. See *Section 5* for the security proof.

3.3.3. Anonymity

For anonymity, the game between an adversary $\mathcal{B}$ and a challenger C is examined.
IND-CCA2: C interacts with $\mathcal{B}$.
Initialize: C performs the setup with λ and sends *params* to $\mathcal{B}$.
Phase 1: $\mathcal{B}$ can make a limited number of adaptive inquiries akin to *IND-CCA2* game.
Challenge: $\mathcal{A}$ determines when *Phase 1* concludes. $\mathcal{A}$ chooses a message m^* and an ID_{A^*} and ID_{B^*} that it wants to challenge. However, these three restrictions hold:

1. $\mathcal{B}$ has not performed a private key inquiry on ID_{A^*}.
2. $\mathcal{B}$ has not inquired about partial private key or public key replacement for ID_{A^*}.
3. $\mathcal{B}$ has not executed a proxy key inquiry for (ID_{B^*}, ID_{u^*}) or a private key inquiry for ID_{u^*}.

Then, C picks a random bit $\eta \in \{0,1\}$ and calculates $\delta =$ *Off-SC*$(S_{k_A}, P_{k_A}, P_{k_B})$ and $\sigma_{\mathrm{AB}} =$ *On-SC*$(\delta, m_\eta, S_{k_A}, P_{k_A})$. Finally, C sends σ_{AB} to $\mathcal{B}$.
Phase 2: $\mathcal{B}$ performs polynomially limited requests, as in *Phase 1*, with the following constraints.

1. $\mathcal{B}$ cannot send a private key inquiry for ID_{A^*}.
2. $\mathcal{B}$ cannot send both partial private key and public key replacement inquiries for ID_{A^*}.
3. $\mathcal{B}$ has not executed a proxy key inquiry for (ID_{B^*}, ID_{u^*}) or a private key inquiry for ID_{u^*}.
4. $\mathcal{B}$ cannot send a de-signcrypt query for the triple $(\sigma_{AB^*}, ID_{A^*}, ID_{B^*})$ unless P_{k_A} is replaced after the challenge phase.
5. $\mathcal{B}$ cannot query both re-encryption keys for $(\sigma_{AB^*}, ID_{A^*}, ID_{B^*}, ID_{u^*})$ and private keys for identity ID_{u^*}.
6. $\mathcal{B}$ has not executed either re-encryption or decryption queries on $(\sigma_{AB^*}, ID_{A^*}, ID_{B^*}, ID_{u^*})$, which return σ_{Au^*} and $(\sigma_{Au^*}, ID_{A^*}, ID_{B^*}, ID_{u^*})$, respectively, for any identity ID_{u^*}.

Guess: $\mathcal{B}$ creates β^*, and if $\beta^* = \beta$, then $\mathcal{B}$ wins the game.
$\mathcal{B}$'s advantage is defined as follows:
$\mathrm{Adv}(\mathcal{B}) = |2\Pr[\beta^* = \beta] - 1|$, where $\Pr[\beta^* = \beta]$ indicates the probability that $\beta^* = \beta$.

Definition 3. The HOOCLS-PRE scheme is $(\varepsilon, t, q_{ppk}, q_{sk}, q_{pk}, q_{pkr}, q_{ke}, q_{kp}, q_{dsc}, q_{dec},)-$ *ANON-CCA2* secure if no polynomial time adversary $\mathcal{B}$ with runtime t has at least an advantage of ε subsequent to numerous q_{ppk} inquiries, q_{sk} inquiries, q_{pk} inquiries, q_{pkr} inquiries, q_{ke} inquiries, q_{kp} inquiries, q_{dsc} inquiries and q_{dec} inquiries in *ANON-CCA2*. See *Section* 5 for the security proof.

4. HOOCLS-PRE SCHEME

This section presents an efficient HOOCLS-PRE scheme. We consider a scenario where an IoV node, identified as ID_A, aims to securely transmit data to a delegate (Bob), identified as ID_B. To enable the same data access by other authorized users, such as Carol with an identity ID_C, the IoV node signcrypts a message m to generate a ciphertext σ_{AB} and delivers it to Bob. Then, a certain proxy re-encrypts the ciphertext σ_{AB} into another ciphertext σ_{AC} so that Carol can decrypt it. In this scheme, the IoV nodes and the delegate operate in the CLC domain, while the authorized user operates in the IBC domain. In addition, the KGC acts as a trusted third party, generating a partial private key for CLC users and a private key for IBC users. The scheme consists of twelve algorithms.

1. *Setup (λ)*: Given a security parameter λ, KGC chooses groups $\mathbb{G}_1$ (additive) and $\mathbb{G}_2$ (multiplicative) with prime order q; a generator P of $\mathbb{G}_1$; a bilinear map $\hat{e}: \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$; and hash functions $H_1: \{0,1\}^* \to \mathbb{G}_1, H_2: \mathbb{G}_1 \times \{0,1\}^n \to \mathbb{Z}_q^*, H_3: \mathbb{G}_2 \to \{0,1\}^n$, $H_4: \mathbb{G}_1^2 \times \{0,1\}^n \times \{0,1\}^* \times \{0,1\}^* \to \mathbb{G}_1$ and $H_5: \mathbb{G}_2 \times \{0,1\}^* \times \{0,1\}^* \to \mathbb{G}_1$, where $\{0,1\}^n$ is the message space. The KGC selects a master secret key $s \in \mathbb{Z}_q^*$ at random and calculates the master public key $P_{pub} = sP$. Finally, the KGC publishes *params* $= \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P, P_{pub}, H_1, H_2, H_3, H_4, H_5\}$ and preserves the master secret key s.
2. *PPKGen*: Given an identity $ID_i, i \in \{0,1\}^*$, the KGC calculates the partial private key as follows:
   (a) Compute $Q_i = H_1(ID_i)$.
   (b) Compute $d_i = sQ_i$ and sends d_i to the user.
3. *SV*: A user with ID_i selects $x_i \in \mathbb{Z}_q^*$ as its secret value.
4. *SKGen*: Given x_i and d_i, a user in CLC sets $S_{k_i} = x_i d_i$ as its full private key.
5. *PKGen*: Given x_i and Q_i, a user in the CLC sets $P_{k_i} = x_i Q_i$ as its public key.
6. *IB-KE*: Given a user's identity ID_i, the KGC computes $Q_{ID_i} = H_1(ID_i)$, $Sk_{ID_i} = sQ_{ID_i}$ and sends Sk_{ID_i} as the private key for IBC users.
7. *Off-SC* : Given the S_{k_A} and P_{k_A} of the IoV node and P_{k_B} of the delegate as input, the IoV node performs the *Off-SC* process as follows:
   (a) Choose $t \in \mathbb{Z}_q^*$.
   (b) Compute $h = tP_{k_A}$.
   (c) Compute $\omega = H_3(\hat{e}(tS_{k_A}, P_{k_B}))$.
   (d) Output $\delta = (h, \omega)$.
8. *On-SC:* Given a message m, the private and public keys of the IoV node S_{k_A}, P_{k_A} and *Off-SC* δ. The algorithm works as follows:
   (a) Compute $r = H_2(h\|m)$.
   (b) Compute $Z = (t + r)S_{k_A}$.
   (c) Compute $C = \omega \oplus m$.
   (d) Compute $U = H_4(h, Z, C, ID_A, ID_B)$.
   (e) Compute $V = tU$.
   (f) Output $\sigma_{\mathrm{AB}} = (h, Z, C, V)$.
   Then, the IoV node sends the first-level ciphertext $\sigma_{\mathrm{AB}} = (h, Z, C, V)$ to the delegate.
9. *DSC*: Given a first-level ciphertext $\sigma_{\mathrm{AB}} = (h, Z, C, V)$, P_{k_A} of the IoV node and S_{k_B} of the delegate, the delegate uses this algorithm to confirm that the ciphertext came from the IoV node. The algorithm performs the following:
   (a) Compute $\omega = H_3(\hat{e}(h, S_{k_B}))$.
   (b) Compute $m = C \oplus \omega$.
   (c) Compute $r = H_2(h\|m)$.
   (d) Check whether $\hat{e}(Z, P) = \hat{e}(h + rP_{k_A}, P_{pub})$. If so, accept m; otherwise, return $\perp$ and reject the ciphertext.
10. *PRKGen*: Given the delegate's (Bob's) private key S_{k_B}, the authorized user's public key Q_{ID_C}, and their identities ID_B and ID_C, this algorithm performs the following:
    (a) Compute $\beta = H_5(\hat{e}(S_{k_B}, Q_{ID_C}), ID_B, ID_C)$.
    (b) Output a proxy key $K_{\mathrm{BC}} = \beta\text{-}S_{k_B}$.
11. *Re-Enc*: Given a first-level ciphertext $\sigma_{\mathrm{AB}} = (h, Z, C, V)$ and a proxy key K_{BC}, the algorithm performs the following:
    (a) Compute $U = H_4(h, Z, C, ID_A, ID_B)$.
    (b) Check whether $\hat{e}(V, P_{k_A}) = \hat{e}(U, h)$. If not, return $\perp$.
    (c) Otherwise, compute $T = H_3(\hat{e}(h, K_{\mathrm{BC}}))$.
    (d) Compute $C' = C \oplus T$.
    (e) Output $\sigma_{\mathrm{AC}} = (h, Z, C', V)$ as the second-level ciphertext and send it to an authorized user.
12. *Dec*: Given a second-level ciphertext $\sigma_{\mathrm{AC}} = (h, Z, C', V)$, the authorized user's private key Sk_{ID_C}, and the public keys of the IoV node and the delegate (Bob), P_{k_A} and P_{k_B} respectively, the algorithm proceeds as follows.
    (a) Compute $\beta' = H_5(\hat{e}(Sk_{ID_C}, P_{k_B}), ID_B, ID_C)$.
    (b) Compute $\omega' = H_3(\hat{e}(h, \beta'))$.
    (c) Compute $m = C' \oplus \omega'$.
    (d) Compute $r = H_2(h\|m)$.
    (e) Check whether $\hat{e}(Z, P) = \hat{e}(h + rP_{k_A}, P_{pub})$. If not, return $\perp$.
    (f) Otherwise, output the message m.
    The following describes how the message decryption process works:

$$
\begin{aligned}
m =& C' \oplus \omega' \\
=& C \oplus T \oplus \omega' \\
=& C \oplus T \oplus H_3(\hat{e}(h, \beta')) \\
=& C \oplus H_3(\hat{e}(h, K_{\mathrm{BC}})) \oplus H_3(\hat{e}(h, \beta')) \\
=& H_3(\hat{e}(tS_{k_A}, P_{k_B})) \oplus H_3(\hat{e}(h, K_{\mathrm{BC}})) \oplus H_3(\hat{e}(h, \beta')) \oplus m \\
=& H_3(\hat{e}(tx_A d_A, P_{k_B})) \oplus H_3(\hat{e}(h, K_{\mathrm{BC}})) \oplus H_3(\hat{e}(h, \beta')) \oplus m
\end{aligned}
$$

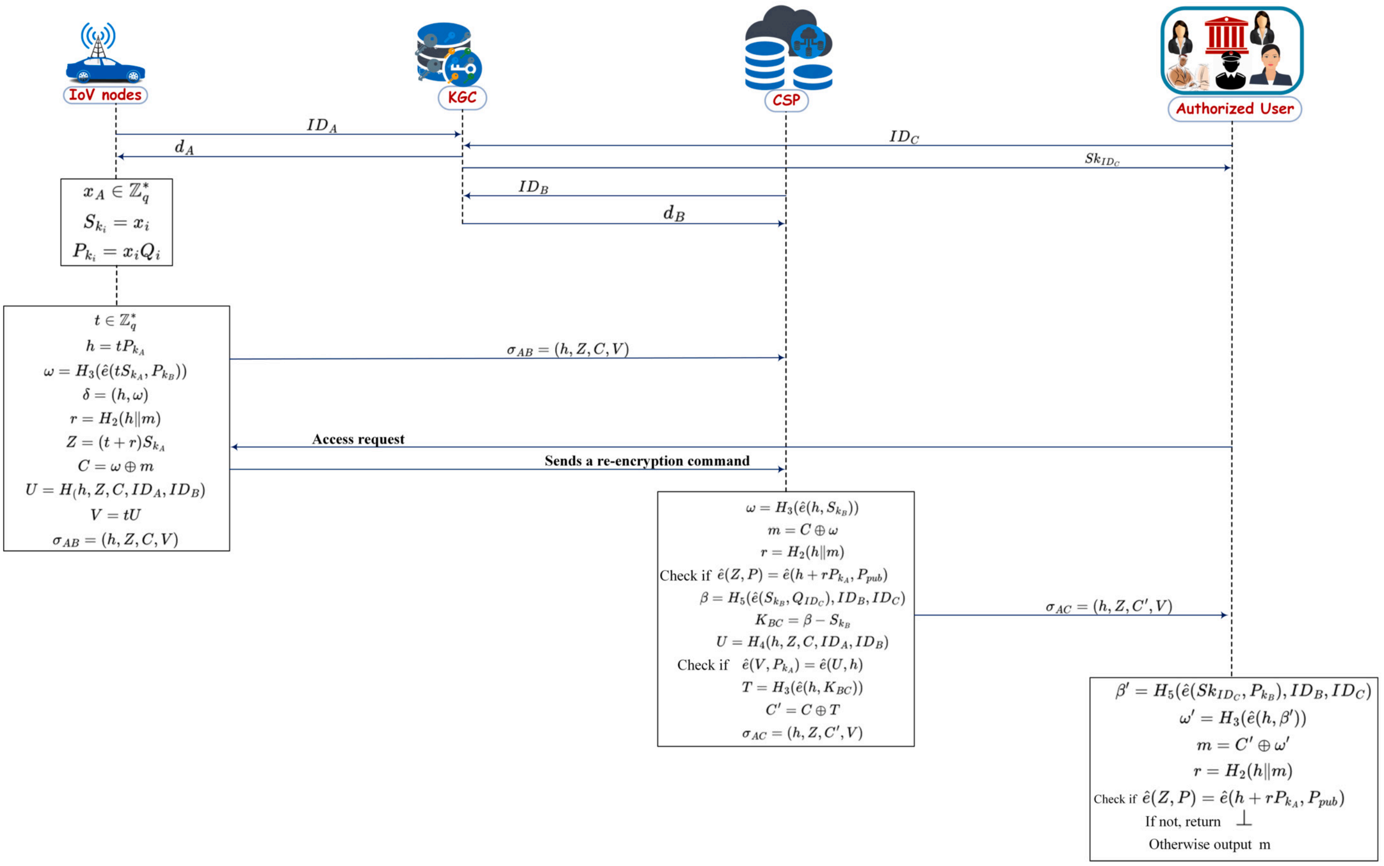

Fig. 3. Efficient HOOCLS-PRE communication.

$$=H_3(\hat{e}(tx_A sQ_A, P_{k_B})) \oplus H_3(\hat{e}(h, K_{BC})) \oplus H_3(\hat{e}(h, \beta')) \oplus m$$

$$=H_3(\hat{e}(tx_A Q_A, sP_{k_B})) \oplus H_3(\hat{e}(h, K_{BC})) \oplus H_3(\hat{e}(h, \beta')) \oplus m$$

$$=H_3(\hat{e}(tP_{k_A}, S_{k_B})) \oplus H_3(\hat{e}(h, K_{BC})) \oplus H_3(\hat{e}(h, \beta')) \oplus m$$

$$=H_3(\hat{e}(h, S_{k_B})) \oplus H_3(\hat{e}(h, K_{BC})) \oplus H_3(\hat{e}(h, \beta')) \oplus m$$

$$=H_3(\hat{e}(h, S_{k_B} + K_{BC})) \oplus H_3(\hat{e}(h, \beta')) \oplus m$$

$$=H_3(\hat{e}(h, S_{k_B} + \beta - S_{k_B})) \oplus H_3(\hat{e}(h, \beta')) \oplus m$$

$$=H_3(\hat{e}(h, \beta)) \oplus H_3(\hat{e}(h, \beta')) \oplus m$$

$$m = m$$

4.1. Correctness analysis

The following formula verifies the equation's validity for both the delegate and an authorized user.

$$\hat{e}(Z, P) = \hat{e}(h + rP_{k_A}, P_{pub})$$

$$= \hat{e}(h, P_{pub})\hat{e}(rP_{k_A}, P_{pub})$$

$$= \hat{e}(tP_{k_A}, sP)\hat{e}(rP_{k_A}, sP)$$

$$= \hat{e}(sP, tx_A Q_A)\hat{e}(sP, rx_A Q_A)$$

$$= \hat{e}(P, tsx_A Q_A)\hat{e}(P, rsx_A Q_A)$$

$$= \hat{e}(P, tS_{k_A})\hat{e}(P, rS_{k_A})$$

$$= \hat{e}(P, (t + r)S_{k_A})$$

$$= \hat{e}(P, Z)$$

Theorems 1, 2, and 3. Here, Fig. 3 shows the efficient HOOCLS-PRE communication.

5. Security analysis

HOOCLS-PRE is proven to ensure confidentiality, unforgeability, and anonymity, as demonstrated in *Theorems 1, 2,* and *3*.

5.1. Confidentiality

Theorem 1. *In the random oracle model, if the adversary $\mathcal{A}$ holds a nonnegligible advantage ε compromising the IND-CCA2 security of the HOOCLS-PRE scheme within time frame t and performs q_{ppk} inquiries, q_{sk} inquiries, q_{pk} inquiries, q_{pkr} inquiries, q_{ke} inquiries, q_{kp} inquiries, q_{sc} inquiries, q_{dsc} inquiries, q_{rec} inquiries, q_{dec} inquiries, and q_{H_i} inquiries to oracles H_i $(i = 1, 2, 3, 4, 5)$, then there is a C that can solve the DBDHP with an advantage*

$$\varepsilon_{dbdh} \geq \left(\frac{\varepsilon}{q_{H_1}}\right)\left(1 - \frac{q_{sc}\left(q_{H_2} + q_{H_3} + q_{H_4}\right)}{2^\lambda}\right)\left(1 - \frac{q_{dsc}}{2^\lambda}\right)$$

at time

$$t' \leq t + O\left(q_{kp} + q_{sc} + q_{rec}q_{H_2} + q_{dsc}q_{H_2} + q_{dec}\right)t_p,$$

where t_p represents the time for a single pairing operation.

Proof. It is illustrated how C utilizes $\mathcal{A}$ as a function to resolve a given scenario (P, aP, bP, cP, h) of the *DBDHP*.
Initial: C executes the *Setup* algorithm with λ and sends *params* along with $P_{pub} = bP$ to $\mathcal{A}$.
Phase 1: C maintains a list L_i (where i ranges from 1 to 5) to simulate the hash oracles H_1, H_2, H_3, H_4 and H_5. Additionally, it keeps a list L_k to store private and public key information, L_{pk} for the proxy key, and a record L_r records challenge identity parameters. The assumptions made are that the queries in H_1 are distinct and that $\mathcal{A}$ requests

the queries in $H_1(ID_i)$ prior to the identity ID_i being utilized in the remaining queries. It is further supposed that $\mathcal{A}$ neither inquiries de-signcryption for ciphertexts from the signcrypt oracle nor decryption for ciphertexts from the re-encryption oracle, but only requests these operations for observed ciphertexts. Furthermore, by employing the irreflexivity assumption [56], it is assumed that the sender and recipient have distinct identities. Initially, all the lists are empty. Upon receiving queries from $\mathcal{A}$, C randomly selects ℓ from the range $(1, \dots, q_{H_1})$ and responds to $\mathcal{A}$'s queries accordingly.

H_1 inquiries: An index i is applied to these queries; it is initialized to 1. $\mathcal{A}$ makes multiple H_1 queries on selected identities. At the ℓ^{th} H_1 inquiry, with $\ell \neq i$, C picks a random $b_i \in \mathbb{Z}_q^*$ and adds the tuple $\left(ID_i, b_i\right)$ to the list L_1, then returns $b_i P$ as the answer and increments i. For the $H_1(ID_i)$ query with $\ell = i$, C answers $H_1(ID_\ell) = b_i P$.

H_2 inquiries: For the $H_2(h\|m)$ query, C first verifies whether the entry is in L_2. Return the previously set value if so. Otherwise, C responds with a random $z_i \in \mathbb{Z}_q^*$ as the answer and adds $(z_i\|h\|m)$ to L_2.

H_3, H_4 and H_5 Queries: The H_3, H_4 and H_5 queries are handled similarly to those for H_2. When $\mathcal{A}$ makes inquiries regarding the hash values, C first reviews the appropriate list. If an entry exists, it returns the previously set value. If not, C generates a random answer and stores both the inquiry and answer in the corresponding list.

Partial private key inquiries: When $\mathcal{A}$ queries q_{ppk} on ID_i, if $ID_i = ID_\ell$, the process fails and stops. Otherwise, C checks L_k for an existing value; if none is found, C proceeds as follows:

1. Compute $Q_i = H_1(\mathrm{ID}_i)$.
2. Compute $d_i = sQ_i$ and add $(\mathrm{ID}_i, d_i, Q_i)$ to L_k.
3. C then sends d_i to $\mathcal{A}_1$.

Private key inquiries: When $\mathcal{A}$ requests q_{sk} inquiry for the identity ID_i, if $\mathrm{ID}_i = \mathrm{ID}_\ell$, the process fails. Otherwise, C checks L_k for an existing tuple $(\mathrm{ID}_i, d_i, Q_i)$; if none is found, C selects $x_i \in \mathbb{Z}_q^*$ randomly, computes $sk_i = x_i d_i$, and adds $(\mathrm{ID}_i, d_i, x_i, Q_i)$ to L_k. Here, d_i is obtained from a previous *partial private key inquiry* with ID_i.

Public key inquiries: $\mathcal{A}$ chooses ID_i and forwards it to C. If the list L_k has a set $(ID_i, x_i, Q_i, P_{k_i})$, then C returns P_{k_i} to $\mathcal{A}$. Otherwise, C selects a random $e_i \in \mathbb{Z}_q^*$, calculates $P_{k_i} = e_i \cdot Q_i$ and adds $(ID_i, e_i, Q_i, P_{k_i})$ to L_k. Then, P_{k_i} is returned to $\mathcal{A}$.

Public key replacement inquiries: For a q_{pkr} inquiry on $(ID_i, x_i, Q_i, P_{k_i})$, C updates the list L_k with tuple $(ID_i, \perp, \perp, P_{k_i})$, where $\perp$ indicates an unknown number.

Key extraction inquiries: $\mathcal{A}_1$ queries the identity ID_i for key extraction inquiry. If $ID_i = ID_\ell$, the process terminates. Otherwise, C checks L_k and returns the existing value; if there is no value, C performs the following:

1. Compute $Q_{ID_i} = H_1(ID_i)$.
2. Compute $Sk_{ID_i} = sQ_{ID_i}$.
3. Add $(ID_i, Q_{ID_i}, Sk_{ID_i})$ to L_k.
4. C then sends Sk_{ID_i} to A_1.

Proxy key inquiries: $\mathcal{A}$ requests a proxy key query for two identities ID_i and ID_j. C first performs private key and key extraction inquiries to obtain S_{k_i} and Q_{ID_j}, respectively. Then, C calculates $\beta = H_5(\hat{e}(S_{k_i}, Q_{ID_j}), ID_i, ID_j)$ and returns $K_{ij} = \beta - S_{k_i}$, then it adds (K_{ij}, β, S_{k_i}) to L_{pk}. Note that the process fails if $i = \ell$.

Signcrypt queries: It is assumed that $\mathcal{A}$ has completed various hash and key generation queries before making a *signcrypt* query. Then, $\mathcal{A}$ selects a message m and two identities ID_i and ID_j. C provides two possible responses:

Case 1: $ID_i \neq ID_\ell$.

1. C performs private key and public key inquiries to obtain S_{k_i} and P_{k_j}.
2. Then, C executes the *signcrypt* $(m, S_{k_i}, P_{k_j}, ID_i, ID_j)$ as usual and returns σ_{ij} to $\mathcal{A}$.

Case 2: $ID_i = ID_\ell$.

1. C retrieves S_{k_i}, P_{k_i}, and P_{k_j} through private key and public key inquiries.
2. It chooses $t \in \mathbb{Z}_q^*$.
3. It computes $h = tP_{k_i}$.
4. It computes $\omega = H_3(\hat{e}(tS_{k_i}, P_{k_j}))$.
5. It computes $r = H_2(h\|m)$.
6. It inserts the tuple $H_2(h\|m)$ into L_2.
7. It computes $Z = (t + r)S_{k_i}$.
8. It computes $C = \omega \oplus m$.
9. It computes $U = H_4(h, Z, C, ID_i, ID_j)$.
10. It inserts the tuple $H_4(h, Z, C, ID_i, ID_j)$ into L_4.
11. It computes $V = tU$.
12. It adds (h, Z, C, V) to list L_r.
13. It returns $\sigma_{ij} = (h, Z, C, V)$ to $\mathcal{A}$.

It is noted that $\mathcal{A}$ is unaware that σ_{ij} is an invalid ciphertext of m for ID_i and ID_j, as $\mathcal{A}$ does not request *de-signcrypt* for σ_{ij}.

De-signcrypt inquiries: When $\mathcal{A}$ requests such a query $\sigma_{ij} = (h, Z, C, V)$ for identities ID_i and ID_j, C provides two possible responses:

Case 1: If $ID_i \neq ID_\ell$

1. C performs private key and public key inquiries to obtain S_{k_j} and P_{k_i}.
2. Then, C executes the *de-signcrypt* algorithm $(\sigma_{ij}, S_{k_j}, P_{k_i})$ as usual and sends the result to $\mathcal{A}$.

Case 2: $ID_i = ID_\ell$.

1. Obtain $\omega = H_3(\hat{e}(h, S_{k_j}))$ by querying H_3.
2. Look over the list L_2 for entries of the form $H_2(h\|m)$, listed by $\mu \in \{1, \dots, q_{H_2}\}$.
3. Compute a message $m = C \oplus \omega$.
4. If $m \neq m_\mu$, proceed to the next element in L_2.
5. Obtain $r = H_2(h\|m)$ by querying H_2.
6. Check whether $\hat{e}(Z, P) = \hat{e}(h + rP_{k_i}, P_{pub})$. If not, move to the next element in L_2.
7. Return the message m_μ to $\mathcal{A}$.
8. If no message is found in L_2, return $\perp$.

Re-encryption query: When $\mathcal{A}$ requests such a query for $\sigma_{ij} = (h, Z, C, V)$ for identities ID_j and ID_u, C provides two possible responses: Case 1: $ID_i \neq ID_\ell$.

1. C performs a proxy key query for ID_j and ID_u to obtain K_{ju}.
2. Executes *Re-Enc* (σ_{ij}, K_{ju}) and sends the outcome to $\mathcal{A}$.

Case 2: $ID_i = ID_\ell$.

1. It obtains $U = H_4(h, Z, C, ID_i, ID_j)$ by sending the query H_4.
2. It obtains $P_{k_i} = x_i Q_i$ from the list L_k.
3. It checks whether $\hat{e}(V, P_{k_i}) = \hat{e}(U, h)$. If not, it returns $\perp$.
4. C executes the *SKGen* and *IB-KE* algorithms to obtain Sk_{ID_j} and Q_{ID_u}.
5. It obtains $\beta = H_5(\hat{e}(S_{k_j}, Q_{ID_u}), ID_j, ID_u)$ by sending the query H_5.
6. It computes $K_{ju} = \beta$-S_{k_j}.
7. It computes $T = H_3(\hat{e}(h, K_{ju}))$ by sending the query H_3.
8. It computes $C' = C \oplus T$.
9. It sends $\sigma_{iu} = (h, Z, C', V)$ to $\mathcal{A}$.

Decryption queries: When $\mathcal{A}$ requests such a query for $\sigma_{iu} = (h, Z, C', V)$, C provides two possible responses:

Case 1: $ID_i \neq ID_\ell$.

1. $\mathcal{C}$ performs private key and public key inquiries to obtain Sk_{ID_u} and P_{k_i}.
2. Then, $\mathcal{C}$ executes the *de-signcrypt* $(\sigma_{iu}, Sk_{ID_u}, P_{k_i})$ as usual and sends the result to $\mathcal{A}$.

Case 2: $ID_i = ID_\ell$.

1. $\mathcal{C}$ performs private key and public key inquiries to obtain Sk_{ID_u} and P_{k_i}.
2. It computes $\beta' = H_5(\hat{e}(Sk_{ID_u}, P_{k_j}), ID_j, ID_u)$.
3. It computes $\omega' = H_3(\hat{e}(h, \beta'))$.
4. It computes a message $m = C' \oplus \omega'$.
5. It obtains $r = H_2(h\|m)$ by making the query H_2.
6. It checks whether $\hat{e}(Z, P) = \hat{e}(h + rP_{k_i}, P_{pub})$. If not, it returns $\perp$.
7. It returns the message m to $\mathcal{A}$.

Challenge: $\mathcal{A}$ selects two plaintexts of identical length m_0 and m_1, as well as two identities ID_A and ID_B to be challenged. If $ID_B \neq ID_\ell$, $\mathcal{C}$ ends. If not, $\mathcal{C}$ random picks bit $\eta \in \{0, 1\}$ and $t \in \mathbb{Z}_q^*$ and initiates $h^* = tP_{k_A}$, $r^* = H_2(h^*\|m_\eta)$, $Z^* = (t + r^*)S_{k_A}$, $\omega^* = H_3(\hat{e}(tS_{k_A}, P_{k_B}))$, $C^* = \omega^* \oplus m_\eta$, $U^* = H_4(h^*, Z^*, C^*, ID_i, ID_j)$, and $V^* = tU^*$. If a tuple exists in L_r, $\mathcal{C}$ chooses a different Z^* until the corresponding (h^*, Z^*, C^*, V^*) is not in any list tuple L_r. Finally, $\mathcal{C}$ returns the ciphertext $\sigma_{AB} = (h^*, Z^*, C^*, V^*)$ to $\mathcal{A}$.
Phase 2: $\mathcal{A}$ makes more polynomially bounded inquiries under *IND-CCA2* constraints, and $\mathcal{C}$ replies as in *Phase 1*.
Guess: $\mathcal{A}$ produces a guess bit α^* and wins if $\alpha^* = \alpha$. If $h = \hat{e}(P, P)^{abc}$, $\mathcal{C}$ returns 1; otherwise, it returns 0, showing that $h \neq \hat{e}(P, P)^{abc}$.
$\mathcal{A}$'s advantage is defined as

$$\mathrm{Adv}_{\mathrm{HOOCLS-PRE}}^{\mathrm{IND-CCA2}}(\mathcal{A}) = |2\Pr[\alpha^* = \alpha] - 1|$$

$$P_1 = |\Pr[\alpha^* = \alpha] - \frac{1}{2}|$$

$$P_1 = \Pr[\alpha^* = \alpha|$$

$$\sigma_{AB} = (m_\eta, S_{k_A}, P_{k_A}, P_{k_B})]$$

$$= \frac{\varepsilon + 1}{2} - \frac{q_{sc}(q_{sc} + q_{H_2})}{2^\lambda})$$

and $P_0 = \Pr[\alpha^* = i|h \in \mathbb{G}_2] = \frac{1}{2}$ for $i = 0, 1$.
Thus, we have

$$\mathrm{Adv}(\mathcal{C}) = |\, P_{a,b,c,\in\mathbb{Z}_p^*,\theta\in\mathbb{G}_2}[1 \leftarrow \mathcal{C}(P, aP, bP, cP, \theta)]$$

$$- P_{a,b,c,\in\mathbb{Z}_p^*}[1 \leftarrow \mathcal{C}(P, aP, bP, cP, \hat{e}(P, P)^{abc})]\,|$$

$$= \frac{|\, P_1 - P_0\,|}{(2^{q_{H1}})^2},$$

$$\varepsilon_{\mathrm{dbdh}} \geq \left(\frac{\varepsilon}{q_{H_1}}\right)\left(1 - \frac{q_{sc}\left(q_{H_2} + q_{H_3} + q_{H_4}\right)}{2^\lambda}\right)\left(1 - \frac{q_{dsc}}{2^\lambda}\right). \quad \square$$

5.2. Unforgeability

Theorem 2. *The HOOCLS-PRE scheme fulfills EUF-CMA security under the CDHP against adversaries $\mathcal{F}_I$ and $\mathcal{F}_{II}$. The EUF-CMA-I and EUF-CMA-II games, described below, demonstrate the security of Theorem 2.*
EUF-CMA-I: *In the random oracle model, if an adversary $\mathcal{F}_I$ has a non-negligible advantage ε in compromising the EUF-CMA-I security of the HOOCLS-PRE scheme within time t and performs q_{ppk} inquiries, q_{sk} inquiries, q_{pk} inquiries, q_{pkr} inquiries, q_{ke} inquiries, q_{kp} inquiries, q_{sc} inquiries, q_{dsc} inquiries, q_{rec} inquiries, q_{dec} inquiries, and q_{H_i} inquiries to oracles H_i $(i = 1, 2, 3, 4, 5)$, then there is a $\mathcal{C}$ that can resolve the CDHP with an advantage*

$$\varepsilon_{cdh} \geq \frac{10(q_{sc} + 1)(q_{sc} + q_{H_3})q_{H_1}}{(2^\lambda - 1)}$$

in a time

$$t' \leq 120686 q_{H_1} q_{H_3} \frac{t + O((q_{kp} + q_{sc} + q_{rec}q_{H_2} + q_{dsc}q_{H_2})t_p)}{\varepsilon(1 - \frac{1}{2^\lambda})}$$

where t_p represents time for a single pairing operation.

Proof. It is illustrated how $\mathcal{C}$ employs $\mathcal{F}_I$ as a function to address a given scenario (P, aP, bP) of the *CDHP*.
Initial: $\mathcal{C}$ executes the *Setup* algorithm with λ and sends *params* along with $P_{pub} = bP$ to $\mathcal{F}_I$.
Attack: $\mathcal{C}$ simulates $\mathcal{F}_I$'s in the *EUF-CMA-I* game, responding to $\mathcal{F}_I$'s inquiries as described in *Theorem 1*. For H_1 inquiries, $\mathcal{C}$ sets the challenge identity $ID_A = ID_\ell$.
Forgery: $\mathcal{F}_I$ outputs a triple $(ID_A, ID_B, \sigma_{AB})$, where $\sigma_{AB} = (h, Z, C, V)$. For an identityless chosen message attack, a generic forged message (ID_A, m) is used. $\mathcal{F}_1'$ generates $((ID_A, m), r, Z)$ and $((ID_A, m), r^*, Z^*)$ by applying the forking lemma with the same commitment but distinct random values r and r^*. The machine $\mathcal{C}$ solves the *CDH* problem by employing $\mathcal{F}_I'$.

1. Through the execution of $\mathcal{F}_1'$, $\mathcal{C}$ generates $(ID_A, m), r, Z)$ and (ID_A, m).
2. It computes $abP = (r - r^*)^{-1}(Z - Z^*)$.
3. It then returns abP as the solution to the *CDH* problem.

If $\mathcal{F}_1$ succeeds within time t with a certain probability, based on the forking lemma [57], the following is true:

$$\varepsilon_{cdh} \geq \frac{10\left(q_{sc} + 1\right)\left(q_{sc} + q_{H_3}\right)q_{H_1}}{(2^\lambda - 1)}$$

$\mathcal{C}$ solves the *CDH* problem within a specific timeframe.

$$t' \leq 120686 q_{H_1} q_{H_3} \frac{t + O(\left(q_{kp} + q_{sc} + q_{rec}q_{H_2} + q_{dsc}q_{H_2}\right)t_p)}{\varepsilon(1 - \frac{1}{2^\lambda})}$$

EUF-CMA-II: *In the random oracle model, if an adversary $\mathcal{F}_{II}$ has a non-negligible advantage ε in compromising the EUF-CMA-II security of the HOOCLS-PRE within a time t and performs q_{sk} private key inquiries, q_{pk} public key inquiries, q_{ke} key extraction inquiries, q_{kp} proxy key inquiries, q_{sc} signcrypt inquiries, q_{dsc} de-signcrypt inquiries, q_{rec} re-encryption inquiries, q_{dec} decryption inquiries, and q_{H_i} inquiries to oracles H_i $(i = 1, 2, 3, 4, 5)$, then there is a $\mathcal{C}$ that can resolve the CDHP with an advantage*

$$\varepsilon_{cdh} \geq \frac{10(q_{sc} + 1)(q_{sc} + q_{H_3})q_{H_1}}{(2^\lambda - 1)}$$

in a time

$$t' \leq 120686 q_{H_1} q_{H_3} \frac{t + O((q_{kp} + q_{sc} + q_{rec}q_{H_2} + q_{dsc}q_{H_2})t_p)}{\varepsilon(1 - \frac{1}{2^\lambda})}$$

where t_p represents time for a single pairing operation. $\square$

Proof. It is illustrated how $\mathcal{C}$ employs $\mathcal{F}_{II}$ as a function to address a given scenario (P, aP, bP) of the *CDHP*.
Initialize: $\mathcal{C}$ executes the *Setup* algorithm with λ and sends *params* along with $P_{pub} = bP$ to $\mathcal{F}_{II}$.
Attack: $\mathcal{C}$ simulates $\mathcal{F}_{II}$'s in the *EUF-CMA-II* game, responding to $\mathcal{F}_{II}$'s inquiries as described in *Theorem 1*, except for q_{ppk} and q_{pkr} inquiries. For H_1 inquiries, $\mathcal{C}$ sets the challenge identity $ID_A = ID_\ell$.
Forgery: $\mathcal{F}_{II}$ outputs a triple $(ID_A^*, ID_B^*, \sigma_{Au}^*)$, where $\sigma_{AB}^* = (h^*, Z^*, C^*, V^*)$. For an identityless chosen message attack, a generic forged message (ID_A^*, m) is used. $\mathcal{F}_1'$ generates $((ID_A^*, m), r, Z)$ and $((ID_A^*, m), r^*, Z^*)$ by applying the forking lemma with the same commitment but distinct random values r and r^*. The machine $\mathcal{C}$ solves the *CDH* problem by employing $\mathcal{F}_{II}'$.

1. By executing $\mathcal{F}'_{II}$, $\mathcal{C}$ generates $(ID^*_A, m), r, Z)$ and (ID^*_A, m).
2. It computes $abP = (r - r^*)^{-1}(Z - Z^*)$.
3. It then returns abP as the solution to the *CDH* problem.

If $\mathcal{F}_{II}$ succeeds within time t with a certain probability, based on the forking lemma [57], the following is true:

$$\varepsilon_{cdh} \geq \frac{10\left(q_{sc}+1\right)\left(q_{sc}+q_{H_3}\right)q_{H_1}}{(2^{\lambda}-1)}$$

$\mathcal{C}$ solves the *CDH* problem within a specific timeframe.

$$t' \leq 120686 q_{H_1} q_{H_3} \frac{t + O(\left(q_{kp} + q_{sc} + q_{rec}q_{H_2} + q_{dsc}q_{H_2}\right)t_p)}{\varepsilon(1-\frac{1}{2^{\lambda}})} \quad \square$$

5.3. Anonymity

Theorem 3. *In the random oracle model, if an adversary $\mathcal{B}$ holds a non-negligible advantage ε in compromising the ANON-CCA2 security of the HOOCLS-PRE within a time frame t and performs q_{ppk} inquiries, q_{sk} inquiries, q_{pk} inquiries, q_{pkr} inquiries, q_{ke} inquiries, q_{kp} inquiries, q_{sc} inquiries, q_{dsc} inquiries, q_{rec} inquiries, q_{dec} inquiries, and q_{H_i} inquiries to oracles H_i* $(i = 1, 2, 3, 4)$*, then an algorithm $\mathcal{C}$ can be created that addresses the DBDHP effectively*

$$\varepsilon_{dbdh} \geq \left(\frac{\varepsilon}{q_{H_1}}\right)\left(1 - \frac{q_{sc}\left(q_{H_2}+q_{H_3}+q_{H_4}\right)}{2^{\lambda}}\right)\left(1-\frac{q_{dsc}}{2^{\lambda}}\right)$$

at time

$$t' \leq t + O\left(q_{kp} + q_{sc} + q_{rec}q_{H_2} + q_{dsc}q_{H_2} + q_{dec}\right)t_p,$$

where t_p represents time for a single pairing operation.

Proof. It is illustrated how $\mathcal{C}$ utilizes $\mathcal{B}$ as a function to a given random scenario (P, aP, bP, cP, h) of the *DBDHP*.
Initialize: $\mathcal{C}$ executes the *Setup* algorithm with λ and sends *params* along with $P_{pub} = bP$ to $\mathcal{B}$.
Phase 1: $\mathcal{B}$ make a polynomially bounded inquiries, similar to *Theorem 1*.
Challenge: $\mathcal{B}$ chooses a plaintext m^* and target identities ID_{A^*} and ID_{u^*}, which it anticipates will be challenged, and forwards them to $\mathcal{C}$. If $ID_{A^*} \neq ID_{\ell}$, $\mathcal{C}$ terminates. Otherwise, $\mathcal{C}$ randomly selects $e \in \{0, 1\}$ and $t \in \mathbb{Z}^*_q$ and yields σ_{Au^*} with new target identities $(ID_{A^*}, ID_{u^*}, ID_{\ell})$ as follows: It sets $h^* = tP_{k_A}$, $r^* = H_2(h^* \| m_\eta)$, $Z^* = (t + r^*)S_{k_A}$, $\omega^* = H_3(\hat{e}(tS_{k_A}, P_{k_B}))$, $C^* = \omega^* \oplus m_\eta$, $U^* = H_4(h^*, Z^*, C^*, ID_i, ID_j)$, and $V^* = tU^*$. If a tuple exists in L_r, $\mathcal{C}$ selects a different Z^* until a tuple (h^*, Z^*, C^*, V^*) is found that is only in list L_r. Finally, $\mathcal{C}$ signcrypts the message to create $\sigma_{Au^*} \leftarrow$ *Signcrypt* $(m_e, ID_{A^*}, S_{k^*_A}, ID_{u^*}, Q_{ID^*_u}, ID_{\ell})$ and sends σ_{Au^*} to $\mathcal{B}$.
Phase 2: $\mathcal{B}$ makes more polynomially bounded inquiries under *IND-CCA2* constraints, except for q_{ppk} and q_{pkr} inquiries. $\mathcal{C}$ replies as in *Phase 1*.
Guess: $\mathcal{B}$ creates β^* and wins if $\beta^* = \beta$. $\mathcal{C}$ returns '1' if $h = \hat{e}(P, P)^{abc}$ and '0' otherwise, indicating $h \neq \hat{e}(P, P)^{abc}$.
$\mathcal{B}$'s advantage is defined as

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{ANON-CCA2}}_{\mathrm{HOOCLS-PRE}}(\mathcal{B}) &= |2\Pr[\beta^* = \beta] - 1| \\
P_1 &= |\Pr[\beta^* = \beta] - \frac{1}{2}| \\
P_1 &= \Pr[\beta^* = \beta| \\
\sigma_{Vu^*} &= SC(m_e, ID_{A^*}, S_{k^*_A}, ID_{u^*}, Q_{ID^*_u}, ID_{\ell}) \\
&= \frac{\varepsilon + 1}{2} - \frac{q_{sc}(q_{sc} + q_{H_2})}{2^{\lambda}})
\end{aligned}$$

and $P_0 = \Pr[\beta^* = i | h \in \mathbb{G}_2] = \frac{1}{2}$ for $i = 0, 1$

Thus, we have

$$\begin{aligned}
\mathrm{Adv}(\mathcal{C}) = &| P_{a,b,c,\in\mathbb{Z}^*_p,\theta\in\mathbb{G}_2}[1 \leftarrow \mathcal{C}(P, aP, bP, cP, \theta)] \\
&- P_{a,b,c,\in\mathbb{Z}^*_p}[1 \leftarrow \mathcal{C}(P, aP, bP, cP, \hat{e}(P, P)^{abc})] | \\
&= \frac{| P_1 - P_0 |}{(2^{q_{H1}})^2}
\end{aligned}$$

$$\epsilon_{gbdh} \geq (\frac{\epsilon}{q_{H_1}})(1 - \frac{q_s(q_{H_2} + q_{H_3} + q_{H_4})}{2^{\lambda}})(1 - \frac{q_u}{2^{\lambda}}) \quad \square$$

6. Performance

In this section, the major computational cost, environment, ciphertext sizes, and security properties of the proposed scheme are evaluated in comparison with those of Hundera et al. [42], Li et al. [44], Wang et al. [45], Ahene et al. [48] and Obiri et al. [49], as presented in Tables 2 and 3. First, a comparison of the computational cost and the environment is given in Table 2. In Table 2, P denotes the pairing operation in $\mathbb{G}_2$, M is the scalar multiplication in $\mathbb{G}_1$, and E is the exponentiation computation in $\mathbb{G}_2$. Table 2 excludes other operations because these three operations consume the longest running time for the entire algorithm [58]. From Table 2, it is clear that the proposed scheme incurs lower computational costs than the schemes [42], [44], [45], and [48] in the most resource-intensive phases, specifically the SC and DSC algorithms. However, it incurs a higher computational cost in the DSC algorithm compared to the scheme [49]. Nevertheless, the latter scheme [49] fails to meet the ANON-CCA2 security properties and incurs higher communication costs than the proposed scheme, which is designed for a homogeneous environment. Regarding the operational environment, the three schemes [42], [44], and [45] operate within the IBC environment, while the two schemes [48] and [49] operate within the CLC environment. However, in a heterogeneous IoV environment, the sender and receiver must be in different cryptosystems. Therefore, a scheme functioning within the same cryptosystem is impractical for use in such environments. The proposed scheme is designed to address this limitation, ensuring practical applicability in heterogeneous IoV environments.

In Table 3, the symbol ✓ indicates that the scheme fulfills the security property, whereas × indicates its absence. For the ciphertext size, $|x|$ represents the bits of x. From Table 3, it is observed that no schemes achieve ANON-CCA2 security. The proposed scheme meets the IND-CCA2, EUF-CMA, and ANON-CCA2 security requirements, thus providing stronger security guarantees for IoV environments. Regarding the ciphertext size, the proposed scheme produces shorter ciphertexts for both the first and second levels than the other two schemes. It has a similar first-level and shorter second-level ciphertext compared to those of Ahene et al. [48], but Ahene et al.'s [48] scheme incurs higher computational costs in running the SC and DSC algorithms and fails to achieve the ANON-CCA2 security property.

The experiment was conducted using Type A pairing with the PBC library [59] on a desktop ONDA B760-VH4 13th Gen equipped with an Intel® Core™ i5-13600KF 3.50 GHz processor, a 24-GB GPU (NVIDIA GeForce RTX 3090), and 64-GB RAM. The PBC library is a free C library for pairing-based cryptography calculations. Type A pairs are made on the curve $y^2 = (x^3 + x) \bmod q$ for a prime $q = 3 \bmod 4$, where the order of $\mathbb{G}_1$ is p. According to [38], the average execution time for a scalar multiplication operation in $\mathbb{G}_1$ is approximately 6.38ms, an exponentiation computation in $\mathbb{G}_2$ takes approximately 11.20ms, and a pairing operation requires approximately 20.01ms.

For ciphertext size, a $|m| = 160$-bit message and an 80-bit AES [60] security level are considered, leading to q size of 512 bits. Thus, a $\mathbb{G}_1$ group element is 1024 bits, which is reduced to 65 bytes with standard compression [61]. $\mathbb{G}_2$ elements are also 1024 bits. Ciphertext sizes are compared across the six schemes as follows:

Table 2
Comparison of computational costs.

Scheme	SC	DSC	PRKGen	Re-Enc	Dec	Environment
Hundera et al. [42]	4M + 1P	2M + 3P	1P	1M + 1P	2M + 3P	IBC
Li et al. [44]	4M + 1P	1M + 5P	1P	1M + 3P	1M + 4P	IBC
Wang et al. [45]	3M + 2P	1M + 4P	1P	1M + 1P	2M + 6P	IBC
Ahene et al. [48]	3M + 1E + 2P	3M + 3P	4M + 1E + 2P	1M	4M + 3P	CLC
Obiri et al. [49]	5M	9M	7M	8M	12M + 1E	CLC
HOOCLS-PRE	2M	1M + 3P	1P	3P	1M + 4P	CLC-IBC

Table 3
Comparison of security and ciphertext size.

Scheme	Security			Ciphertext size	
	IND-CCA2	EUF-CMA	ANON-CCA2	First Level	Second Level
Hundera et al. [42]	✓	✓	×	$3\|\mathbb{G}_1\| + \|m\|$	$3\|\mathbb{G}_1\| + \|\mathbb{G}_2\| + \|m\|$
Li et al. [44]	✓	✓	×	$4\|\mathbb{G}_1\| + \|\mathbb{G}_2\|$	$3\|\mathbb{G}_1\| + \|\mathbb{G}_2\|$
Wang et al. [45]	×	✓	×	$2\|\mathbb{G}_1\| + \|\mathbb{G}_2\| + \|m\|$	$2\|\mathbb{G}_1\| + \|\mathbb{G}_2\| + \|m\|$
Ahene et al. [48]	✓	✓	×	$2\|\mathbb{G}_1\| + \|m\|$	$3\|\mathbb{G}_1\| + \|m\|$
Obiri et al. [49]	✓	✓	×	$2\|\mathbb{G}_1\| + \|\mathbb{Z}_q^*\| + \|m\|$	$3\|\mathbb{G}_1\| + \|\mathbb{Z}_q^*\| + 2\|m\|$
HOOCLS-PRE	✓	✓	✓	$2\|\mathbb{G}_1\| + \|m\|$	$2\|\mathbb{G}_1\| + \|m\|$

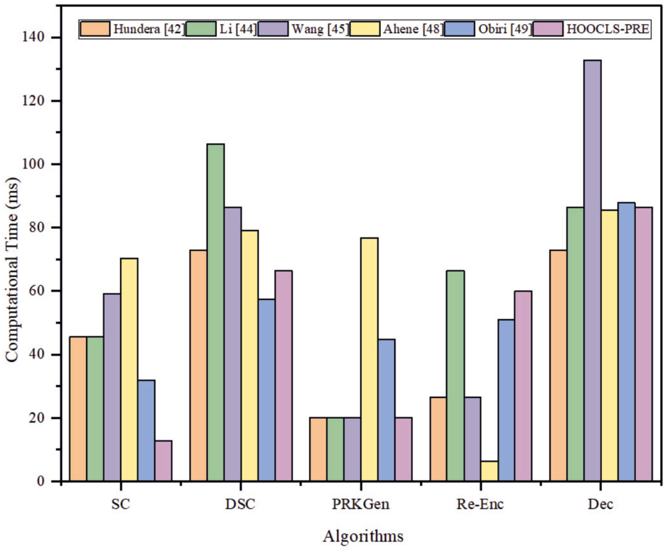

Fig. 4. Comparison of computational cost.

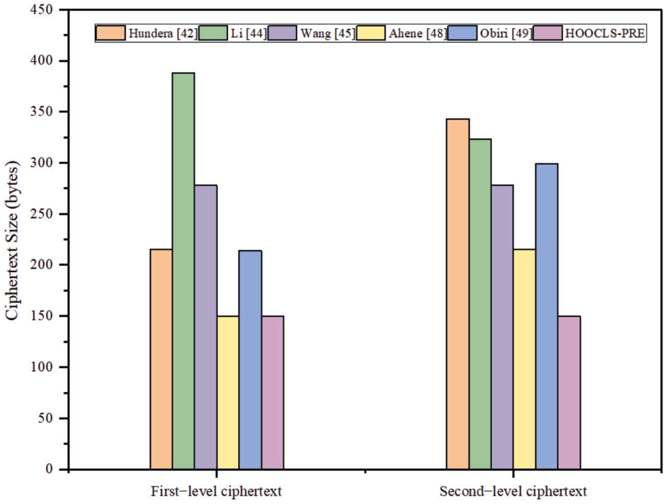

Fig. 5. The ciphertext size of the schemes.

1. In Hundera et al. [42], the first-level ciphertext size is $3|\mathbb{G}_1| + |m| = 3 \times 65 + 20 = 215$ bytes, and the second-level ciphertext size is $3|\mathbb{G}_1| + |\mathbb{G}_2| + |m| = 3 \times 65 + 128 + 20 = 343$ bytes.
2. In Li et al. [44], the first-level ciphertext size is $4|\mathbb{G}_1| + |\mathbb{G}_2| = 4 \times 65 + 128 = 388$ bytes, and the second-level ciphertext size is $3|\mathbb{G}_1| + |\mathbb{G}_2| = 3 \times 65 + 128 = 323$ bytes.
3. In Wang et al. [45], the first-level ciphertext size is $2|\mathbb{G}_1| + |\mathbb{G}_2| + |m| = 2 \times 65 + 128 + 20 = 278$ bytes, and the second-level ciphertext size is $2|\mathbb{G}_1| + |\mathbb{G}_2| + |m| = 2 \times 65 + 128 + 20 = 278$ bytes.
4. In Ahene et al. [48], the first-level ciphertext size is $2|\mathbb{G}_1| + |m| = 2 \times 65 + 20 = 150$ bytes, and the second-level ciphertext size is $3|\mathbb{G}_1| + |m| = 3 \times 65 + 20 = 215$ bytes.
5. In Obiri et al. [49], the first-level ciphertext size is $2|\mathbb{G}_1| + |\mathbb{Z}_q^*| + |m| = 2 \times 65 + 64 + 20 = 214$ bytes, and the second-level ciphertext size is $3|\mathbb{G}_1| + |\mathbb{Z}_q^*| + 2|m| = 3 \times 65 + 64 + 2 \times 20 = 299$ bytes.
6. In our HOOCLS-PRE scheme, the first-level ciphertext size is $2|\mathbb{G}_1| + |m| = 2 \times 65 + 20 = 150$ bytes, and the second-level ciphertext size is $2|\mathbb{G}_1| + |m| = 2 \times 65 + 20 = 150$ bytes.

The ciphertext sizes for the six schemes are summarized in Fig. 5. The proposed scheme requires offline storage of $2|\mathbb{G}_1| + |\mathbb{G}_2| = 2 \times 65 + 128 = 258$ bytes. As illustrated in Fig. 5, The proposed scheme generates shorter ciphertexts for both the first and second levels than other schemes; it has ciphertext sizes similar to those of scheme Ahene et al. [48], yet the latter incurs higher computational costs during the SC and DSC algorithms and does not have the ANON-CCA2 security property. In addition, all referenced schemes, Li et al. [44], Wang et al. [45] and Ahene et al. [48], operate within homogeneous cryptosystems, rendering them less effective in practical heterogeneous IoV environments. For energy consumption, a point multiplication uses 19.1 mJ, exponentiation uses 21.6 mJ, and a pairing uses 45.6 mJ [62,63]. To signcrypt a message, the schemes by Hundera et al. [42], Li et al. [44], Wang et al. [45], Ahene et al. [48], Obiri et al. [49], and the proposed scheme use approximately 122 mJ, 122 mJ, 148.5 mJ, 170.1 mJ, 95.5 mJ, and 38.2 mJ, respectively. The proposed scheme consumes a negligible amount of computational energy. Furthermore, Fig. 4 further demonstrates the superior efficiency of HOOCLS-PRE, providing a clear visual comparison that highlights the performance advantages of the proposed scheme. This comparison clearly shows the capabilities of HOOCLS-PRE in terms of efficiency and effectiveness. This efficiency is due to the division of the signcryption process into offline and online stages, with two-point multiplication and one pairing operation being precomputed offline. Consequently, the online phase of our scheme is exceptionally efficient, requiring only two multiplications. This design allows the proposed scheme to execute the entire signcryption process more rapidly than existing schemes as soon as a message becomes available, highlighting its effectiveness and efficiency in practical IoV applications.

7. Conclusion

In this paper, an efficient and secure access control scheme for cloud-assisted IoV systems is proposed. By utilizing online and offline signcryption techniques, the computational burden on IoV nodes is significantly reduced. Moreover, the proposed scheme achieves confidentiality, integrity, authentication, nonrepudiation, and anonymity. The security of this scheme is also proven in terms of IND-CCA2, EUF-CMA, and ANON-CCA2 under the DBDH and CDH assumptions in the random oracle model. Experimental analysis demonstrates that HOOCLS-PRE surpasses existing schemes in terms of computational cost and communication overhead. Therefore, HOOCLS-PRE is highly appropriate for cloud-assisted IoV environments. Future work will focus on integrating HOOCLS-PRE with 5G and AI to enhance performance and energy efficiency.

CRediT authorship contribution statement

Negalign Wake Hundera: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Muhammad Umar Aftab:** Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Dagmawit Mesfin:** Writing – review & editing, Visualization, Validation, Methodology, Investigation, Formal analysis, Conceptualization. **Fatene Dioubi:** Writing – review & editing, Visualization, Validation, Methodology, Conceptualization. **Huiying Xu:** Supervision, Methodology, Funding acquisition, Conceptualization. **Xinzhong Zhu:** Supervision, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

I would like to declare on behalf of my co-authors that the work described was original research that has not been published previously, and not under consideration for publication elsewhere, in whole or in part. All the authors listed have approved the manuscript that is enclosed.

Data availability

No data was used for the research described in the article.

References

[1] H. Zhang, X. Lu, Vehicle communication network in intelligent transportation system based on Internet of things, Comput. Commun. 160 (2020) 799–806.
[2] M.N. Bhuiyan, M.M. Rahman, M.M. Billah, D. Saha, Internet of things (iot): a review of its enabling technologies in healthcare applications, standards protocols, security, and market opportunities, IEEE Int. Things J. 8 (13) (2021) 10474–10498.
[3] M. Al-Hawawreh, M. Alazab, M.A. Ferrag, M.S. Hossain, Securing the industrial Internet of things against ransomware attacks: a comprehensive analysis of the emerging threat landscape and detection mechanisms, J. Netw. Comput. Appl. 223 (2024) 103809.
[4] Q. Cui, X. Hu, W. Ni, X. Tao, P. Zhang, T. Chen, K.-C. Chen, M. Haenggi, Vehicular mobility patterns and their applications to Internet-of-vehicles: a comprehensive survey, Sci. China Inf. Sci. 65 (11) (2022) 1–42.
[5] P. Rani, R. Sharma, Intelligent transportation system for Internet of vehicles based vehicular networks for smart cities, Comput. Electr. Eng. 105 (2023) 108543.
[6] J. Zhang, H. Fang, H. Zhong, J. Cui, D. He, Blockchain-assisted privacy-preserving traffic route management scheme for fog-based vehicular ad-hoc networks, IEEE Trans. Netw. Serv. Manag. 20 (3) (2023) 2854–2868.
[7] M. Vinodhini, S. Rajkumar, Narrow band of vehicular things communication system using hybrid pelican-beetle swarm optimization approach for intelligent transportation system, Veh. Commun. 45 (2024) 100723.
[8] S. Venkatesan, S. Bhatnagar, I.M.H. Cajo, X.L.G. Cervantes, Efficient public key cryptosystem for wireless network, Neuroquantology 21 (5) (2023) 600–606.
[9] P.K. Premkamal, S.K. Pasupuleti, A.K. Singh, P. Alphonse, Enhanced attribute based access control with secure deduplication for big data storage in cloud, Peer-to-Peer Netw. Appl. 14 (1) (2021) 102–120.
[10] R. Sarma, C. Kumar, F.A. Barbhuiya, Macfi: a multi-authority access control scheme with efficient ciphertext and secret key size for fog-enhanced iot, J. Syst. Archit. 123 (2022) 102347.
[11] I. Keshta, Y. Aoudni, M. Sandhu, A. Singh, P.A. Xalikovich, A. Rizwan, M. Soni, S. Lalar, Blockchain aware proxy re-encryption algorithm-based data sharing scheme, Phys. Commun. 58 (2023) 102048.
[12] S. Khan, F. Luo, Z. Zhang, M.A. Rahim, M. Ahmad, K. Wu, Survey on issues and recent advances in vehicular public-key infrastructure (vpki), IEEE Commun. Surv. Tutor. 24 (3) (2022) 1574–1601.
[13] S. Tanwar, A. Kumar, Secure key issuing scheme in id-based cryptography with revocable id, Inf. Secur. J. 31 (6) (2022) 676–685.
[14] X. Liu, Z. Wang, Y. Ye, F. Li, An efficient and practical certificateless signcryption scheme for wireless body area networks, Comput. Commun. 162 (2020) 169–178.
[15] W. Li, C. Xia, C. Wang, T. Wang, Secure and temporary access delegation with equality test for cloud-assisted IoV, IEEE Trans. Intell. Transp. Syst. 23 (11) (2022) 20187–20201.
[16] M. Obaidat, M. Khodjaeva, J. Holst, M. Ben Zid, Security and privacy challenges in vehicular ad hoc networks, in: Connected Vehicles in the Internet of Things: Concepts, Technologies and Frameworks for the IoV, vol. 9, 2020, pp. 223–251.
[17] B. Hildebrand, M. Baza, T. Salman, S. Tabassum, B. Konatham, F. Amsaad, A. Razaque, A comprehensive review on blockchains for Internet of vehicles: challenges and directions, Comput. Sci. Rev. 48 (2023) 100547.
[18] J. Xu, M. Li, Z. He, T. Anwlnkom, Security and privacy protection communication protocol for Internet of vehicles in smart cities, Comput. Electr. Eng. 109 (2023) 108778.
[19] Y. Zheng, Digital signcryption or how to achieve cost (signature & encryption)≪ cost (signature)+ cost (encryption), in: Advances in Cryptology—CRYPTO'97: 17th Annual International Cryptology Conference Santa Barbara, California, USA August 17–21, 1997 Proceedings 17, vol. 1294, Springer, 1997, pp. 165–179.
[20] A. Elkhalil, R. Elhabob, N. Eltayieb, et al., An efficient signcryption of heterogeneous systems for Internet of vehicles, J. Syst. Archit. 113 (2021) 101885.
[21] J. Baek, R. Steinfeld, Y. Zheng, Formal proofs for the security of signcryption, J. Cryptol. 20 (2) (2007) 203–235.
[22] S. Niu, H. Shao, Y. Su, C. Wang, Efficient heterogeneous signcryption scheme based on edge computing for industrial Internet of things, J. Syst. Archit. 136 (2023) 102836.
[23] X. Yu, W. Zhao, D. Tang, Efficient and provably secure multi-receiver signcryption scheme using implicit certificate in edge computing, J. Syst. Archit. 126 (2022) 102457.
[24] R. Guo, L. Xu, X. Li, Y. Zhang, X. Li, An efficient certificateless ring signcryption scheme with conditional privacy-preserving in vanets, J. Syst. Archit. 129 (2022) 102633.
[25] A. Shamir, Identity-based cryptosystems and signature schemes, in: Advances in Cryptology: Proceedings of CRYPTO 84 4, vol. 196, Springer, 1985, pp. 47–53.
[26] J. Malone-Lee, Identity-based signcryption, Cryptol. ePrint Arch. 1462 (2002) 47–53.
[27] H. Xiong, K.-K.R. Choo, A.V. Vasilakos, Revocable identity-based access control for big data with verifiable outsourced computing, IEEE Trans. Big Data 8 (1) (2017) 1–13.
[28] Y. Zhao, Y. Wang, Y. Liang, H. Yu, Y. Ren, Identity-based broadcast signcryption scheme for vehicular platoon communication, IEEE Trans. Ind. Inform. 19 (6) (2022) 7814–7824.
[29] H. Xiong, Y. Hou, X. Huang, Y. Zhao, Secure message classification services through identity-based signcryption with equality test towards the Internet of vehicles, Veh. Commun. 26 (2020) 100264.
[30] S.S. Al-Riyami, K.G. Paterson, Certificateless Public Key Cryptography, International Conference on the Theory and Application of Cryptology and Information Security, vol. 2894, Springer, 2003, pp. 452–473.
[31] S. Mandal, B. Bera, A.K. Sutrala, A.K. Das, K.-K.R. Choo, Y. Park, Certificateless-signcryption-based three-factor user access control scheme for iot environment, IEEE Int. Things J. 7 (4) (2020) 3184–3197.
[32] G. Xu, J. Dong, C. Ma, J. Liu, U.G.O. Cliff, A certificateless signcryption mechanism based on blockchain for edge computing, IEEE Int. Things J. 10 (14) (2022) 11960–11974.
[33] J. Chen, L. Wang, M. Wen, K. Zhang, K. Chen, Efficient certificateless online/offline signcryption scheme for edge iot devices, IEEE Int. Things J. 9 (11) (2021) 8967–8979.
[34] B. Seth, S. Dalal, V. Jaglan, D.-N. Le, S. Mohan, G. Srivastava, Integrating encryption techniques for secure data storage in the cloud, Trans. Emerg. Telecommun. Technol. 33 (4) (2022) e4108.
[35] X. Yu, D. Tang, W. Zhao, Privacy-preserving cloud-edge collaborative learning without trusted third-party coordinator, J. Cloud Comput. 12 (1) (2023) 19.
[36] F. Luo, S. Al-Kuwari, W. Susilo, D.H. Duong, Chosen-ciphertext secure homomorphic proxy re-encryption, IEEE Trans. Cloud Comput. 10 (4) (2020) 2398–2408.
[37] G. Kan, C. Jin, H. Zhu, Y. Xu, N. Liu, An identity-based proxy re-encryption for data deduplication in cloud, J. Syst. Archit. 121 (2021) 102332.

[38] N.W. Hundera, Q. Mei, H. Xiong, D.M. Geressu, A secure and efficient identity-based proxy signcryption in cloud data sharing, KSII Trans. Int. Inf. Syst. 14 (1) (2020) 455–472.

[39] S. Maiti, S. Misra, P2b: privacy preserving identity-based broadcast proxy re-encryption, IEEE Trans. Veh. Technol. 69 (5) (2020) 5610–5617.

[40] S. Yao, R.V.J. Dayot, H.-J. Kim, I.-H. Ra, A novel revocable and identity-based conditional proxy re-encryption scheme with ciphertext evolution for secure cloud data sharing, IEEE Access 9 (2021) 42801–42816.

[41] S. Hussain, I. Ullah, H. Khattak, M. Adnan, S. Kumari, S.S. Ullah, M.A. Khan, S.J. Khattak, A lightweight and formally secure certificate based signcryption with proxy re-encryption (cbsre) for Internet of things enabled smart grid, IEEE Access 8 (2020) 93230–93248.

[42] N.W. Hundera, C. Jin, D.M. Geressu, M.U. Aftab, O.A. Olanrewaju, H. Xiong, Proxy-based public-key cryptosystem for secure and efficient iot-based cloud data sharing in the smart city, Multimed. Tools Appl. 81 (21) (2022) 29673–29697.

[43] H. Zhu, Y. Wang, C. Wang, X. Cheng, An efficient identity-based proxy signcryption using lattice, Future Gener. Comput. Syst. 117 (2021) 321–327.

[44] F. Li, B. Liu, J. Hong, An efficient signcryption for data access control in cloud computing, Computing 99 (5) (2017) 465–479.

[45] C. Wang, X. Cao, An improved signcryption with proxy re-encryption and its application, in: 2011 Seventh International Conference on Computational Intelligence and Security, IEEE, 2011, pp. 886–890.

[46] Z. Liu, Y. Hu, X. Zhang, H. Ma, Certificateless signcryption scheme in the standard model, Inf. Sci. 180 (3) (2010) 452–464.

[47] Q. Yanfeng, T. Chunming, L. Yu, X. Maozhi, G. Baoan, Certificateless proxy identity-based signcryption scheme without bilinear pairings, China Commun. 10 (11) (2013) 37–41.

[48] E. Ahene, Z. Qin, A.K. Adusei, F. Li, Efficient signcryption with proxy re-encryption and its application in smart grid, IEEE Int. Things J. 6 (6) (2019) 9722–9737.

[49] I.A. Obiri, A.A. Addobea, E. Affum, J. Ankamah, A.K. Kwansah Ansah, A certificateless signcryption with proxy-encryption for securing agricultural data in the cloud, J. Comput. Secur. 32 (2) (2023) 1–39.

[50] T. Bhatia, A.K. Verma, Cryptanalysis and improvement of certificateless proxy signcryption scheme for e-prescription system in mobile cloud computing, Ann. Telecommun. 72 (9) (2017) 563–576.

[51] F. Li, H. Zhang, T. Takagi, Efficient signcryption for heterogeneous systems, IEEE Syst. J. 7 (3) (2013) 420–429.

[52] F. Li, Y. Han, C. Jin, Practical signcryption for secure communication of wireless sensor networks, Wirel. Pers. Commun. 89 (4) (2016) 1391–1412.

[53] A.A. Omala, A.S. Mbandu, K.D. Mutiria, C. Jin, F. Li, Provably secure heterogeneous access control scheme for wireless body area network, J. Med. Syst. 42 (6) (2018) 1–14.

[54] D. Boneh, M. Franklin, Identity-Based Encryption from the Weil Pairing, Annual International Cryptology Conference, vol. 2139, Springer, 2001, pp. 213–229.

[55] F. Li, Y. Han, C. Jin, Certificateless online/offline signcryption for the Internet of things, Wirel. Netw. 23 (1) (2017) 145–158.

[56] V.S. Naresh, S. Reddi, S. Kumari, V.D. Allavarpu, S. Kumar, M.-H. Yang, Practical identity based online/off-line signcryption scheme for secure communication in Internet of things, IEEE Access 9 (2021) 21267–21278.

[57] J.C. Choon, J. Hee Cheon, An identity-based signature from gap Diffie-Hellman groups, in: Public Key Cryptography—PKC 2003: 6th International Workshop on Practice and Theory in Public Key Cryptography Miami, FL, USA, January 6–8, 2003 Proceedings 6, vol. 2567, Springer, 2002, pp. 18–30.

[58] Y. Zhou, L. Zhao, Y. Jin, F. Li, Backdoor-resistant identity-based proxy re-encryption for cloud-assisted wireless body area networks, Inf. Sci. 604 (2022) 80–96.

[59] B. Lynn, Pbc library-pairing-based cryptography, http://crypto.stanford.edu/pbc/, 2007.

[60] J. Daemen, V. Rijmen, J. Daemen, V. Rijmen, Related block ciphers, in: The Design of Rijndael: AES—The Advanced Encryption Standard, 2002, pp. 161–173.

[61] K.-A. Shim, An efficient conditional privacy-preserving authentication scheme for vehicular sensor networks, IEEE Trans. Veh. Technol. 61 (4) (2012) 1874–1883.

[62] F. Li, P. Xiong, Practical secure communication for integrating wireless sensor networks into the Internet of things, IEEE Sens. J. 13 (10) (2013) 3677–3684.

[63] A.A. Omala, N. Robert, F. Li, A provably-secure transmission scheme for wireless body area networks, J. Med. Syst. 42 (6) (2016) 1–14.