mathematics

MDPI

Article

An Attribute-Based End-to-End Policy-Controlled Signcryption Scheme for Secure Group Chat Communication

Feng Yu [1,2], Linghui Meng [1,2,*], Xianxian Li [1,2], Daicen Jiang [3], Weidong Zhu [1,2] and Zhihua Zeng [4]

[1] Key Laboratory of Education Blockchain and Intelligent Technology, Ministry of Education, Guangxi Normal University, Guilin 541004, China
[2] Guangxi Key Lab of Multi-Source Information Mining and Security, Guangxi Normal University, Guilin 541004, China
[3] Southern Power Grid Supply Chain (Guangxi) Co., Ltd., Guangzhou 510530, China
[4] State Key Laboratoary of Nuclear Power Safefy Monitioring Technology and Equipment, China Nuclerar Power Engineering Co., Ltd., Shenzhen 518172, China
* Correspondence: linghuimeng2023@163.com

Abstract: Secure instant communication is an important topic of information security. A group chat is a highly convenient mode of instant communication. Increasingly, companies are adopting group chats as a daily office communication tool. However, a large volume of messages in group chat communication can lead to message overload, causing group members to miss important information. Additionally, the communication operator's server may engage in the unreliable behavior of stealing information from the group chat. To address these issues, this paper proposes an attribute-based end-to-end policy-controlled signcryption scheme, aimed at establishing a secure and user-friendly group chat communication mode. By using the linear secret sharing scheme (LSSS) with strong expressive power to construct the access structure in the signcryption technology, the sender can precisely control the recipients of the group chat information to avoid message overload. To minimize computational cost, a signcryption step with constant computational overhead is designed. Additionally, a message-sending mechanism combining "signcryption + encryption" is employed to prevent the operator server from maliciously stealing group chat information. Rigorous analysis shows that PCE-EtoE can resist adaptive chosen-ciphertext attacks under the standard model. Simulation results demonstrate that our theoretical derivation is correct, and that the PCE-EtoE scheme outperforms existing schemes in terms of computational cost, making it suitable for group chat communication.

Keywords: policy-controlled signcryption; end-to-end group chat communication; attributed-based signcryption; instant communication; data security

MSC: 94A60; 68P25; 68M10; 94A62

Citation: Yu, F.; Meng, L.; Li, X.; Jiang, D.; Zhu, W.; Zeng, Z. An Attribute-Based End-to-End Policy-Controlled Signcryption Scheme for Secure Group Chat Communication. *Mathematics* **2024**, *12*, 2906. https://doi.org/10.3390/math12182906

Academic Editor: Antanas Cenys

Received: 14 August 2024
Revised: 12 September 2024
Accepted: 17 September 2024
Published: 18 September 2024

1. Introduction

Information security [1,2] has become increasingly vital as digital communication platforms proliferate. Instant messaging apps, such as WeChat, QQ, WhatsApp [3], and Telegram [4], each have more than one billion active users. They have gradually replaced the traditional SMS service, providing users with various conveniences in their daily life and work, including personal messaging and group chat communication. These apps not only meet people's basic communication needs, but also extend to social and business functionalities [5]. Most companies conduct their businesses through group chats. With hundreds of millions of messages exchanged every day, users are increasingly concerned about the security and ease of use in group chat communication. However, there are the following problems in group chat communication: (1) whether the source of the message is true, and whether the message is confidential [6]; (2) a large number of group messages may cause information overload, leading users to miss critical information in the group; and

(3) the communication operator of the group chat communication may extract information, resulting in information leakage within the group [7]. For companies, this may lead to the leakage of commercial secrets and cause significant losses [8].

At present, there are two main types of secure instant communication [9]: end-to-server encrypted communication and End-to-End encrypted communication. Figure 1a shows the end-to-server encrypted communication pattern: the sender encrypts the message using the shared key and then sends it to the server. After the server receives the ciphertext message, it decrypts it using the shared key and obtains the plaintext after determining the source of the message. Then, it encrypts the message using the shared key of the server and the receiver and transmits the ciphertext of the message to the receiver. Figure 1b shows the end-to-end encrypted communication mode. The sender performs double-layer encryption, and the server decrypts the source to obtain the first-level ciphertext, which is then encrypted, and then this ciphertext message is transmitted to the receiver. The biggest problem with end-to-server encrypted communication is that it relies too much on the operator's server for security. Communication operators can obtain the plaintext of the message, but dishonest communication operators may extract keywords in the communication and resell the data for profit, which poses commercial threats to companies that rely on group chat communication to build workflows. The mode of end-to-end encrypted communication uses a double-layer encryption method, so that the server cannot access the plaintext of the message and can only forward the message. However, only encrypting the message cannot guarantee its authenticity. Signcryption, as a technology that can encrypt and sign messages in the same logical step, can ensure the authenticity and confidentiality of data simultaneously. We consider signcryption as the first layer of encryption in end-to-end encrypted communication, which differs from the mere application of two layers of encryption, and provides the receiver with the ability to verify the authenticity of the message [10,11].

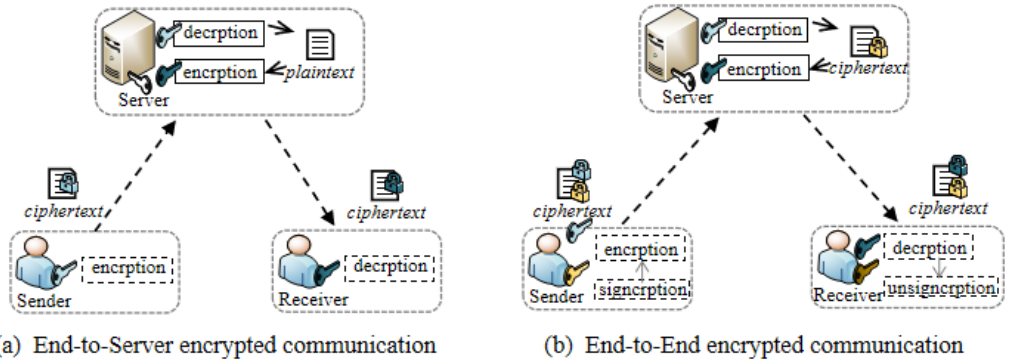


Figure 1. Comparison of end-to-server encrypted communication and end-to-end encrypted communication.

Group chat communication [12], from the sender's perspective, follows a one-to-many communication mode. Attribute-based signcryption technology can not only realize one-to-many encrypted transmission and provide access control, but also allow users to verify that the attributes meet the policy to determine the authenticity of the data. Currently, in most group chats, all group members generally receive a large amount of information, but only some of this information is directly related to their specific roles, responsibilities, or current tasks. The ubiquity and non-differentiation of this information flow may make it difficult for users to filter out the information that is truly important to them, thus distracting users, reducing work efficiency, and increasing the risk of missing critical information. The access control function can mitigate this message overload problem. However, attribute-based signcryption technology verifies and decrypts the ciphertext using a decryption key, and the key distribution is carried out by a third party. If the third party is unreliable, decryption will fail. Susilo [13] proposed a policy-controlled signcryption scheme, which is still an

specially attribute-based signcryption scheme, but the ciphertext is verified by credentials, making it more flexible than using a key. Users can verify the credentials to ensure their correctness. However, directly applying the policy-controlled signcryption proposed by Susilo [13] to group chat communication results in the problem of high computational cost. Attribute-based cryptographic schemes often require high computational cost, especially as the computational cost for the sender increases linearly with the number of attributes in the policy. When the strategy is too complex, it will impose a significant computational burden on the sender in group chat communication. The previous discussion focused on text message protection in group chat communication. Image message protection is also important in group chat communication. Many researchers [14–18] used many hash methods to protect image information, and this scheme also protects image information in group chat communication. By converting the picture to a binary file and applying policy-controlled signcryption technology , the security protection of image information in group chat communication was completed, ensuring that only recipients in the group who met the sender's requirements could receive the image information.

In order to solve the above problems, in this paper, we propose an attribute-based end-to-end policy-controlled signcryption scheme for group chat communication, which has constant signcryption computation consumption. The scheme constructs a strong expressive access policy to ensure that only recipients conforming to the access policy can receive information in the group, alleviating the problem of message overload in group chat communication. The scheme employs the ECDH key agreement protocol and double-layer ciphertext protection, ensuring that the operator server cannot access the plaintext of the group chat information, thereby preventing information leakage to the operator.

Our contributions can be listed as follows:

- **Lightweight**: We propose a lightweight signcryption scheme for communication. The computational cost for the sender in the group chat is constant, and the sender only requires three power operations and one linear mapping operation during the signcryption process. Although the computational cost for the receiver still increases with the number of attributes, it is lower than that of existing schemes. In the experiment, end devices with varying computing resources were used for testing, and the computing times for both the signcryption and un-signcryption operations were within a reasonable range.
- **Strong expressive capability**: The signcryption scheme for communication (PCE-EtoE) that we propose demonstrates strong expressive capability. Compared with the monotone Boolean function access structure in PCEA [13], the PCE-EtoE scheme uses LSSS to design an access structure with strong expressive power, capable of describing complex access policies including "and", "or" and other predicates.
- **Mitigate message overload**: Our proposed PCE-EtoE scheme effectively prevents users from receiving unnecessary information that is irrelevant to their needs. In PCE-EtoE, the sender in group communication can construct a complex and fine-grained access policy, ensuring that only receivers conforming to the policy can receive the group chat information.This scheme not only ensures controllable message flow but also enhances the efficiency of the messages received, thereby avoiding message overload and preventing users from missing key information.
- **Security**: In the defined security model, PCE-EtoE ensures message confidentiality through indiscernibility under chosen plaintext attacks (IND-CPA). Additionally, the scheme employs double encryption technology (encryption + signcryption), which prevents the communication operator from decrypting the second-level ciphertext, thereby ensuring that the communication operator cannot steal the content of the group chat communications.

In the next two sections, we will provide a brief review of the related work and preliminaries. The security model of the PCE-EtoE scheme is presented in Section 3. Section 4 describes the proposed PCE-EtoE scheme. The security and functional analysis of

PCE-EtoE is provided in Section 5. Section 6 offers a comparative analysis of the theoretical performance and practical performance of PCE-EtoE. The conclusion is presented in the final section.

2. Related Work

The proposed PCS-EtoE mainly involves attribute-based signcryption and end-to-end encrypted communication. Therefore, in this section, we review related work in these areas.

Hong et al. [19] proposed a key policy KP-ABCS scheme that can perform outsourced decryption and introduced key updates to prevent key leakage. However, due to the lack of public verification and high implementation costs, the scheme is not practical. Rao and Dutta [20] proposed the first KP-ABSC scheme with constant-size ciphertext, which achieves constant ciphertext size and reduces storage costs to some extent, but its computational overhead is too large. Yu and Cao [21] proposed attribute-based signcryption with a hybrid access policy, which combines key policy (KP) and ciphertext policy (CP). However, the threshold value is used in expressions, and the expression ability is poor. Xu et al. [22] proposed CP-ABSC supporting multi-authority, which can protect users' attribute privacy. Although outsourced decryption is used in the scheme, the overall overhead of the scheme is too large, and users may still incur excessive computational overhead due to the large number of attributes during signcryption. Zhao et al. [23] proposed an efficient multi-authority attribute-based signcryption scheme, which realizes multi-authority access control and protects the privacy of the signcryption. Even if the ciphertext is not associated with the signing key, the information of the data owner cannot be obtained from the ciphertext. However, the computational cost is still linear with the number of attributes. Wang et al. [24] proposed an attribute-based signcryption scheme with a ciphertext policy and declaration predicate mechanism (CP2-ABSC), but it still incurs too much computational overhead and has poor practicability. Susilo et al.'s [13] PCEA is a special ABSC that can use credentials for un-signcryption, solving user decryption failures caused by unreliable operations of a TA or KCGS (Key and Credential Generation Server) in some cases. However, the computational overhead is too large and there are too many mapping operations, making it unsuitable for direct application in group chat communication scenarios. The schemes proposed in [21,22,24] over-rely on outsourced computing services provided by operators, which cannot prevent dishonest behavior in outsourced decryption. In the scenario of end-to-end group chat communication using attribute-based signcryption to protect ciphertext, over-reliance on outsourced computing provided by operators threatens the confidentiality of information sent by users. It also increases the computational cost for the communication operator. In the above-mentioned schemes, the computational cost of signcryption increases linearly with the number of attributes or authorities to varying degrees. Since real-time messages need to be exchanged frequently in group chat communication, the above schemes are not suitable.

In end-to-end encrypted communication, Cohn-Gordon et al. [25] designed an asynchronous key exchange protocol to ensure that all members can maintain end-to-end encrypted communication without overlapping online time. Gupta et al. [26] proposed an end-to-end encryption layer based on ciphertext-policy attribute-based encryption (CP-ABE) for data confidentiality and integrity in Message Queuing Telemetry Transport (MQTT). However, additional signature operations are required to enable the receiver to determine the origin of the data. Dhinesh et al. [27] analyzed and discussed end-to-end encryption (E2EE) implementations of various messaging applications. All these messaging applications use the encrypt-then-sign approach, which is significantly less computationally efficient than signcryption.

Table 1 summarizes the features of representative ABSC schemes and the proposed PCE-EtoE in terms of access structure and functionality. Compared with existing schemes, our PCE-EtoE supports more key features. In particular, PCE-EtoE supports the control of complex message flow, does not rely on outsourced computation, and is resistant to operator data theft. Without the help of outsourced computation, the cost of signcryption is constant,

which is highly practical in group chat communication. In contrast, the computational cost of signcryption in other schemes increases linearly with the number of attributes.Later, we will show that in the secure model, the PCE-EtoE satisfies IND-CPA under the Decisional Bilinear Diffie–Hellman (DBDH) problem.

Table 1. Comparison of features for the existing ABSC scheme and the proposed PCE-EtoE.

Scheme	Access Structure	Functionality				
		CM	LC	CC	NR	RO
[24]	Access Tree	×	×	×	✓	×
[19]	MBF	×	×	×	×	×
[20]	LSSS	✓	×	×	×	×
[21]	MBF and Threshold	×	×	×	✓	×
[22]	MBF	×	×	×	×	×
[23]	LSSS	✓	×	×	✓	×
[13]	MBF	×	×	×	✓	×
PCE-EtoE	LSSS	✓	✓	✓	✓	✓

Note: CM: complex message flow control; LC: lightweight computational cost; CC: constant computational cost of signcryption; NR: no reliance on outsourced computation; RO: resist operator stealing information; MBF: monotone Boolean function.

3. Preliminaries

3.1. Symmetric Bilinear Mapping

Let ϕ be the group generation algorithm. When given the security parameter λ, it outputs the parameters $(G_1, p, G_T, \hat{e}, g)$, where p represents a large prime number, G_1 and G_T are two cyclic groups of order p, g is the generator of G_1, and $\hat{e} : G_1 \times G_1 \to G_T$ denotes a bilinear map if and only if the following three conditions are met:

- **Bilinearity:** $\forall(u, v \in G_1, a, b \in Z_p)$: $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$;
- **Non-degeneracy:** $\exists u, v \in G_1$: $\hat{e}(u, v) \neq 1$;
- **Computability:** $\exists \hat{e} : \forall u, v \in G_1; \hat{e}(u, v) \in G_T$.

3.2. Decisional Bilinear Diffie–Hellman Assumption

In this section, we introduce the DBDH (Decisional Bilinear Diffie–Hellman) assumption, which is the decisional version of the BDH problem and is extensively utilized in designing cryptographic protocols. The details are as follows:

1. Given cyclic groups G_1 and G_2 of order p as prime numbers.
2. Randomly select the generator $g \in G_1$ and the random numbers $c_1, c_2, c_3 \in Z_p$.
3. $g, g^{c_1}, g^{c_2}, g^{c_3}, \hat{e}(g, g)^{c_1 c_2 c_3}$ and $T \in G_2$ to send A.

The core of the DBDH problem is to determine whether T is equal to $e(g, g)^{c_1 c_2 c_3}$, if so, A outputs 1; otherwise, output 0.

The DBDH assumption states that if no polynomial time algorithm can solve the DBDH problem with a non-negligible advantage, then the DBDH assumption holds in the G_1 and G_2 groups.

4. Security Model

The security model of the proposed scheme is defined based on a game between the challenger and the attacker, described as follows:

(1) **Initialization:** The challenger runs the Setup, KCGSKeyGen, and UserKeyGen algorithms, generating PK_{KCGS}, SK_{KCGS}, $PK_{U_{id}}$, $SK_{U_{id}}$ and system parameters *param*, and provides PK_{KCGS} and SK_{KCGS} to the attacker. The attacker chooses a set of access policies *POL* to send to the challenger.

(2) **Phase 1:** The attacker can request to query the credential of any ciphertext not utilized in the un-signcryption challenge.
(3) **Challenge:** The attacker submits two randomly chosen messages, m_1 and m_2, of equal length to the challenger. The challenger randomly selects $\delta \in \{0,1\}$ and performs the signcryption operation on the message m_δ according to the access policy POL submitted by the attacker.
(4) **Phase 2:** The same as phase 1,the attacker can request to query the credential of any ciphertext not utilized in the un-signcryption challenge.
(5) **Guess:** The attacker outputs the guess $\hat{\delta}$. Thus, the advantage of the attacker in this game is defined as $Pr[\delta = \hat{\delta}] - \frac{1}{2}$.

Definition 1. *If the attacker cannot win the game with a non-negligible advantage in polynomial time, then the PCE-EtoE scheme is indistinguishable under chosen plaintext attacks (IND-CPA) .*

5. The Proposed PCE-EtoE Scheme

In this section, we provide a detailed description of the proposed PCE-EtoE scheme. To fully understand the proposed PCE-EtoE scheme, we present its framework in Figure 2, which involves four participating entities as follows:

- **Sender:** Complete the transmission of messages in the group chat. Control the specific flow of information in the group chat to ensure that the flow of dissemination of group chat messages is effectively managed.
- **Receiver:** Only receive relevant and valid information in the group chat.
- **Key and Credential Generation Server (KCGS):** KCGS is responsible for generating public parameters and creating credentials and keys for users (sender and receiver). KCGS is a semi-honest server, which may tamper with part of the content in the credential.
- **Communication Operator's Server (COS):** It is responsible for generating the elliptic curve parameters, completing the key agreement for the secondary encryption key between the users and COS, and forwarding the ciphertext information.

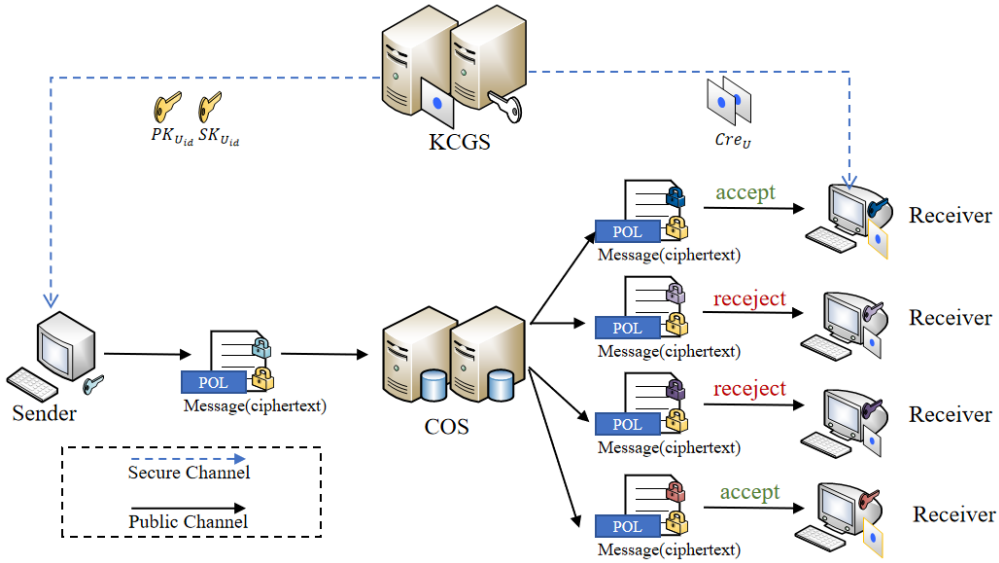


Figure 2. Framework of the PCE-EtoE scheme.

5.1. Overview

In this section, we provide a comprehensive review of the PCE-EtoE scheme. Table 2 presents the symbols used in the scheme and their meanings. As shown in Figure 3, our proposed scheme consists of six stages: Setup, Key and Credential Generation, key

agreement, Message Sending, Message Forwarding, and Message Receiving.The first two and Message Sending are randomized.

(1) **Setup:** Public Parameter Generation (Setup): The setup is aimed at initializing the system. Setup is a probabilistic polynomial time (PPT) algorithm where the security parameter l is input, and the public parameter $param$ is the output.

(2) **Key and Credential Generation:**
(a) KCGS Key Generation (KCGSKeyGen): KCGSKeyGen is aimed at constructing the KCGS's public and private keys. KCGSKeyGen is a PPT algorithm that outputs public key of KCGS PK_{KCGS} and private key of KCGS SK_{KCGS} after inputting public parameters.
(b) User Credential Generator (CreGen): CreGen is aimed at completing the construction of a user's credential. CreGen is a PPT algorithm that outputs user's credential Cre_U after inputting public parameters, user's attributes S, and public key of KCGS PK_{KCGS}.
(c) User Key Generation (UserKeyGen): UserKeyGen is aimed at constructing the user's public and private keys. UserKeyGen is a PPT algorithm that outputs public key of user $PK_{U_{id}}$ and private key of user $SK_{U_{id}}$ after inputting public parameters and public key of KCGS PK_{KCGS}.

(3) **Key Agreement:** Key agreement is aimed at completing the secure negotiation of a shared key between the sender and receiver. Input the user's public key for secondary encryption, PK_{KCGS}, and finally, output the shared key of the user and the COS (K_{S_iC} or K_{R_iC}).

(4) **Message Sending (MesSen):** Message Sending is aimed at completing the secure transmission of messages. Input access policy POL, group chat message M, public key of KCGS PK_{KCGS}, private key of user $SK_{U_{id}}$, shared key, and finally, output secondary ciphertexts ES_{se}.

(5) **Message Forwarding (MesForw):** Message Forwarding is aimed at accomplishing the secure forwarding of messages. Input secondary ciphertexts ES_{se}, the set of shared key $\{K_{R_iC}\}_{i<qc}$ (qc is the number of people in the group chat), and finally, output timestamp TS_m/TS and secondary forwarding ciphertext $\{ES_{RE}\}$.

(6) **Message Receiving (MesRec):** Message Receiving is aimed at completing the secure receiving of messages. Input the secondary forwarding ciphertext for the specified receiver ES_{re_i}, the receiver's certificate Cre_U, the public key of the sender $PK_{U_{id}}$, and output group chat message M or $\perp$.

The proposed PCE-EtoE is formally defined as

$$\prod_{PCE-EtoE} = \begin{bmatrix} \text{param} \leftarrow \text{Setup}(1^t) \\ (PK_{KCGS}, \text{SK}_{\text{KCGS}}) \leftarrow \text{KCGSKeyGen}(param) \\ Cre_U \leftarrow CreGen(param, S, PK_{KCGS}) \\ \left(SK_{U_{id}}, PK_{U_{id}}\right) \leftarrow \text{UserKeyGen}(param, PK_{KCGS}) \\ (0/1) \leftarrow \text{CreVer}(Cre_U) \\ K_{R_i/S_iC} \leftarrow \text{key Agreement}(y_{R_i/S_iC}, y_C) \\ ES_{se} \leftarrow \text{MesSen}(\text{POL}, M, PK_{KCGS}, PK_{U_{id}}, SK_{U_{id}}, K_{S_iC}) \\ (TS_m/TS, \{ES_{RE}\}) \leftarrow \text{MesForw}(ES_{se}, \{K_{R;C}\}_{i<q}) \\ (M/\perp) \leftarrow \text{MesRec}(ES_{re_i}, \text{Cre}_{\text{U}}, \text{PK}_{\text{CS}}) \end{bmatrix}$$

Table 2. Definitions of symbols used in the scheme.

Symbol	Description	Symbol	Description
PK_{KCGS}	Public key of KCGS	POL	Access policy
SK_{KCGS}	Private key of KCGS	$ES_{se}, ES_{re}, ES_{RE}$	Secondary ciphertext
S	User's attributes	Enc	AES encryption
Cre_U	User's credential	Dec	AES decryption
$PK_{U_{id}}$	Public key of user	σ	First-class ciphertext
$SK_{U_{id}}$	Private key of user	TS_{max}	Maximum time difference
$K_{S_i,C}$	The shared key of the $sender_i$ and the COS	TS_{now}	Current timestamp
$K_{R_i,C}$	The shared key of the $receiver_i$ and the COS	M	Group chat messages
TS_m	The timestamp of the text message	TS	Timestamp of the image message

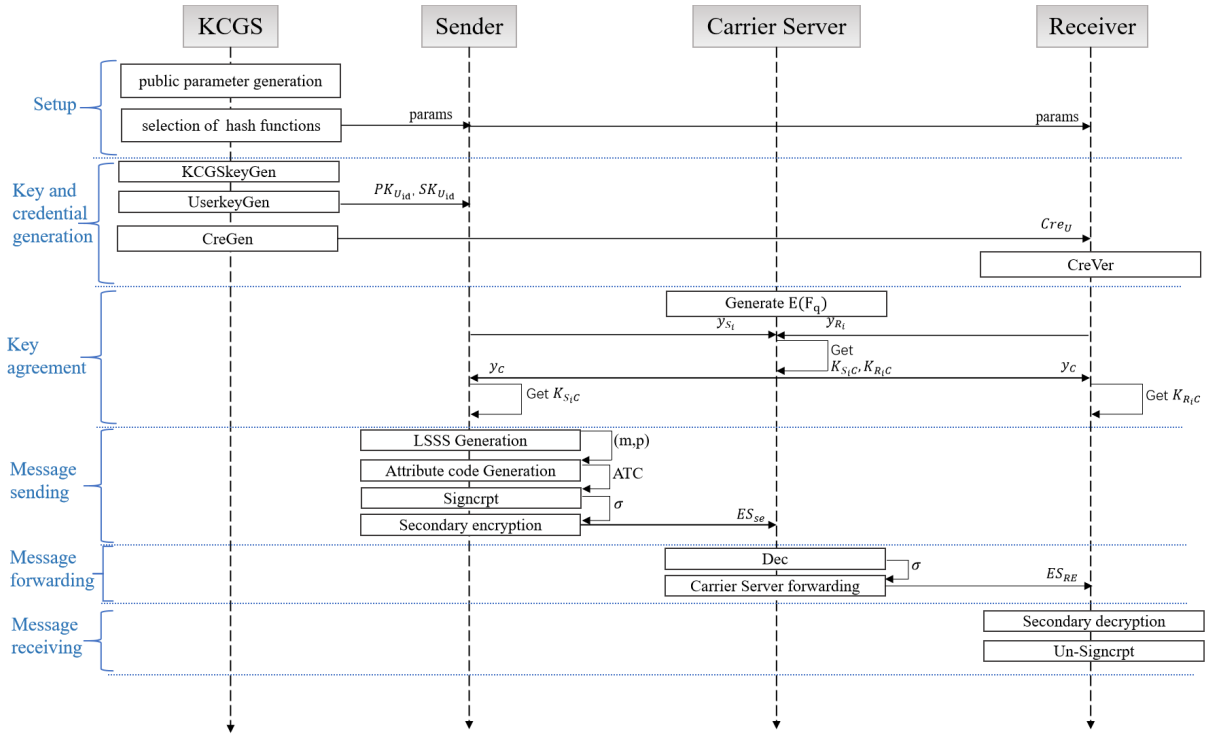


Figure 3. The process flow diagram of the PCE-EtoE system.

5.2. Setup

5.2.1. Public Parameter Generation

Input the security parameter l; KCGS randomly selects the prime $p = poly(1^l)$. Setup selects a random generator $g \in G_1$, then randomly selects $a \in Z_N$ and computes g^a. A bilinear mapping function ($\hat{e} : G_1 \times G_1 \rightarrow G_T$) is then created.

5.2.2. Selection of Hash Functions

Define four hash functions, the first is the attribute hash function $H_1 : S \rightarrow G_1$. The second is the collision-resistant hash function $H_2 : \{0,1\}^* \rightarrow Z_N$. The third is also the collision-resistant hash function $H_3 : m \rightarrow \{0,1\}^{ml}, ml = M.length$. The last is the file hash function: $H_4 : m \rightarrow G_1$.

The Setup algorithm outputs the system public parameters, denoted as $param = (l, g, \hat{e}, H_1, H_2, H_3, g_a, g_b)$.

The above procedure is shown in Algorithm 1.

Algorithm 1 Setup

Input: l
Output: $param$
1: select $p = poly(1^l), g \in G_1, a \in Z_N$
2: compute g^a
3: choose $\hat{e} : G_1 \times G_1 \to G_T$
4: choose hash functions $H_1 : S \to G_1, H_2 : \{0,1\}^* \to Z_N, H_3 : m \to \{0,1\}^{ml}, H_4 : m \to G_1$
5: **return** $param = (l, g, \hat{e}, H_1, H_2, H_3, g_a, g_b)$

5.3. Key and Credential Generation

5.3.1. KCGSKeyGen

Input the system parameter $param$, KCGS randomly selects $\alpha, \beta \in Z_n/0$, uses $SK_{KCGS} = (\alpha, \beta)$ as the private key of KCGS, computes $U = g^\alpha$ and $W = g^\beta$, and uses $PK_{KCGS} = (U, W)$ as the public key of KCGS. KCGSKeyGen outputs SK_{KCGS} and PK_{KCGS}.

5.3.2. CreGen

The set of attributes $S = \{S_1, S_2, \ldots, S_i\}$, the system parameter $param$, and the public key of KCGS PK_{KCGS} are input. KCGS randomly selects $t \in Z_N/0$ and computes $CK = g^\alpha g^{at}$, $CL = g^t$. KCGS computes $A_i = H_1(S_i), SK_i = A_i^t, S_i \in S$, where S_i denotes the i-th attribute in S. KCGS takes the $Cre_U = \{CK, CL, SK_i\}$ as a credential for the user. CreGen outputs Cre_U.

5.3.3. CreVer

The users send their identity ID to KCGS. KCGS matches the corresponding attribute set according to the identity ID and runs CreGen to obtain Cre_U. The user checks the validity of $\{SK_i\}$, CK, and CL in Cre_U as follows:

$$\hat{e}(SK_i, g) = \hat{e}(A_i, CL) \tag{1}$$

$$\hat{e}(CK, g) = \hat{e}(U, g)\hat{e}(g^a, CL) \tag{2}$$

If (1) and (2) hold, then the validity of Cre_U holds.

5.3.4. UserKeyGen

Input the system parameter $param$ and PK_{KCGS}; KCGS then randomly selects $\mu, \gamma \in Z_n/0$, uses $SK_{U_{id}} = (\mu, \gamma)$ as the private key of the user, computes $Y_1 = g^{\mu\gamma}$ and $Y_2 = W^{\mu\gamma}$, and uses $PK_{U_{id}} = (Y_1, Y_2)$ as the public key of the user. UserKeyGen outputs $SK_{U_{id}}$ and $PK_{U_{id}}$.

The above procedure is shown in Algorithm 2.

Algorithm 2 Key and Credential Generation

Input: $param$,S
Output: SK_{KCGS}, PK_{KCGS}, $Cre_U/\perp$, $SK_{U_{id}}$, $PK_{U_{id}}$
KCGS select $\alpha, \beta \in Z_n/0$
Let $SK_{KCGS} = (\alpha, \beta)$
Compute $U = g^{\alpha}$, $W = g^{\beta}$
Let $PK_{KCGS} = (U, W)$ // KCGSKeyGen's procedure
KCGS select $t \in Z_N/0$
Compute $CK = g^{\alpha} g^{at}$, $CL = g^{t}$
for i $\in S$ **do**
    $A_i = H_1(S_i), SK_i = A_i^t$
end for
Let $Cre_U = \{CK, CL, \{SK_i\}\}$ // CreGen's procedure
KCGS select $\mu, \gamma \in Z_n/0$
Let $SK_{U_{id}} = (\mu, \gamma)$
Compute $Y_1 = g^{\mu\gamma}$, $Y_2 = W^{\mu\gamma}$
Let $PK_{U_{id}} = (Y_1, Y_2)$ // UserKeyGen's procedure
if $\hat{e}(SK_i, g) = \hat{e}(A_i, CL)$&& $\hat{e}(CK, g) = \hat{e}(U, g)\hat{e}(g^a, CL)$ **then**
    return SK_{KCGS}, PK_{KCGS}, Cre_U/,$SK_{U_{id}}$, $PK_{U_{id}}$
else
    return SK_{KCGS}, PK_{KCGS}, $\perp$,$SK_{U_{id}}$, $PK_{U_{id}}$
end if// CreVer's procedure

5.4. Key Agreement

The key agreement in our scheme involves interaction between the user and server ends of the Communication Operator Server (COS). The agreement process is established using the Elliptic Curve Diffie–Hellman (ECDH) algorithm. Given the parameters $z, n \in F_q$, where $4z^3 + 27 \neq 0$, the group $E(F_q)$ is defined as follows:

$$E(F_q) = \{Q = (x, y) | y^2 = x^3 + zx + nmodq, x, y \in F_q\} \bigcup \{\infty\}$$

where ∞ is the point at infinity.

The framework for key agreement in group chat communication is shown in Figure 4. The process of key agreement between group chat user $Sender_a$ and the Communication Operator Server (COS) is detailed below:

(1) $Sender_a$ computes the public key $y_{S_a} = g^{x_{S_a}} modq$ according to the published elliptic curve $E(F_q)$ and sends y_{S_a} to the Communication Operator Server (COS).
(2) After receiving y_{S_a}, the COS computes $y_C = g^{x_C} modq$ and $K_{S_aC} = y_{S_a}^{x_C} modq$, and then sends y_C to $Sender_a$. Here, K_{S_aC} is the shared key between $Sender_a$ and COS.
(3) After $Sender_a$ obtains y_C, it computes the shared key $K_{S_aC} = y_C^{x_{S_a}} modq$.
The shared key obtained after key agreement is used as the second-level encryption key in secure group chat communication. After establishing the shared keys between the COS and all users in the group chat, all the shared keys and their corresponding user identities are stored in the shared key table.

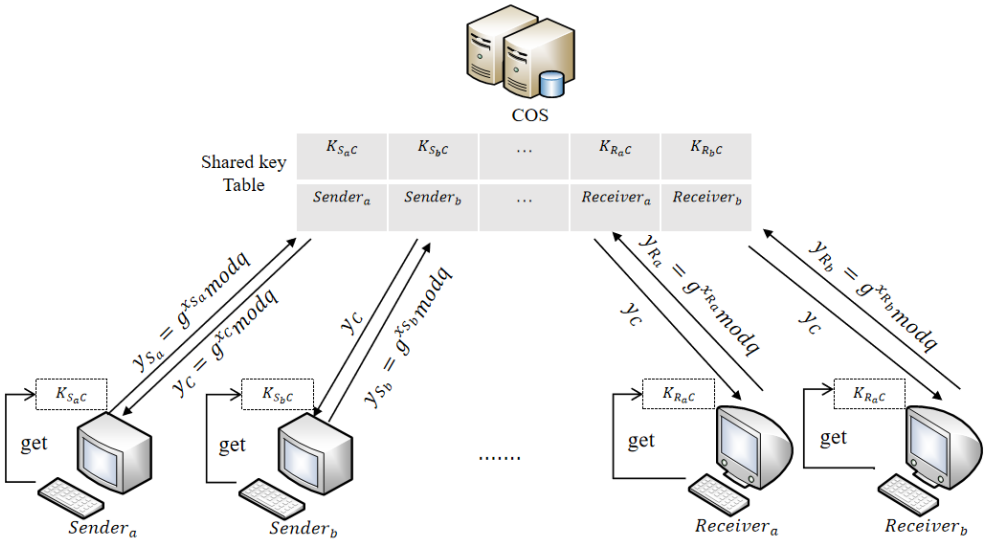


Figure 4. Framework of key agreement.

5.5. Message Sending (MesSen)

The sender sets the access policy *POL* to control the flow of messages, e.g.:

$(Sales\_department$ **and** $First - level\_manager)$ **or** $(Purchasing\_department$ **and** $Ordinary\_staff)$.

5.5.1. LSSS Generation

The Boolean access policy *POL* is transformed into a linear secret sharing scheme (*LSSS*) shared matrix [28], and matrix M serves as the shared generator matrix of *POL* with n rows and l columns. The function p maps each row of matrix M onto the attribute names in the *POL*. Finally, (M,p) is used to represent the access policy.

5.5.2. Attribute Code Generation

The sender randomly selects a column vector $v = (s, y_1, \ldots, y_l)$, where $s \in Z_p, y_1, \ldots, y_l \in Z_p$. s is the secret value to be shared, and $y_1, \ldots, y_l$ are randomly selected group elements in the integer group. The sender computes $C_s = g^s$ and $\lambda_i = v \times M_i$, where M_i represents the i-th row in matrix M, and λ_i represents the partial secret value corresponding to the i-th attribute value in the access policy. Mark P_i as the attribute value in *POL*, for example, P_1='A department'. $\{P_i\}$ is the set of attributes in *POL*. Generate Attribute Code (*ATC*) based on P_i and compute $B_i = H_1(P_i)$ for each P_i. Randomly select $d_1, d_2, \ldots, d_i \in Z_p$ and then count $AC_i = g^{a\lambda_i} B_i^{d_i}$ and $ST_i = g_i^d$ to obtain $ATC = \{\{AC_i\}, \{ST_i\}, C_s\}$; store *ATC* as a special label in the user's device to avoid repetitive computation.

5.5.3. Signcryption

Upon inputting message M, its type is determined. If the message is an image file (e.g., jpg, png), the image data are converted into a binary file M_{bin} and given the label T. Randomly select $\bar{r}, j \in Z_p$, compute partial signatures $\sigma_1 = g^{\bar{r}}$ and $\sigma_2 = Y_1^{\bar{r}}$, and then calculate

$$\triangle = \sigma_1 || \sigma_2 || PK_{KCGS} || PK_{U_{id}} || j \tag{3}$$

$$w = \hat{e}(C_s, U) \tag{4}$$

$$\xi = j \bigoplus H_2(w || H_2(\triangle)) \tag{5}$$

If the message type is text type, it will use (3), (4), and (5) to compute

$$CT_M = M \bigoplus H_3(w||H_2(\triangle)) \tag{6}$$

$$\theta = \triangle||\xi||\{AC_i\}||\{ST_i\}||M \tag{7}$$

$$\sigma_3 = H_4(\theta)^{\mu\gamma} \tag{8}$$

Then, we derive σ={CT_M,σ_1,σ_2,σ_3,$H_2(\triangle)$,ATC,TS_m,$PK_{U_{id}}$} from (6), (7), and (8) ($PK_{U_{id}}$ is the public key of the sender).

If the message type is a picture type, it will use (3), (4), and (5) to compute

$$CT_{M_{bin}} = M_{bin} \bigoplus H_3(w||H_2(\triangle)) \tag{9}$$

$$\theta = \triangle||\xi||\{AC_i\}||\{ST_i\}||M \tag{10}$$

$$\sigma_3 = H_4(\theta)^{\mu\gamma} \tag{11}$$

Then, we derive σ={$CT_{M_{bin}}$,σ_1,σ_2,σ_3,$H_2(\triangle)$,ATC,TS_m,$PK_{U_{id}}$,T} from (9), (10), and (11).

5.5.4. Secondary Encryption

To ensure that the COS can verify the origin of the message, σ is encrypted using the shared key K_{S_iC} of the sender and the COS, denoted as follows:

$$ES_{se} = Enc(\sigma, K_{S_iC}) \tag{12}$$

Then, the secondary ciphertext ES_{se} is sent to the COS. The above procedure is shown in Algorithm 3.

Algorithm 3 Message Sending

Input: POL, M, PK_{KCGS}, $PK_{U_{id}}$, $SK_{U_{id}}$, K_{S_iC}
Output: ES_{se}

LSSS(POL)->(M,p) // LSSS Generation's procedure
Select $s \in Z_p, y_1, \ldots, y_l \in Z_p$,Let $v = (s, y_1, \ldots, y_l)$, Compute $C_s = g^s$
for i $\in [0, M.RowNumber]$ **do**
  $\lambda_i = v \times M_i$
end for
Select $d_1, d_2, \ldots, d_i \in Z_p$
for i $\in [0, P.Number]$ **do**
  Compute $B_i = H_1(P_i)$, $AC_i = g^{a\lambda_i} B_i^{d_i}$, $ST_i = g_i^d$
end for
Let $ATC = \{\{AC_i\}, \{ST_i\}, C_s\}$ // Attribute Code Generation's procedure
Select $\overline{r}, j \in Z_p$
Compute $\sigma_1 = g^{\overline{r}}$, $\sigma_2 = Y_1^{\overline{r}}$, $\triangle = \sigma_1||\sigma_2||PK_{KCGS}||PK_{U_{id}}||j$, $w = \hat{e}(C_s, U)$
Compute $\xi = j \bigoplus H_2(w||H_2(\triangle))$
Compute $CT_M/CT_{M_{bin}} = M/M_{bin} \bigoplus H_3(w||H_2(\triangle))$, $\theta = \triangle||\xi||\{AC_i\}||\{ST_i\}||M$, $\sigma_3 = H_4(\theta)^{\mu\gamma}$ // Signcrpyt 's procedure
Compute ES_{se}=Enc($CT_{M_{bin}}$,σ_1,σ_2,σ_3,$H_2(\triangle)$,ATC,TS_m,$PK_{U_{id}}$,T,K_{S_iC})
return ES_{se}

5.6. Message Forwarding (MesForw)

The COS first checks whether $|TS - TS_{now}| < TS_{max}$. If false, a packet loss delay message is displayed to the user and prompts information resend. If true, proceed with the following steps.

After receiving ES_{se}, the COS decrypts it as follows:

$$\sigma = Dec(ES_{se}, K_{S_iC}) \tag{13}$$

Then, the COS executes Algorithm 4 using the set of shared keys of the receiver and the COS $\{K_{R_iC}\}_{i<qc}$ as input.

Algorithm 4 COS Forwarding

Input: $\{K_{R_iC}\}$,ID_{U_i}
Output: TS_m,ES_{RE}
1: **for** i $\in$ [1, qc] **do**
2: $ES_{re_i} = Enc(\sigma, K_{R_iC})$
3: $ES_{RE}.append(ES_{re_i}, ID_{U_i})$
4: **end for**
5: **return** ES_{RE},TS_m

Then, the COS retrieves the ES_{RE} and forwards the contained ES_{re_i} to the user, who is identified ID_{U_i}.

5.7. Message Receiving (MesRece)

5.7.1. Secondary Decryption

The receiver verifies if $|TS_m - TS_{now} < TS_{max}|$ is true; if not, they resend the receive request to the COS. If it is true, the user performs the following actions.

After receiving ES_{se}, the receiver in the group chat uses the shared key K_{R_iC} between the receiver and the COS to decrypt it as follows:

$$\sigma = Dec(ES_{re_i}, K_{R_iC}) \tag{14}$$

After successful decryption, the preliminary source of the message is confirmed to be the COS.

5.7.2. Un-Signcrypt

The receiver inputs Cre_U and computes

$$\hat{e}(\sigma_1, g) = \hat{e}(\sigma_2, Y_1) \tag{15}$$

which may hold or not. If (15) is not true, the receiver rejects the message. Otherwise, proceed with the following calculation. There exists constants $\{\omega_i \in Z_N\}_{i\in I}$ satisfying $\Sigma_{i\in I}\omega_i\lambda_i = s$ in time polynomial in the size of the share-generating matrix M. Therefore, if $\{\lambda_i\}$ are valid shares of any secret (s) according to (M , p), then calculate

$$\varphi = \frac{\hat{e}(C_s, CK)\Pi_{i\in I}\hat{e}(SK_i, ST_i)^{\omega_i}}{\Pi_{i\in I}\hat{e}(CL, AC_i)^{\omega_i}} \tag{16}$$

$$\hat{j} = \xi \bigoplus H_2(w||H_2(\triangle)) \tag{17}$$

$$M = CT_M \bigoplus H_3(w||H_2(\triangle)) \tag{18}$$

$$\hat{\theta} = \sigma_1||\sigma_2||PK_{U_{id}}||\hat{j}||\xi||\{AC_i\}||M||\{ST_i\} \tag{19}$$

Finally, it uses (16), (17) , (18), and (19) to verify whether $\hat{e}(H_4(\hat{\theta}), Y_1) = \hat{e}(\sigma_3, g)$ is true. If it is true, the ciphertext σ is valid, and the message M is received; otherwise, the receiver rejects the message and returns the symbol $\perp$.

The above procedure is shown in Algorithm 5.

Algorithm 5 Message Receiving

Input: ES_{re_i},$\mathrm{Cre_U}$, $\mathrm{PK_{CS}}$
Output: $M/\perp$
if $|TS_{m-TS_{now}} < TS_{max}|$ **then**
  Compute $\sigma = \mathrm{Dec}(ES_{re_i}, K_{R_iC})$ // Secondary decryption's procedure
  if $\hat{e}(\sigma_1, g) = \hat{e}(\sigma_2, Y_1)$ **then**
  Compute $\varphi = \frac{\hat{e}(C_s,CK)\Pi_{i\in I}\hat{e}(SK_i,ST_i)^{\omega_i}}{\Pi_{i\in I}\hat{e}(CL,AC_i)^{\omega_i}}$
  Compute $\hat{j} = \xi \bigoplus H_2(w||H_2(\triangle))$
  Compute $M = CT_M \bigoplus H_3(w||H_2(\triangle))$
  Compute $\hat{\theta} = \sigma_1||\sigma_2||PK_{U_{id}}||\hat{j}||\xi||\{AC_i\}||M||\{ST_i\}$ // Un-signcrypt's procedure
  return M
else
  return $\perp$
end if

6. Security and Functional Analysis

6.1. Correctness

In this section, we analyze the correctness of the proposed scheme. We need to check that the user can use $\Sigma_{i\in I}\omega_i\lambda_i$ of the LSSS matrix to recover s, provided that the user's credentials meet the requirements of the access policy. The detailed derivation process is as follows:

$$\begin{aligned}
\varphi &= \frac{\hat{e}(C_s, CK)\Pi_{i\in I}\hat{e}(SK_i, ST_i)^{\omega_i}}{\Pi_{i\in I}\hat{e}(CL, AC_i)^{\omega_i}} \\
&= \frac{\hat{e}(g^s, g^{\alpha}g^{at})\Pi_{i\in I}\hat{e}(A_i^t, g^{d_i})^{\omega_i}}{\Pi_{i\in I}\hat{e}(g^t, g^{a\lambda_i}B_i^{d_i})^{\omega_i}} \\
&= \frac{\hat{e}(g^s, g^{\alpha}g^{at})\Pi_{i\in I}\hat{e}(A_i^t, g^{d_i})^{\omega_i}}{\Pi_{i\in I}\hat{e}(g^t, g^{a\lambda_i})^{\omega_i}\hat{e}(g^t, B_i^{d_i})^{\omega_i}} \\
&\overset{\text{if } B_i = A_i}{=} \frac{\hat{e}(g^s, g^{\alpha}g^{at})}{\Pi_{i\in I}\hat{e}(g^t, g^{a\lambda_i})^{\omega_i}} \\
&= \frac{\hat{e}(g^s, g^{\alpha}t)\hat{e}(g,g)^{\alpha s}}{\hat{e}(g^t, g^a)^s} \\
&= \hat{e}(g,g)^{\alpha s}
\end{aligned}$$

6.2. Confidentiality

Theorem 1. *If the assumptions of the DBDH hard problem hold, there is no attacker who can break the PCE-EtoE scheme with a non-negligible advantage in polynomial time.*

Proof of Theorem 1. The challenger $\mathcal{B}$ chooses four random numbers $\hat{a}, \hat{b}, \hat{c}, \hat{\theta} \in Z_N$, and then chooses the random number $\delta \in \{0,1\}$. If $\delta = 0$, the challenger $\mathcal{B}$ sets $Z = \hat{e}(\hat{g}, \hat{g})^{\hat{a}\hat{b}\hat{c}}$. If $\delta = 1$, then let $Z = \hat{e}(\hat{g}, \hat{g})^{\hat{\theta}}$. Finally, the challenger $\mathcal{B}$ sends the tuple $(\hat{g}^{\hat{a}}, \hat{g}^{\hat{b}}, \hat{g}^{\hat{c}}, Z)$ to simulator $\mathcal{C}$, and then simulator $\mathcal{C}$ interacts with attacker $\mathcal{A}$ instead of challenger $\mathcal{B}$. The detailed steps are as follows:

(1) Initialization: Simulator $\mathcal{C}$ first sets $g = \hat{g}$, randomly selects $x_1 \in Z_N$, sets $U = \hat{g}^{\hat{a}}$ and $W = \hat{g}^{\hat{b}+x_1}$ in PK_{KCGS}, and sets $\hat{h} = \hat{a}\hat{b}$ and $Y_1 = \hat{g}^{\hat{h}}$ and $Y_2 = \hat{g}^{\hat{h}+x_1}$ in $PK_{U_{id}}$. Finally, the two public keys and system parameters are provided to attacker $\mathcal{A}$, and the list H is initialized.

(2) Phase 1: Attacker $\mathcal{A}$ can query simulator $\mathcal{C}$ for the private key, and the simulator queries the list $H1$, returns the result if it exists, selects the random number F if it does not exist, and updates the list $H1$. An attacker can submit a user set S to simulator $\mathcal{C}$ to

query for any credential that is not used to challenge the ciphertext of un-signcryption. For the credential query submitted by $\mathcal{A}$, the simulator selects $t \in [1, n-1]$. For each attribute $i \in S$, the simulator queries the list $H2$ if it contains the attribute; if it does not, it randomly selects $k_i, q \in [1, n-1]$ and adds the element $(i, k_i, \hat{g}^{\hat{b}k_i})$ to the list and updates it. Choose random numbers $x_2, x_3, x_4 \in Z_N$; let $CL = \hat{g}^{\frac{\hat{a}th+x_4}{q}}$, $CK = \hat{g}^{\hat{a}t\frac{x_3}{x_2}+\hat{c}t\frac{x_3}{x_2}+\hat{b}\frac{x_4}{x_2}}$; randomly choose $c_1 \in Z_N$; let $c_2 = c - c_1$; and calculate $SK_i = \hat{g}^{\frac{-c_2\hat{a}\hat{b}th}{k_i}}$. Finally, simulator $\mathcal{C}$ sends the un-signcryption's credential to $\mathcal{A}$.

(3) Challenge: Attacker $\mathcal{A}$ submits two randomly chosen messages m_0 and m_1 of the same length to simulator $\mathcal{C}$, where m_0 and m_1 have the same POL. The simulator $\mathcal{C}$ first randomly selects $\delta \in \{0, 1\}$, and then sets $w = \frac{\hat{e}(\hat{g}^{\hat{a}}, g^{c_1})^{tx_3}\hat{e}(\hat{g}^{c_1}, \hat{g}^{\hat{c}})^{tx_3}}{Z^{th}}$, randomly selects a column vector $v = (s, y_1, \ldots, y_l)$, where $s \in Z_N, y_1, \ldots, y_l \in Z_N$. Calculate $\lambda_i = v \times M_i$, where $\Sigma_{i \in I}\omega_i\lambda_i = s$ exists. Compute $AC_i = g^{\frac{\lambda_i c_1 \hat{b}k_i}{s}}, ST_i = g^{\frac{k_i}{\omega_i}}$. Let $C_s = \hat{g}^{x_2 c_1}$ and then perform signcryption, choose $\hat{r}, \hat{j} \in Z_N$ at random, compute $\sigma_1 = \hat{g}^{\hat{r}}, \sigma_2 = Y_1^{\hat{r}}$, and then compute as follows:

$$\begin{aligned}
\triangle &= \sigma_1 || \sigma_2 || PK_{KCGS} || PK_{U_{id}} || \hat{j} \\
w &= \hat{e}(C_s, U) \\
\xi &= \hat{j} \bigoplus H_2(w || H_2(\triangle)) \\
CT_M &= m_\delta \bigoplus H_3(w || H_2(\triangle)) \\
\theta &= \triangle || \xi || \{AC_i\} || \{ST_i\} || m_\delta \\
\sigma_3 &= H_4(\theta)^{\hat{h}}
\end{aligned}$$

Finally, the simulator sends the ciphertext $\sigma = \{CT_M, \sigma_1, \sigma_2, \sigma_3, H_2(\triangle), \{AC_i\}, \{ST_i\}, TS\}$ to attacker $\mathcal{A}$.

(4) Phase 2: Similar to phase 1, attacker $\mathcal{A}$ can submit a user attribute set S to the simulator to query any credential not used for the un-signcryption challenge ciphertext.

(5) Guess: Attacker $\mathcal{A}$ outputs a guess $\hat{\sigma}$ of σ. If $\hat{\sigma} = \sigma$, and then simulator $\mathcal{C}$ outputs 0, indicating that the guess is $Z = \hat{e}(\hat{g}, \hat{g})^{\hat{a}\hat{b}\hat{c}}$. Otherwise, simulator $\mathcal{C}$ outputs 1, indicating that the guess is $Z = \hat{e}(\hat{g}, \hat{g})^{\hat{\theta}}$. If $Z = \hat{e}(\hat{g}, \hat{g})^{\hat{a}\hat{b}\hat{c}}$, then input σ and Cre_U, and compute

$$\begin{aligned}
\varphi_1 &= \frac{\Pi_{i \in I}\hat{e}(SK_i, ST_i)^{\omega_i}}{\Pi_{i \in I}\hat{e}(CL, AC_i)^{\omega_i}} \\
&= \frac{\hat{e}(\hat{g}, \hat{g})^{-c_2 t\hat{a}\hat{h}\hat{b}}}{\hat{e}(\hat{g}, \hat{g})^{c_1\hat{b}x_4 + c_1\hat{a}t\hat{h}\hat{b}}} \\
&= \hat{e}(\hat{g}, \hat{g})^{-(c_1\hat{b}x_4 + c_1\hat{a}t\hat{h}\hat{b})}
\end{aligned}$$

and because

$$\varphi_2 = \hat{e}(C_s, CK) = \hat{e}(\hat{g}, \hat{g})^{(\hat{a}+\hat{c})tx_3 + c_1\hat{b}x_4}$$

we can obtain the following:

$$\varphi = \varphi_1 \times \varphi_2 = \frac{\hat{e}(\hat{g}, \hat{g})^{(\hat{a}+\hat{c})tx_3}}{\hat{e}(\hat{g}, \hat{g})^{\hat{a}\hat{b}\hat{c}t\hat{h}}} = \frac{\hat{e}(\hat{g}^{\hat{a}}, g^{c_1})^{tx_3}\hat{e}(\hat{g}^{c_1}, \hat{g}^{c})^{tx_3}}{Z^{t\hat{h}}}.$$

The above demonstrates the validity of the signcryption ciphertext σ. Since attacker $\mathcal{A}$'s advantage is denoted as ε, the probability that attacker $\mathcal{A}$ can correctly guess σ in this case is

$$Pr[\mathcal{C}(\hat{g}^{\hat{a}}, \hat{g}^{\hat{b}}, \hat{g}^{\hat{c}}, Z = \hat{e}(\hat{g}, \hat{g})^{\hat{a}\hat{b}\hat{c}}) = 0] = \frac{1}{2} + \varepsilon$$

If $Z = \hat{e}(\hat{g}, \hat{g})^{\hat{\theta}}$, since $\hat{\theta}$ is randomly chosen in Z_N, the probability that attacker $\mathcal{A}$ guessed correctly in this case is

$$Pr[\mathcal{C}(\hat{g}^{\hat{a}}, \hat{g}^{\hat{b}}, \hat{g}^{\hat{c}}, Z = \hat{e}(\hat{g}, \hat{g})^{\hat{\theta}}) = 0] = \frac{1}{2}$$

In summary, the advantages of Simulator $\mathcal{C}$ are as follows:

$$\frac{1}{2}(Pr[\mathcal{C}(\hat{g}^{\hat{a}}, \hat{g}^{\hat{b}}, \hat{g}^{\hat{c}}, Z = \hat{e}(\hat{g}, \hat{g})^{\hat{\theta}}) = 0] + Pr[\mathcal{C}(\hat{g}^{\hat{a}}, \hat{g}^{\hat{b}}, \hat{g}^{\hat{c}}, Z = \hat{e}(\hat{g}, \hat{g})^{\hat{a}\hat{b}\hat{c}}) = 0]) - \frac{1}{2} = \frac{\varepsilon}{2}$$

The above can be proven as follows: under the DBDH assumption, the proposed scheme satisfies IND-CPA security.

□

6.3. Resist Communication Operator Theft

Theorem 2. *If the COS cannot obtain the message of plaintext in the group chat communication, it is claimed that PCE-EtoE has the capability to resist communication operator theft.*

Proof of Theorem 2. Sender A in the group chat first signcrypts the message and encrypts it with AES to obtain the ciphertext ES_{se}. Then, A sends ES_{se} to the COS, which decrypts ES_{se} with K_{S_AC} to obtain σ. According to Theorem 1, the COS cannot construct a valid credential Cre_U to un-signcrypt σ; thus, the plaintext M of the message cannot be obtained. Therefore, it can be demonstrated that PCE-EtoE has the capability to resist communication operator theft. □

6.4. Mitigate Message Overload

Theorem 3. *If the receiver in a group chat can only receive messages related to itself, PCE-EtoE is said to have the capability to mitigate message overload.*

Proof of Theorem 3. Suppose sender A in a group chat formulates $POL = (S_A$ **and** $(S_B$ **or** $S_C))$, and the attribute stored in KCGS for receiver B is (S_A, S_D). A performs the steps of sending the message, the COS performs the steps of forwarding the message, and B receives the ciphertext ES_{re_B}. The receiver first performs AES decryption with the shared key to obtain the ciphertext σ and uses the credentials Cre_U generated by (S_A, S_D) to un-signcrypt σ, but B cannot calculate the correct $\hat{e}(g, g)^{\alpha s}$ (i.e., it cannot correctly un-signcrypt). The receiver then rejects the message, avoiding the reception of irrelevant messages. This demonstrates that PCE-EtoE has the capability to mitigate message overload. □

7. Performance Evaluation

7.1. Theoretical Performance

We evaluate the performance of the PCE-EtoE scheme in comparison to various existing schemes. The symbols used in this section are defined in Table 3.

Table 3. Definitions of symbols used in performance evaluation.

Symbols	Description
E, E_T	Exponentiation operations in groups G_T/G
l_s, l_e	Numbers of signature/encryption attributes involved
P	Bilinear mapping operation

In attribute-based cryptography, bilinear mapping operations and exponentiation are the most computationally intensive. The cost of other operations, such as hashing, XOR, and constant operations, is negligible. Table 4 compares the computational cost of signcryption and un-signcryption operations in the proposed PCE-EtoE scheme to various

existing schemes, and Table 5 compares the computational cost of credential generation and verification in the proposed PCE-EtoE scheme to existing schemes.

Table 4. Comparison of theoretical computation cost for signcryption and un-signcryption.

Scheme	User Signcryption	User Un-Signcryption
[24]	$(2l_s + 2l_e + 3)E + 2E_T + P$	$(l_s \log l_s + l_e \log l_e + 1)E_T + (2l_s + 2l_e + 1)P$
[19]	$(2l_s + 2l_e + 1)E$	$(l_s + 2)E + l_sP$
[20]	$(2l_s + 9)E + E_T$	$(2l_e + 2)E + 6P$
[21]	$(2l_s + 7)E + E_T$	$2l_eE + E_T + 8P$
PCEA [13]	$(10 + l_s)E + l_sP$	$(12 + 2l_s)P + E_T$
[22]	$(l_s + 5)E + E_T$	$(l_s + 2)E + (l_s + 3)P + E_T$
[23]	$(2l_s + 10)E + E_T$	$(2l_e + 2)E + 6P$
PCE-EtoE	$3E + P$	$(4 + 2l_s)P$

Table 5. Comparison of theoretical computation cost for CreGen and CreVer.

Scheme	User Signcryption	User Un-Signcryption
PCEA [13]	$4l_sE$	$(1 + 4l_s)P$
PCE-EtoE	$(2 + l_s)E$	$(3 + 2l_s)P$

It can be inferred from Table 4 that in schemes [19–24], when performing the signcryption operation, the computation required is linear with the number of attributes, while in our proposed scheme, the computation requires only a constant $3E + P$, which does not increase with the number of attributes. Although the un-signcryption operation of PCE-EtoE does not achieve a constant computational cost, it is the lowest among the schemes, being slightly lower than that of [22,23] and significantly lower than that of other schemes. Since only PCEA uses certificates for un-signcryption, the computational costs of credential generation and verification are compared solely with the PCEA scheme in Table 5. It can be observed from Table 5 that the computational cost of certificate generation in PCE-EtoE is lower than that of PCEA by $(3l_s - 2)E$, and the computational cost of certificate verification in PCE-EtoE is also lower than that of PCEA by $(3l_s - 2)E$.

7.2. Actual Performance

As shown in Figure 5, we used a variety of terminal devices in terms of hardware to ensure the wide applicability and performance of the scheme. Specifically, we used the following equipment:

Dell Computer: Equipped with a 12th-generation Intel i7-1700 processor, with a main frequency of 2.10 GHz and 16 GB of memory.

Raspberry Pi 4 Model B: Powered by a 1.5 GHz quad-core Cortex-A72 CPU with 4 GB of RAM.

By testing on these two significantly different devices, we were able to fully evaluate the performance of the solution in different hardware environments, thus ensuring its reliability and stability in diverse application scenarios. In terms of software, the programming language used was Java 1.8, the IDE environment was IDEA 2020, and the cryptography open-source libraries used were JPBC 2.0.0, Crypto, and Security. The test curves used were Type A and secp256r1.

We performed signcryption and un-signcryption tests on text messages of different sizes using DELL computers and Raspberry Pi devices.

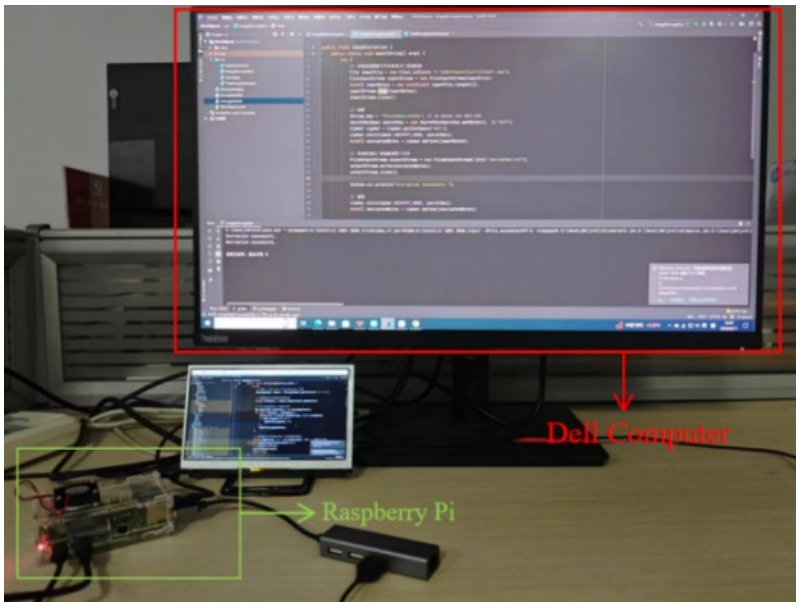
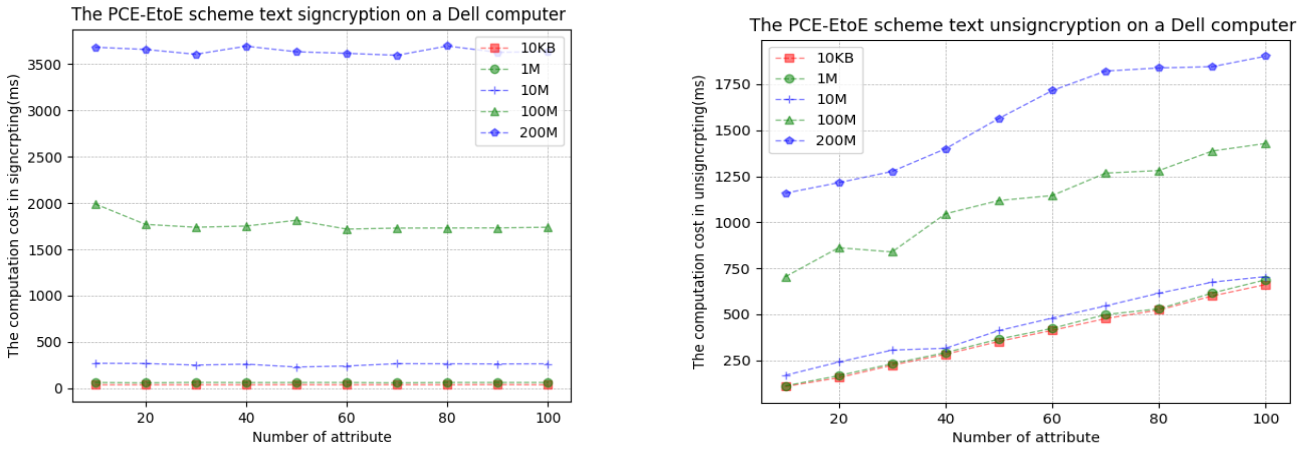


Figure 5. Experimental environment.

For the DELL computers, we chose 10 KB, 1 MB, 10 MB, 100 MB, and 200 MB text messages for our experiments. Figure 6a shows the signcryption computation time for different file sizes on DELL computers. The computation time was relatively low for smaller files (10 KB and 1 MB) and significantly increased for larger files (100 MB and 200 MB). However, as the number of attributes in the access policy increased, the computation time for the signcryption operation remained essentially unchanged. When the text message size was 200 MB, the signcryption time remained approximately 3.5 s regardless of the number of attributes. Figure 6b shows the un-signcryption time for different text message sizes on DELL computers. The un-signcryption time for 100 MB and 200 MB messages was much higher than for 10 KB and 1 MB files. When the text message size was 200 MB and the number of attributes in the access policy was 100, the decryption time was about 1.8 s.

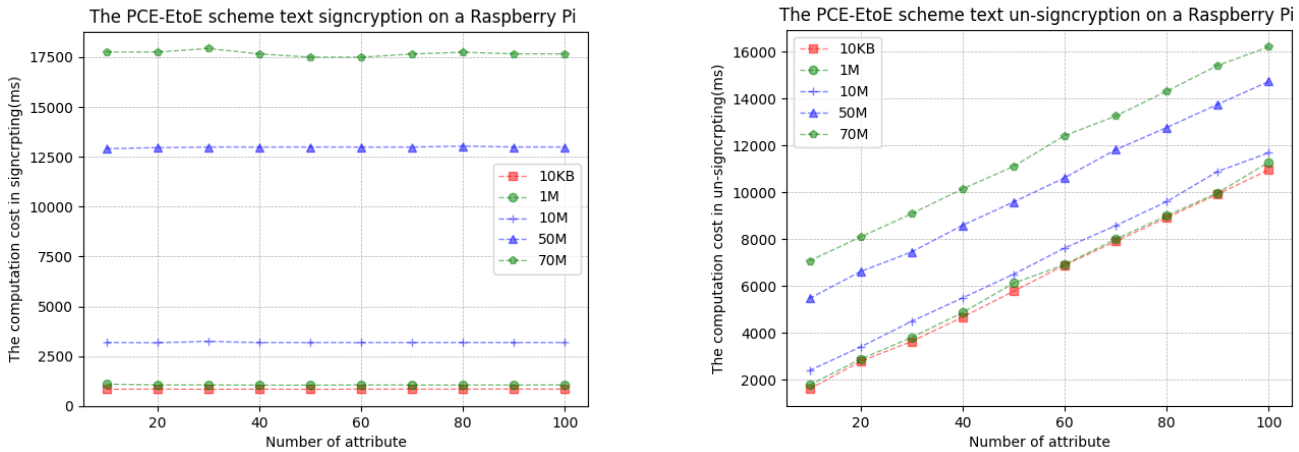


(**a**) (**b**)

Figure 6. PCE-EtoE's un-signcryption and signcryption of text-type messages in a Dell computer. (**a**) The PCE-EtoE scheme text sigcryption-dell; (**b**) The PCE-EtoE scheme text un-sigcryption-dell.

On the Raspberry Pi device, due to limited computing resources, we selected 10 KB, 1 MB, 50 MB, and 70 MB text messages for experiments. Figure 7a shows the signcryption computation times for different text message sizes on the Raspberry Pi. The computation times were relatively low for 10KB and 1MB messages but significantly increased for 50 MB

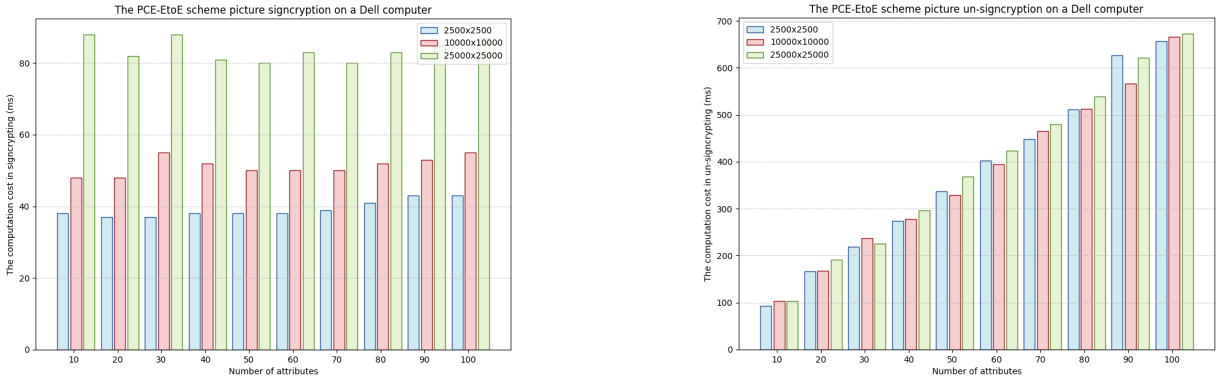
and 70 MB messages. However, as the number of attributes in the access policy increased, the computation time for the signcryption operation remained essentially unchanged. When the size of a text message was 70 MB, the signcryption time was approximately 17.5 s regardless of the number of attributes. Figure 7b shows the un-signcryption time for different text message sizes on a Raspberry Pi. The time for the un-signcryption of 50 MB and 70 MB messages was much higher than for 10 KB and 1 MB files. When the text message size was 70 MB and the number of attributes in the access policy was 100, the decryption time was about 16 s.The computing resources of the Raspberry Pi are insufficient for handling large files. Message size significantly affects the computation time of signcryption and un-signcryption, particularly as file size increases. The tests indicate that increasing the number of attributes in the access policy has a minimal effect on the signcryption and un-signcryption times, particularly for large files. This phenomenon may be attributed to the fact that the impact of increasing the number of attributes on computation time is overshadowed by the overall data processing time.



(**a**) (**b**)

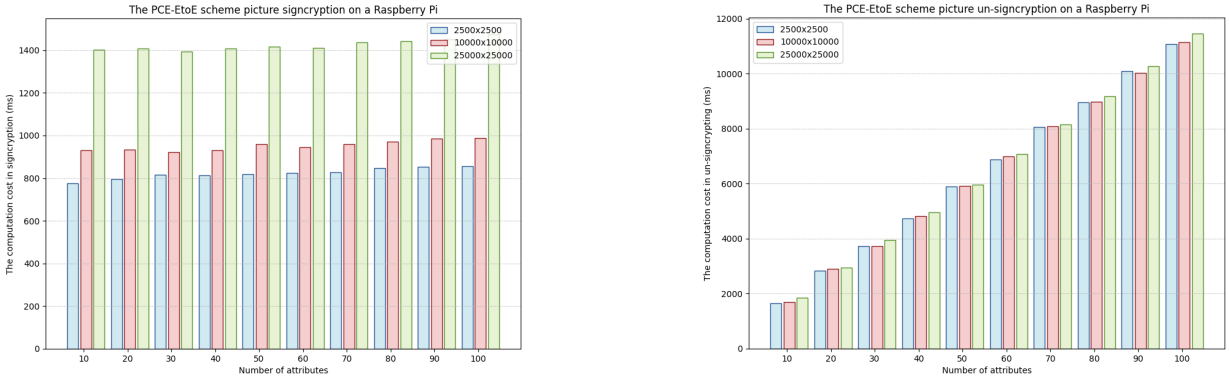
Figure 7. PCE-EtoE's un-signcryption and signcryption of text-type messages in Raspberry Pi. (**a**) PCE-EtoE's text signcryption- Raspberry Pi; (**b**) PCE-EtoE's text un-signcryption- Raspberry Pi.

We performed signcryption and un-signcryption tests of differently sized picture messages on DELL computers and Raspberry Pi devices, respectively. Images with pixel sizes of 2500 × 25,000, 10,000 × 10,000, and 25,000 × 25,000 were selected for testing. Figures 8a and 9a show the calculation time for the signcryption of picture-type messages by PCE-EtoE, and the actual calculation cost is consistent with the theoretical analysis: $3E + P$. The change in signcryption time is only related to the size of the image and not to the number of attributes. The signcryption time for a 2500 × 25,000 picture is 0.08 s on a DELL computer and 1.4 s on a Raspberry Pi. Figures 8b and 9b show the calculation time for the un-signcryption of picture-type messages by PCE-EtoE, and the actual calculation cost is consistent with the theoretical analysis: $(4 + 2l_s)P$. For the same image size, the number of attributes and the un-signcryption time are linearly related. On a DELL computer, when the image pixel size is 2500 × 25,000 and the number of attributes is 100, the un-signcryption time is 0.65 s, and on a Raspberry Pi, it is 11 s.



(**a**) (**b**)

Figure 8. PCE-EtoE's un-signcryption and signcryption of picture-type messages in a Dell computer. (**a**) PCE-EtoE scheme's picture sigcryption-dell; (**b**) PCE-EtoE scheme's picture un-sigcryption-dell.



(**a**) (**b**)

Figure 9. PCE-EtoE's un-signcryption and signcryption of picture-type messages in Raspberry Pi. (**a**) PCE-EtoE scheme's picture sigcryption-Raspberry Pi; (**b**) PCE-EtoE scheme's picture un-sigcryption- Raspberry Pi.

In order to compare the actual performance of PCE-EtoE and related schemes, OMDAC-ABSC [22], NMAS-ABSC [23], and PCEA [13] were selected as comparison schemes. Since a single message in group chat communication is generally smaller than 5 KB, we tested the relevant algorithms for 5 KB messages on a Dell computer for all schemes. In Figure 10, the OURS label represents PCE-EtoE. As shown in Figure 10a,b, the signcryption's time cost of PCE-EtoE is much lower than that of other schemes, and the signcryption's time cost of a single message is less than 0.1 s, which is suitable for group chat communication scenarios. The un-signcryption's cost time of PCE-EtoE is slightly lower than that of other schemes. When the access policy attribute set by the sender is 50, the message un-signcryption's time cost of the receiver is less than 0.4 s, which is also suitable for group chat communication scenarios. As shown in Figure 10c,d, the time cost of CerVer and CerGen in the PCE-EtoE scheme is lower than that of PCEA.

To investigate the execution efficiency of each stage and step in the PCE-EtoE scheme, Figure 11 shows the percentage of each step in the total execution time. Statistics were collected for the percentage of the execution time of Setup, Key and Credential Generation, key agreement, Message Sending, Message Forwarding, and Message Receiving. The parameters in the experiment were as follows: *POL* = (Department A **and** manager)

or (Department B **and** supervisor). The size of the message sent by the sender was 5 KB, and the size of the group was 100 people.

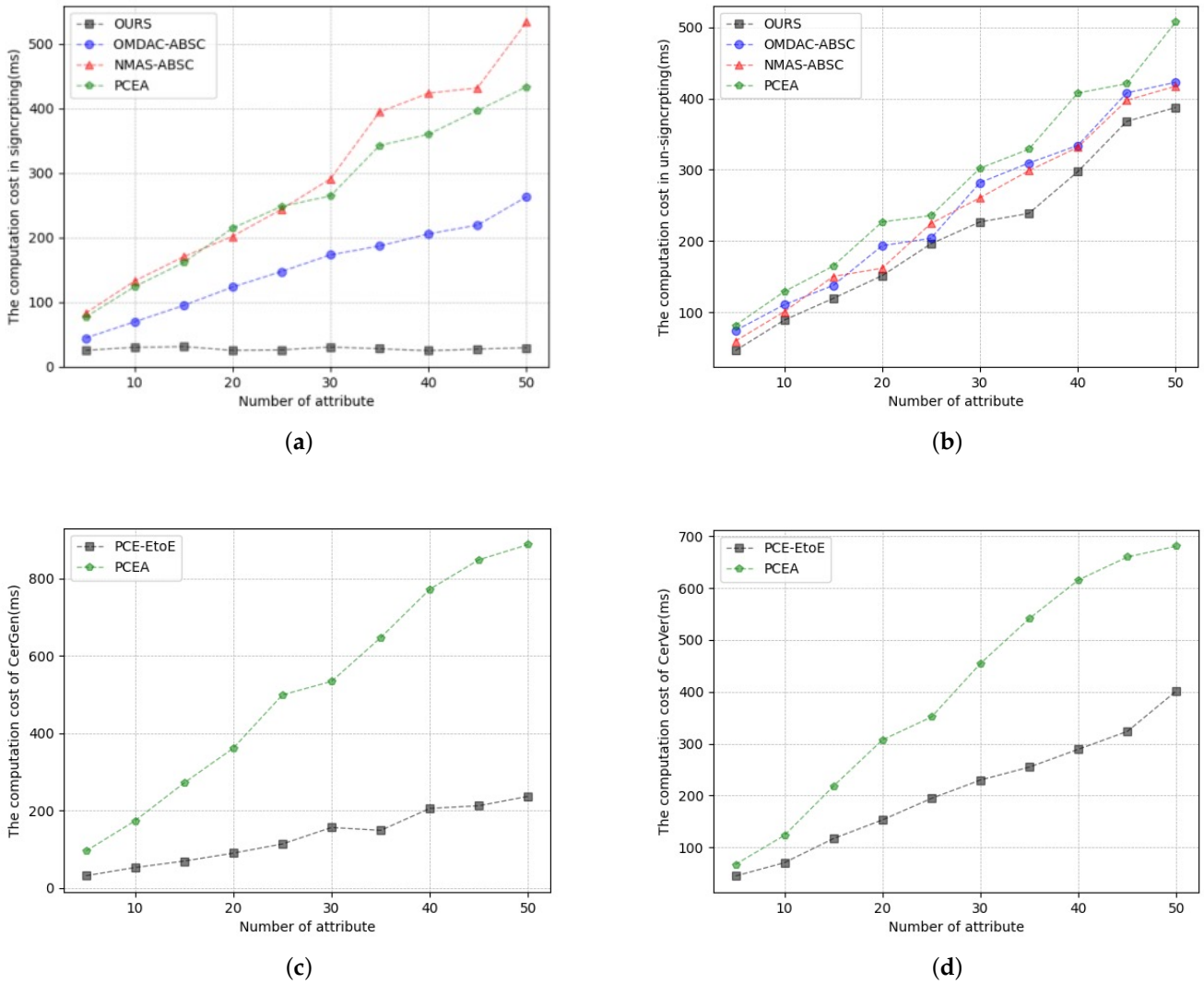


Figure 10. Comparison of PCE-EtoE with related schemes. (**a**) Comparison of PCE-EtoE's signcryption; (**b**) Comparison of PCE-EtoE's un-signcryption; (**c**) Comparison of PCE-EtoE's CreGen; (**d**) Comparison of PCE-EtoE's CerVer.

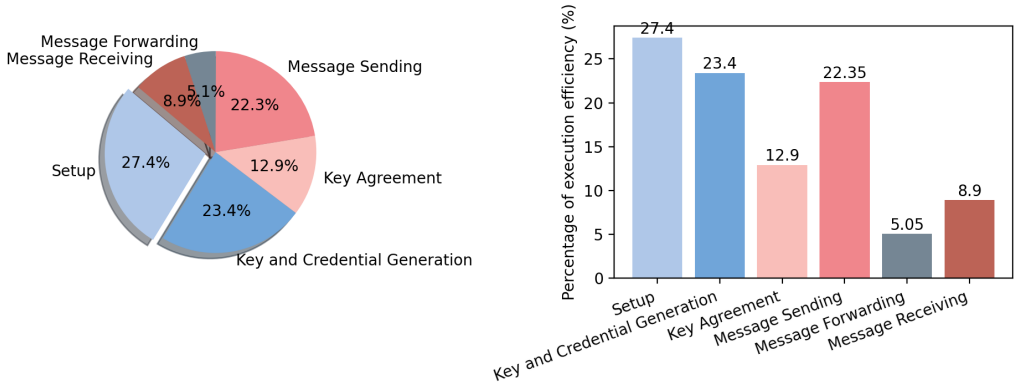
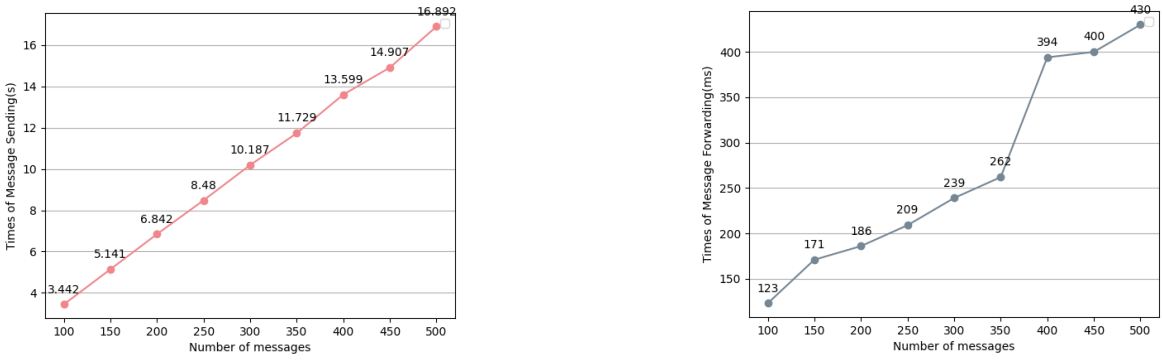


Figure 11. Percentage of the overall time for each step in PCE-EtoE.

As seen in Figure 11, the initialization step takes the largest percentage of the total execution time, about 27.4%. However, this step only needs to be executed once upon system startup. The message decryption time accounts for a relatively small percentage of 8.9%, while the Message Sending step takes about 22.35%. Attribute Code Generation in Message Sending consumes a significant amount of time. If a sender transmits multiple messages to the same receiver in a group chat, the Attribute Code Generation is executed only once.

Using the same parameters as in Figure 11, Figure 12 explores the time required for Message Sending and Message Forwarding by the operator when multiple messages are sent to the same receiver in a group chat, excluding network delay and user input time.



(**a**) (**b**)

Figure 12. The execution time of Message Sending and Message Forwarding for a group of receivers under the same POL in a group chat communication. (**a**) Message Sending; (**b**) Message Forwarding.

As shown in Figure 12a, the time required for sending messages increases linearly with the number of messages. When 500 messages are sent, the time taken is only 16.892 s. As indicated in Figure 12b, when the number of messages forwarded by the carrier server exceeds 350, the time increases rapidly, though it remains at the millisecond level. This does not place a significant computational burden on the communication opertor's server.

As shown in Table 6, to explore the theoretical time complexity of the PCE-ETOE scheme, the theoretical time complexity of each algorithm is analyzed. The time complexity of the algorithms Setup, KCGSKeyGen, UserKeyGen, and MesSen is $O(1)$. The time complexity of CreGen, CreVer, key agreement, MesForw, and MesRece is $O(n)$, which is within acceptable limits.

Table 6. The time complexity of the algorithm.

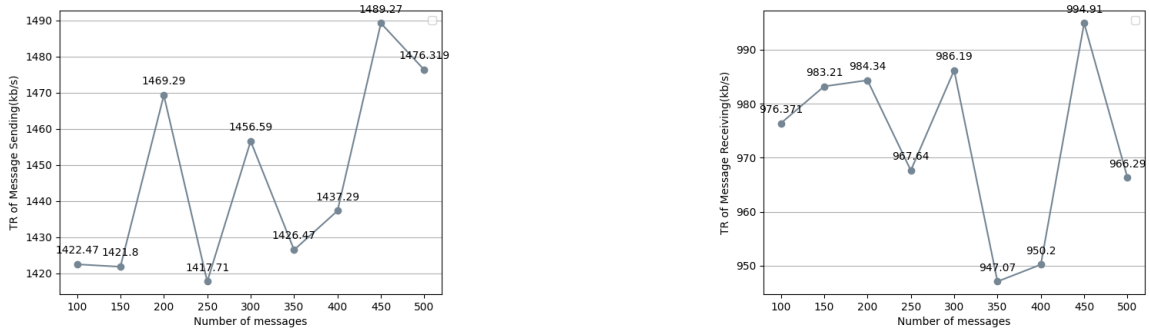
Algorithms	Setup	KCGSKeyGen	CreGen
Time complexity	$O(1)$	$O(1)$	$O(n)$
Algorithms	CreVer	UserKeyGen	Key Agreement
Time complexity	$O(n)$	$O(1)$	$O(1)$
Algorithms	MesSen	MesForw	MesRece
Time complexity	$O(1)$	$O(n)$	$O(n)$

We evaluate the transmission rate (TR) for both Message Sending and Receiving within our scheme in a real-time environment. The TR is calculated using the following formula:

$$Message\ Sending's\ TR = \frac{Message's\ Size}{Time\ of\ signcryption\ +\ Network\ delay}$$

$$Message\ Receiving's\ TR = \frac{Message's\ Size}{Time\ of\ un-signcryption\ +\ Network\ delay}$$

In the real-time environment, we transmit individual messages of no more than 50 KB and perform the experiment with 50 messages per test interval. As shown in Figure 13a, as the number of messages increases, the transmission rate (TR) for Message Sending remains between 1421.8 Kb/s and 1489.27 Kb/s. Similarly, Figure 13b shows that as the number of messages increases, the transmission rate (TR) for message reception remains between 947.07 Kb/s and 994.91 Kb/s. Both the transmission rate for Message Sending and the transmission rate for Message Receiving fall within a reasonable range.



(**a**) (**b**)

Figure 13. The TR of Message Sending and Message Receiving in real-time environment. (**a**) The TR of Message Sending in real-time environment; (**b**) The TR of Message Receiving in real-time environment.

8. Conclusions

To address the issues of the Communication Operator Server's unreliable behavior and message overload during group chat communication, we propose an attribute-based end-to-end policy-controlled signcryption scheme (PCE-EtoE), which features lightweight computational overhead, making it suitable for frequent group chat communications. Using LSSS to construct access structures ensures that only relevant recipients receive the information, thereby filtering out irrelevant information and mitigating message overload. When sending messages, the dual encryption mode of "signcryption + encryption" is used to prevent the communication operator's server from stealing group chat content. We provide theoretical proofs for the scheme's confidentiality, correctness, and resistance to communication operator theft. We evaluated the performance of the PCE-EtoE scheme against previous schemes through theoretical comparisons and actual simulations. Testing on different end devices confirmed that the PCE-EtoE scheme is more lightweight in terms of computational cost. However, the PCE-EtoE scheme does not support the fast retrieval of group chat messages. In the future, we aim to develop an efficient index structure within the PCE-EtoE scheme to facilitate the fast retrieval of historical group chat messages.

Author Contributions: Conceptualization, F.Y. and L.M.; methodology, F.Y. and L.M.; software, L.M. and Z.Z.; validation, F.Y. and L.M.; formal analysis, X.L. and W.Z.; investigation, F.Y.; writing—original draft, L.M.; supervision, F.Y. and D.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China (No. 62062016 and U21A20474), the Guangxi Science and technology project (GuikeAA22068070 and GuikeAD21220114), Jiangsu Provincial Key Laboratory of Network and Information Security (No. BM2003201-2022C4), Technical Service Project of China Guangdong Nuclear Power Engineering Co., Ltd. (No. 3201202200636). Finally, we thank the Center for Applied Mathematics of Guangxi (Guangxi Normal University).

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: Author Daicen Jiang was employed by the Southern Power Grid Supply Chain (Guangxi) Co., Ltd. Author Zhihua Zeng was employed by the China Nuclerar Power Engineering Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest. The Southern Power Grid Supply Chain (Guangxi) Co., Ltd. and China Nuclerar Power Engineering Co., Ltd. had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Siyal, R.; Long, J.; Asim, M.; Ahmad, N.; Fathi, H.; Alshinwan, M. Blockchain-Enabled Secure Data Sharing with Honey Encryption and DSNN-Based Key Generation. *Mathematics* **2024**, *12*, 1956. [CrossRef]
2. Alali, A.S.; Ali, R.; Jamil, M.K.; Ali, J.; Gulraiz. Dynamic S-Box Construction Using Mordell Elliptic Curves over Galois Field and Its Applications in Image Encryption. *Mathematics* **2024**, *12*, 587. [CrossRef]
3. Church, K.; De Oliveira, R. What's up with WhatsApp? Comparing mobile instant messaging behaviors with traditional SMS. In Proceedings of the 15th International Conference on Human-Computer Interaction with Mobile Devices and Services, Munich, Germany, 27–30 August 2013; pp. 352–361.
4. Sutikno, T.; Handayani, L.; Stiawan, D.; Riyadi, M.A.; Subroto, I.M.I. WhatsApp, viber and telegram: Which is the best for instant messaging? *Int. J. Electr. Comput. Eng.* **2016**, *6*, 909–914. . [CrossRef]
5. Ou, C.; Davison, R. Interactive or interruptive? Instant messaging at work. *Decis. Support Syst.* **2011**, *52*, 61–72. [CrossRef]
6. Wang, Z.; Ma, Z.; Luo, S.; Gao, H. Enhanced Instant Message Security and Privacy Protection Scheme for Mobile Social Network Systems. *IEEE Access* **2018**, *6*, 13706–13715. [CrossRef]
7. Afzal, A.; Hussain, M.; Saleem, S.; Shahzad, M.; Ho, A.; Jung, K. Encrypted Network Traffic Analysis of Secure Instant Messaging Application: A Case Study of Signal Messenger App. *Appl. Sci.* **2021**, *11*, 7789. [CrossRef]
8. Noriega, K.E.O.; Segura, J.E.G.; de los Santos, A.C.M. Security in the use of instant messaging applications for internal communication. *SCIÉNDO* **2022**, *25*, 219–227. [CrossRef]
9. Zhang, L.; Pan, G. Research on the Secure Communication Model of Instant Messaging. In Proceedings of the 6th International Conference on Computer Science and Application Engineering, Virtual, China, 21–23 October 2022; pp. 1–6.
10. Schillinger, F.; Schindelhauer, C. End-to-End Encryption Schemes for Online Social Networks. In *Security, Privacy, and Anonymity in Computation, Communication, and Storage*; Springer: Cham, Switzerland, 2019; pp. 133–146. [CrossRef]
11. Basem, O.; Ullah, A.; Hassen, H.R. Stick: An End-to-End Encryption Protocol Tailored for Social Network Platforms. *IEEE Trans. Dependable Secur. Comput.* **2023**, *20*, 1258–1269. [CrossRef]
12. Iduh, B.N. WhatsApp Network Group Chat Analysis Using Python Programming. *Int. J. Latest Technol. Eng. Manag. Appl. Sci. (IJLTEMAS)* **2020**, *9*, 1–5.
13. Thorncharoensri, P.; Susilo, W.; Mu, Y. Policy controlled system with anonymity. *Theor. Comput. Sci.* **2018**, *745*, 87–113. [CrossRef]
14. Liang, X.; Tang, Z.; Zhang, X.; Yu, M.; Zhang, X. Robust hashing with local tangent space alignment for image copy detection. *IEEE Trans. Dependable Secur. Comput.* **2023**, *21*, 2448–2460. [CrossRef]
15. Liang, X.; Tang, Z.; Huang, Z.; Zhang, X.; Zhang, S. Efficient hashing method using 2D-2D PCA for image copy detection. *IEEE Trans. Knowl. Data Eng.* **2021**, *35*, 3765–3778. [CrossRef]
16. Huang, Z.; Tang, Z.; Zhang, X.; Ruan, L.; Zhang, X. Perceptual image hashing with locality preserving projection for copy detection. *IEEE Trans. Dependable Secur. Comput.* **2021**, *20*, 463–477. [CrossRef]
17. Yu, M.; Tang, Z.; Zhang, X.; Zhong, B.; Zhang, X. Perceptual hashing with complementary color wavelet transform and compressed sensing for reduced-reference image quality assessment. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 7559–7574. [CrossRef]
18. Tang, Z.; Zhang, X.; Li, X.; Zhang, S. Robust image hashing with ring partition and invariant vector distance. *IEEE Trans. Inf. Forensics Secur.* **2015**, *11*, 200–214. [CrossRef]
19. Hong, H.; Xia, Y.; Sun, Z.; Liu, X. Provably secure attribute based signcryption with delegated computation and efficient key updating. *KSII Trans. Internet Inf. Syst.* **2017**, *11*, 2646.
20. Rao, Y.S.; Dutta, R. Efficient attribute-based signature and signcryption realizing expressive access structures. *Int. J. Inf. Secur.* **2016**, *15*, 81–109. [CrossRef]
21. Yu, G.; Cao, Z. Attribute-based signcryption with hybrid access policy. *Peer- Netw. Appl.* **2017**, *10*, 253–261. [CrossRef]

22. Xu, Q.; Tan, C.; Fan, Z.; Zhu, W.; Xiao, Y.; Cheng, F. Secure multi-authority data access control scheme in cloud storage system based on attribute-based signcryption. *IEEE Access* **2018**, *6*, 34051–34074. [CrossRef]
23. Zhao, Y.; Ruan, A.; Dan, G.; Huang, J.; Ding, Y. Efficient multi-authority attribute-based signcryption with constant-size ciphertext. In Proceedings of the 2021 IEEE Conference on Dependable and Secure Computing (DSC), Aizuwakamatsu, Fukushima, 30 January–2 February 2021; pp. 1–8.
24. Wang, C.; Huang, J. Attribute-based signcryption with ciphertext-policy and claim-predicate mechanism. In Proceedings of the 2011 Seventh International Conference on Computational Intelligence and Security, Sanya, China, 3–4 December 2011; pp. 905–909.
25. Cohn-Gordon, K.; Cremers, C.; Garratt, L.; Millican, J.; Milner, K. On ends-to-ends encryption: Asynchronous group messaging with strong security guarantees. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, Toronto, ON, Canada, 15–19 October 2018; pp. 1802–1819.
26. Gupta, S.; Sacchetti, T.; Crispo, B. End-to-end encryption for securing communications in industry 4.0. In Proceedings of the 2022 4th IEEE Middle East and North Africa COMMunications Conference (MENACOMM), Amman, Jordan, 6–8 December 2022; pp. 153–158.
27. Vidya Sagar, P.; Dhinesh, K.; Jayakumar, M.; Hemamalini, D. Hybrid Encryption through End to End in Messaging Service Applications. In Proceedings of the 2023 International Conference on Inventive Computation Technologies (ICICT), Lalitpur, Nepal, 26–28 April 2023; pp. 1139–1146.
28. Lewko, A.; Waters, B. Decentralizing attribute-based encryption. In Proceedings of the Annual international conference on the theory and applications of cryptographic techniques, Tallinn, Estonia, 15–19 May 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 568–588.