MBP: Multi-Channel Broadcast Proxy Re-Encryption for Cloud-Based IoT Devices

Sumana Maiti[*,a], Sudip Misra[b] and Ayan Mondal[c]

[a]*Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, 147004 India*
[b]*Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, 721302 India*
[c]*Department of Computer Science and Engineering, Indian Institute of Technology Indore, 453552 India*

ARTICLE INFO

Keywords:
Proxy re-encryption
Broadcast encryption
Rubinstein-Ståhl bargaining
Multi-channel encryption

ABSTRACT

The broadcast proxy re-encryption methods extend traditional proxy re-encryption mechanisms, where a single user shares the cloud data with multiple receivers. When the sender has different data to share with different sets of users, the existing broadcast proxy re-encryption schemes allow him/her to calculate distinct re-encryption keys for different groups in terms of additional computation time and overhead. To overcome these issues, we propose a scheme, named MBP, in IoT application scenario that allows the sender to calculate a single re-encryption key for all the groups of users, i.e., IoT devices. We use the multi-channel broadcast encryption concept in the broadcast proxy re-encryption method, so as single re-encryption key calculation and single re-encryption are done for different groups of IoT devices. It reduces the size of the security elements. However, it increases the computation time of the receiver IoT devices at the time of decryption of both the ciphertexts. To address this issue, we use the Rubinstein-Ståhl bargaining game approach. MBP is secure under a selective group chosen-ciphertext attack using the random oracle model. The implementation of MBP shows that it reduces the communication overhead from the data owner to the cloud server and from the cloud server to the receiver than existing algorithms.

1. Introduction

Proxy re-encryption (PrE) is a flexible method to maintain the secrecy of data, which is stored in a third party. A user stores his/her encrypted files to the cloud server [1, 2]. When the data owner needs to transmit the data to any receiver, s/he calculates a re-encryption key (r-key) using his/her secret key and recipient's identity, and the r-key is delivered to the proxy server which transforms the original ciphertext (o-text) to the re-encrypted ciphertext (r-text) and is shared with the receiver. If the data owner wants to delegate the data with multiple numbers of receivers, s/he uses a broadcast proxy re-encryption method (BPrE) [3], where a single r-key is generated for multiple receivers. The proxy transforms the o-text to the broadcast r-text and the result is finally broadcasted. If any user presents in the receiver group, s/he can recover the plaintext from the r-encrypted ciphertext and if s/he is not a member of the group, s/he cannot get the plaintext. On the other hand, multi-channel broadcast encryption (MBE) [4] is a concept that allows a sender to generate ciphertext for multiple groups instead of a single group. We introduce a new primitive MBP for multi-channel BPrE scheme when there are different groups of intended receivers for different messages. Our scheme generates a single r-key for different messages and sends this to the proxy server, which transforms the o-text to the r-text by one single re-encryption algorithm using the r-key. The intended receiver, which is an IoT device, gets the corresponding plaintext from the r-text using his/her secret key. Multi-channel BPrE reduces the size of ciphertext and r-key. But, at the same time, it grows the computation time of the receiver, which is a resource-constrained IoT device.

To solve this issue, we employed Rubinstein-Ståhl [5] bargaining approach to compute the optimal number of data sent to an IoT device using single encryption and re-encryption. The proposed scheme is applicable to different IoT applications, such as vehicular, healthcare, and agriculture.

1.1. Motivation

When the data owner wishes to transmit the same data with multiple receivers, s/he generates a r-key for a group of recipients instead of a single recipient using a broadcast proxy r-key. However, the existing BPrE schemes deal with a single receiver group and a single data. There may be many groups of recipients present. The data owner wants to delegate different data to different sets of receivers.

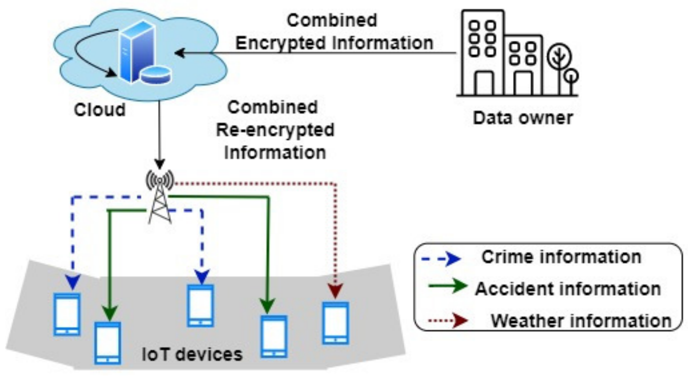

Figure 1: Motivation scenario

*Corresponding author
sumana.maiti@thapar.edu (Maiti); smisra@sit.iitkgp.ernet.in (Misra); ayanm@iiti.ac.in (Mondal)

Fig. 1 shows a motivation scenario of the proposed work in an IoT application scenario. The data owner has different types of data, for example, crime-related information, accident-related information, and weather information. S/he wants to share different types of data with different groups of IoT devices, such as crime-related information with a specific group of IoT devices, accident-related information with a group of IoT devices, and weather information with a group of IoT devices. If a separate r-key needs to be generated for each type of information, then it grows computation time and the required bandwidth of the network. If the data owner sends a single header for all the ciphertexts of different data, then the computation time and the required bandwidth can be reduced; however, it increases the decryption time of the IoT devices as it depends on the number of receivers present in the group. Hence, there is a requirement to find the optimal number of data to balance the size of the security elements and the computation time of the decryption. Moreover, using attribute-based proxy re-encryption [6], the secret key of the user is calculated based on the set of attributes of the user, and the data is encrypted based on a specific access policy. The proxy server converts the encrypted data for another set of access policies. If the user's set of attributes satisfies the access policy, then only s/he can recover the data from the re-encrypted ciphertext. Hence, we need to design a scheme to reduce the computation and communication costs when the data owner needs to share multiple data with multiple sets of receivers.

1.2. Contribution

In this work, we use the multi-channel broadcast encryption [4] in BPrE [7] to calculate one r-key for more than one group to reduce the size of r-key and r-text. We use bargaining game theory to reduce the computation time of the decryption of r-text. The use of a non-cooperative bargaining game approach balances the size of the security elements and the computation time of decryption. We prove that the MBP is indistinguishable selective group chosen-ciphertext attack (ind-sg-CCA) secure. The main innovations of the paper are listed below:

- In BPrE, if different data need to be shared with different groups of receivers, the data owner has to calculate a separate r-key for each data. In MBP, we calculate a single r-key for multiple data using the idea of multi-channel encryption in the BPrE algorithm. It reduces the size of security elements like r-key and r-text.
- The multi-channel encryption concept reduces the size of security elements, but it increases the decryption time of the receiver. In our scenario, the receivers are IoT devices. We find the optimal number of data that can be sent in single re-encryption using Rubinstein-Ståhl bargaining game approach to reduce the decryption time of r-text at the receiver end. We use the Backward Induction method to find the last period equilibrium and the Subgame Perfect Nash Equilibrium strategy to find the equilibrium of the initial time period.
- We model the ind-sg-CCA attack using the random oracle model and prove that MBP is ind-sg-CCA secure using the Decisional Diffie Hellman Exponent Assumption.
- We implement MBP to show how the MBP is more efficient than existing BPrE schemes.

1.3. Paper Organization

We review different related work in Section 2. Section 3 describes the preliminaries. Section 4 explains the problem statement. Section 5 discusses the game formulation. Section 6 defines MBP. Section 7 discusses MBP algorithm and the correctness of the algorithm. Section 8 explains the security analysis of MBP. In Section 9, we implement MBP and compare its performance with different existing schemes. Section 10 concludes the paper.

2. Related work

2.1. Broadcast proxy re-encryption

Proxy Re-encryption (PrE) [12] is used to share the encrypted data, which is stored in the third-party party server to a recipient to avoid the computation and communication cost of the data owner. PrE permits re-encryption for one recipient. There are different PrE schemes [13, 14, 15, 16, 17, 18, 19] present in IoT application scenario. If multiple recipients exist, the sender requires r-key generation multiple times. Hence, if a huge count of recipients is present, s/he needs to generate different r-keys for each recipient, which creates a headache for the data owner. If multiple recipients are present instead of a single recipient, then the data owner needs to calculate the r-key for each recipient. Broadcast PrE (BPrE) [20] is proposed to avoid the generation of r-key multiple times. Here, the data owner generates a single r-key for a group of recipients, and the proxy server converts the original ciphertext to a single r-text. If the recipient's identity is present in the group, then s/he can decrypt the r-text. A selective id chosen-plaintext attack secure conditional identity-based BPrE scheme is proposed in Ref. [8] for a cloud [21] email application. A condition is used to control the re-encryption power of the proxy server. A collusion-resistant and selective id chosen-ciphertext secure BPrE is proposed in Ref. [7]. A fine-grained conditional BPrE scheme is proposed in Ref. [10]. A multi-conditional BPrE scheme is proposed in Ref. [11]. Extending the work Ref. [8], a revocable BPrE is introduced in Ref. [22], where the power of revocation is given to the proxy server. Recently, a privacy-preserving scheme is proposed in Ref. [23], where each member of the recipient group cannot get any other user's identity. Zhang *et al.* [24] proposed to reduce the computation cost of the data sender. The authors introduced another entity, named data disseminator, to generate r-key on behalf of the data sender. Another objective of this work is to keep the identities of the vehicles private. Hence,

Table 1
Comparisons of MBP with the existing BPrE schemes

Scheme	Application area	R-key is generated by sender itself?	CPA secure ?	CCA secure ?	Single re-encryption multiple data?	Game-based solution to find optimal data?
CIBPRE [8]	Cloud email forwarding	✓	✓	✗	✗	✗
CPBRE-DS [9]	Cloud data sharing	✗	✓	✗	✗	✗
PBRE [7]	Cloud data sharing, distributed file system	✓	✓	✓	✗	✗
PRECISE [10]	Online social network	✗	✓	✗	✗	✗
MC-PBRE [11]	Cloud storage	✓	✓	✓	✗	✗
IBBPRE-VANET [11]	VANETs	✗	✓	✗	✗	✗
MBP	IoT application scenario	✓	✓	✓	✓	✓

pseudo-identity is used instead of the actual identity. The original sender cannot decrypt the original ciphertext as the broadcast key of the data disseminator is not shared with the data sender. TABLE 1 summarizes the comparisons of MBP with existing recent BPrE schemes.

2.2. Multi-channel broadcast encryption

MBE is proposed by Phan *et al.* [4] to address bandwidth and zapping time problems in broadcast encryption. This is a symmetric key-based system where the same global header is used for multiple channels. The scheme is selective id chosen-plaintext-attack secure under BDHE assumption. Acharya and Dutta [25] proposed two MBE schemes based on the public key system. The first scheme is based on the idea proposed by Kim *et al.* [26], and it is Chosen-plaintext-attack secure under the DBDHE assumption. The second method is based on the subtree method to make a partition of the subscribed users. Canard *et al.* [27] combined the idea of MBE and attribute-based encryption to reduce the size of the header of ciphertext.

2.3. Non-cooperative Rubinstein-Ståhl bargaining game

In a non-cooperative bargaining game [28]-[29], two players bargain for increasing their utilities, where one player's objective is opposite to another player. The bargaining comes to an agreement when one player cannot improve his/her utility without decreasing the others' utility. This gives a feeling of real-life bargaining. In Ref. [30], the authors proposed a Rubinstein-Ståhl bargaining game model to find out sharing rules for accessing the channels dynamically by radio-enabled nodes. An optimal subcarrier allocation problem is solved in Ref. [31] using the non-cooperative bargaining game. In Ref. [5], the optimal group size is estimated from the list of recipients in the broadcast group using the Rubinstein-Ståhl bargaining game.

3. Preliminaries

Bilinear map Let $\mathbb{G}_1$ and $\mathbb{G}_2$ are two multiplicative groups (MG) with prime order p and g is the generator of group group $\mathbb{G}_1$. The bilinear map [8] $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ has the following properties:

- $e(g^x, g^y) = e(g, g)^{xy}$, where $x, y \in \mathbb{Z}_p$.
- $e(g, g)$ is the generator of $\mathbb{G}_2$.

Bilinear Diffie Hellman Exponent Assumption (BDHE) Let $\mathbb{G}_1$ and $\mathbb{G}_2$ are two MGs. Let h is the generator of $\mathbb{G}_1$. The BDHE [32] is defined as the advantage of any adversary to find $e(h_{n+1}, G_z)$, where adversary has been given $(h, G_z, h_1, h_2, \ldots, h_n, h_{n+2}, \ldots, h_{2n})$, here $h_x = h^{\alpha^x}$. We can say that BDHE holds if the adversary has the negligible advantage to find out $e(h_{n+1}, G_z)$.

Decisional Diffie Hellman Exponent Assumption (DBDHE) Let $\mathbb{G}_1$ and $\mathbb{G}_2$ are two MGs of prime order p. Let h is the generator of $\mathbb{G}_1$. The DBDHE [32] is defined as the advantage of any adversary to distinguish $e(h_{n+1}, G_z)$ and B_1, where adversary has been given $(h, G_z, h_1, h_2, \ldots, h_n, h_{n+2}, \ldots, h_{2n}, e(h_{n+1}, G_z))$ and $(h, G_z, h_1, h_2, \ldots, h_n, h_{n+2}, \ldots, h_{2n}, B_1)$, where $h_x = h^{\alpha^x}$.

Rubinstein-Ståhl bargaining game This is a non-cooperative game between two players for T periods. The players play alternately. In each period, one player makes an offer, and the other player either accepts or rejects the offer. If the offer is accepted within a time period T, then it is finalized. Otherwise, the game goes to the next period. In the next period, the other player plays. In this way, the game continues. If, at T time period, the offer is rejected, then both players get zero utility.

Backward induction The game starts from the last time period and goes to the first time period in a backward way. Assuming the outcome of T time period, the player at time period $T-1$ time period plays such that the other player gets at least the utility, which s/he gets in time period T. In this way, the game continues from the last time period to the first time period.

Multi-channel broadcast encryption: A multi-channel broadcast encryption consists of the algorithms as follows:

- *Setup:* It takes the security parameter as input and generates the system secret key SSK and Encryption key EK.

- *KeyG:* It takes SSK and an identity i as inputs and generates secret key k_i as output.
- *EnCr:* It takes k sets $S = \{S_1, S_2, ..., S_m\}$ and EK as inputs and calculates header Hd and ephemeral keys $K = \{K_1, K_2, ..., K_m\}$, where K_i is used to encrypt the data for set of users S_i.
- *DecryptO:* It takes Hd, $S = \{S_1, S_2, ..., S_m\}$, a user i, and its secret key k_i. If user $i \in S_i$, then s/he can recover the corresponding ephemeral key K_i.

4. Problem statement

4.1. Target problem

In BPrE, if the data owner wishes to delegate the same data to multiple recipients, the sender generates a r-key for multiple recipients instead of a single recipient. The previous BPrE schemes deal with one single receiver group and one single data. There may be multiple groups of recipients present. The data owner needs to share one type of data with a group of recipients and other data with another group of recipients. If we apply the existing BPrE schemes to the solution, the sender needs to generate separate r-keys for separate groups of receivers, which increases the computation time and communication overhead on the sender side. Therefore, in this work, we propose a BPrE scheme using multi-channel broadcast encryption, where a single r-key is generated to share multiple different encrypted data with different groups of receivers.

Though multi-channel broadcast PrE reduces the size of the header in r-key, it increases the computation time on the receiver, which increases linearly with the number of data. Therefore, there is a need to find out the optimal number of data that balances the utility of the size of r-key, r-text, and the computation time of the receiver.

4.2. System model

The system model of an IoT application is shown in Fig. 2, which consists of the sender group S, receiver group $S' = \{S'_1, S'_2, ..., S'_k\}$, proxy server, and cloud server. Here, the receivers are considered as the IoT devices. The data owner user i from the group S stores data $\{M_1, M_2, ..., M_k\}$ to the cloud server in encrypted form. The data are encrypted for the group S. Whenever the data owner requires the data, s/he downloads the data and decrypts it. The data owner needs to calculate a r-key for group $\{S'_1, S'_2, ...S'_k\}$, so that users of a group S'_m get data M_m using their secret key. Here, the decryption time of the receiver increases as the decryption of r-text needs to consider all groups. Therefore, there is a need to reduce the decryption time without violating the idea of one encryption for multiple data.

4.3. Justification of bargaining game

The data owner and the receivers' goals are opposite to each other. To find the optimal number of data m from total data k, the data owner and the receiver have to come

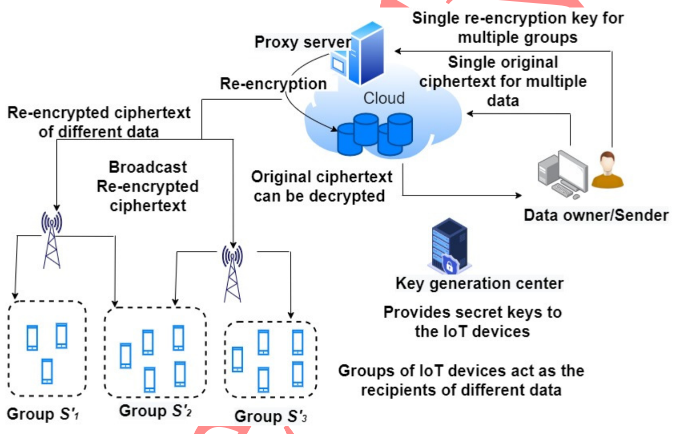
Single re-encryption key for multiple groups
Proxy server
Re-encryption
Cloud
Single original ciphertext for multiple data
Re-encrypted ciphertext of different data
Broadcast Re-encrypted ciphertext
Original ciphertext can be decrypted
Data owner/Sender
Key generation center
Provides secret keys to the IoT devices
Groups of IoT devices act as the recipients of different data
Group S'₁
Group S'₂
Group S'₃

Figure 2: Problem statement

to a solution where both players' utilities get balanced. Therefore, there is a requirement for a non-cooperative bargaining game so that one player cannot increase his/her utility without affecting the other's utility. In this work, we use the Rubinstein-Ståhl bargaining game as it gives a flavor of bargaining in real life, and both players' utilities get maximized.

4.4. Design goals

- Single r-key and single r-text need to be generated to share multiple data to the multiple target groups of receivers.
- There is a need to find the optimal number of data to balance the utilities of the data owner and the receiver.
- The scheme should be secure against the chosen ciphertext attack in the random oracle model.
- The malicious inside user should not discover the secret key of the other users.
- The receiver can not get the data if s/he is not present in the corresponding group of receivers.

5. Game Formulation

In this work, our objective is to share multiple data with multiple groups of recipients using a single r-key. However, if we consider all the data in a single re-encryption, the computation time and the communication overhead increase on the data owner side. Hence, we apply the Rubinstein-Ståhl bargaining game[5] to reduce the computation cost and communication overhead. The idea of the game formulation is borrowed from Ref. [5, 33, 30]. However, in Ref. [5], the objective was to find the optimal number of receivers from the total number of receivers present in a group. In this work, the objective is to find the count of plaintext that the data owner can encrypt in a single encryption to balance the utilities of both the data owner and the receivers.

The bargaining game is played between two players, which are the data owner (Which acts as the source) and the receiver of r-text (which acts as the destination). The game is played for $T = \{1, 2, ..., T\}$ time periods. In each odd time, the data owner (source) offers a value $m \subseteq T_d$, and the destination either accepts the offer or rejects the offer. If the offer is accepted, the data owner and the receiver get their respective utility for m. If the receiver rejects the offer m, the game continues for the next period $t+1$. Similarly, in each even time period, the destination makes the offer, and the source either accepts or rejects it. If any player rejects the offer at the last time T, then both source and destination get no utility.

5.1. Utility Function Formulation

The utility functions depend on the total number of data that needs to be sent, that is T_d. The number of data that can be sent at once is m, where $1 \leqslant m \leqslant T_d$. The game is played for T number of periods. $t = \{1, 2, ..., T\}$. The discount factor Δ is the discount factor of the utility. If the utility is calculated at t period of the game, then the discount factor of the source is Δ_s^t, and the discount factor of the destination is Δ_d^t.

Source utility The source utility function is calculated based on reducing the size of o-text, r-key, and r-text. The source utility function can be written as follows.

$$\mathrm{Utl}_s\left(T_d, m, t\right) = \frac{\left(1 - \frac{1}{m}\right) A_2 \Delta_s^t}{A_1 + A_2},$$

where A_1 and A_2 denote the count of group elements depending on the number of data m which is sent at once, and the count of group elements does not depend on m, respectively.

Destination utility The destination utility function depends on how much computation time of decrypting the r-text it has reduced. The destination utility function can be written as follows.

$$\mathrm{Utl}_d\left(T_d, m, t\right) = \frac{B_1\left(T_d - m\right)}{B_1 T_d + B_2} \Delta_d^t.$$

We consider only the bilinear pairing operation to calculate the computation time for decrypting the r-text, as it is the most expensive cryptographic operation. Here, B_1 and B_2 are the constants, where B_1 denotes the computation time depends on the number of data m which is sent at once and B_2 denotes the computation time which does not depend on m.

5.2. Objective Function

The utility of source Utl_s and utility of destination Utl_d are inversely proportional. Therefore, both the players' objectives are different in this scenario. The objective of this game is to find an optimal value of $m \subseteq T_d$ to balance both the source and destination's utility. The game is played for T number of time periods, where $t \in \{1, 2, ..., T\}$. If t is odd, then the source offers m_t, and the destination either accepts m_t or rejects it. Similarly, if t is even, then the destination offers m_t, and the source either accepts m_t or rejects it. If the offer m_t is accepted at time period t, then both players get their respective utilities. If the offer m_t is rejected, then the game continues for time period $t+1$. In this way, the game continues for T time period. If at time period T, the offer m_t is rejected, then both players get no utilities, and the game ends.

Here, we use the backward induction method to solve the problem. The game continues from the last period T^{th} and ends at 1^{st} time period. At the time period T, whoever the player is, always offers m_T in such a way that it gets its maximum utility and minimizes the other player's utility. The other player still accepts the offer m_T in the last period as rejection also gives no utility. Therefore, in the second last period, the player should offer m_{T-1} in such a way that another player accepts the offer and the game does not go for T period. In this way, the game continues for T to 0 period.

If at period $t+1$, source offers m_{t+1} to maximizes its utility $\mathrm{Utl}_s(T_d, m_{t+1}, t+1)$. Here, destination's utility is $\mathrm{Utl}_d(T_d, m_{t+1}, t+1)$. Therefore, in period t, the destination should offer m_t such that it maximizes its payoff and does not go to the next period $t+1$. The following conditions should be satisfied.

$\mathrm{Utl}_s(T_d, m_t, t) \geq \mathrm{Utl}_s(T_d, m_{t+1}, t+1)\Delta.$

Similarly, if destination offers m_{t+1} at time period $t+1$ to maximizes its utility $\mathrm{Utl}_d(T_d, m_{t+1}, t+1)$. Here, the utility of the source is $\mathrm{Utl}_s(T_d, m_{t+1}, t+1)$. Therefore, at time period t, the source should offer m_t such that it maximizes the source's utility; however, the destination should not reject the offer. Therefore, the following condition should be satisfied.

$\mathrm{Utl}_d(T_d, m_t, t) \geq \mathrm{Utl}_d(T_d, m_{t+1}, t+1)\Delta.$

5.3. Equilibrium Analysis

The game outcome m_T is in Nash equilibrium if it is accepted and no other offer m_{1T} can increase the utility of the offerer without affecting the other player's utility. In the last period, the offerer does not wish to increase the utility because other players may discard the offer. On the other hand, at the last period, the other player does not discard the offer as the result gives zero utility. Hence, the game outcome in the last period is in Nash equilibrium. Using backward induction, the game outcome of the first period of the game is found. It is noteworthy that the game always ends at the first period. The outcome of the game is calculated using backward induction, assuming that the game is played for T time period.

6. Defining MBP

6.1. MBP

MBP system consists of algorithms namely **Setup**, **KeyG**, **DataSelection**, **EnCr**, **RekeyG**, **ReEnCr**, **DecryptO**, and **DecryptR**. The key generation center runs **Setup** algorithm to generate system parameters and system secret key. It runs **KeyG** algorithm to compute the user's secret key. The data owner runs **DataSelection** algorithm to know the number of plaintexts that can participate in single encryption. Then, it runs **EnCr** algorithm to calculate the o-text of different plaintexts. The o-text can be decrypted by **DecryptO** algorithm. If the data owner wants to share different data with different groups of receivers, it generates a single r-key using **RekeyG** algorithm. The proxy server generates r-text using **ReEnCr** algorithm. The r-text can be decrypted by the **DecryptR** algorithm. The formal definition of MBP is defined as follows.

- **Setup:** This algorithm takes a security parameter as input and generates system secret key SSK and system parameter SP as outputs.
- **KeyG:** This algorithm takes SP, SSK, and identity i as inputs and calculates secret key k_i for user i.
- **DataSelection:** This algorithm takes total number of available plaintext T_d, game time period T, constants A_1, A_2, B_1, and B_2 as inputs. It outputs the number of data k that can be sent at a single encryption.
- **EnCr:** This algorithm takes k data M_1, $M_2 ..., M_k$, sender groups $S = \{S_1, S_2, ..., S_k\}$, data owner user i, $i \in S_m$, $\forall m \in \{1, 2, ..., k\}$, system parameter SP as inputs and generates ciphertext C for group G.
- **RekeyG:** This algorithm takes secret key of data owner k_i, system parameter SP, receiver group $S' = \{S'_1, S'_2, ..., S'_k\}$ as inputs and generates a r-key rk for group S' as output.
- **ReEnCr:** This algorithm takes the r-key rk, o-text C, system parameter SP, the sender group $S = \{S_1, S_2, ..., S_k\}$, and the data owner's identity i as inputs and generates the r-text C' as output.
- **DecryptO:** This algorithm takes o-text C, system parameter SP, user's identity i, secret key k_i, sender group $S = \{S_1, S_2, ..., S_k\}$ as inputs. It aborts if $i \notin S$ otherwise, it recovers M_m if $i \in S_m$, where $S_m \subset S$.
- **DecryptR:** This algorithm takes r-text C', system parameter SP, user's identity p, secret key k_p, receiver group $S' = \{S'_1, S'_2, ...S'_k\}$ as inputs. It aborts if $p \notin S'$; otherwise, it recovers M_m if $p \in S'_m$, where $S'_m \subset S'$.

7. MBP: The Proposed Scheme

7.1. in-sg-CCA Security

The in-sg-CCA is an attack between an adversary and a challenger. The MBP scheme is in-sg-CCA secure if the adversary wins the following game.

- **Init:** The adversary ad selects some groups $S^* = \{S^*_1, S^*_2, ..., S^*_m\}$ and an index k to challenge and sends these to challenger ch.
- **Setup:** The challenger ch runs the Setup algorithm and computes SP and SSK. SP is sent to ad and SSK is kept with challenger securely.
- **Query 1:** The adversary ad makes the following queries
  - *KeyG query (i)* If $i \in S^*_k$, ch aborts otherwise ch runs KeyG algorithm for user i and returns k_i to ad.
  - *RekeyG query (i, S'_1, S'_2, ..., S'_m)* The challenger ch runs RekeyG (i, S'_1, S'_2, ..., S'_m) where user i 's secret key k_i is generated by KeyG (i). Finally, the generated r-key rk is sent to ad. Adversary ad cannot query RekeyG query (i, S') and KeyG query (j) where $i \in S^*_k$ and $j \in S'$. Here $S' = \{S'_1, S'_2, ..., S'_m\}$.
  - *ReEnC query (i, S, S′, rk, C)* Here $S = \{S_1, S_2, ..., S_m\}$ and $S' = \{S'_1, S'_2, ..., S'_m\}$ The challenger runs ReEnC (i, S, S', rk, C) algorithm, where rk =RekeyG (i, S'_1, S'_2, ..., S'_m) and k_i =KeyG (i). Finally the r-text C' is sent to ad.
  - *DecryptO (i, C, S_1, S_2, ..., S_m, l)* The challenger ch runs KeyG (i) algorithm to generate k_i and runs DecryptO (i, C, S_1, S_2, ..., S_m, l, k_i) algorithm, where l is the group index. Finally, ch returns the result to ad. Here the only restriction is that ad cannot query DecryptO (i, C, S^*_1, S^*_2, ..., S^*_m, l).
  - *DecryptR (p, C′, S'_1, S'_2, ..., S'_m, l)* The challenger ch runs KeyG (p) algorithm to generate k_p and runs DecryptR (p, C', S'_1, S'_2, ..., S'_m, l, k_p) algorithm, where l is the group index. $p \in S'_l$ Finally ch returns the result to ad. Here the restriction is that ad cannot query DecryptR (p, S^*_1, S^*_2, ..., S^*_m, C', S'_1, S'_2, ..., S'_m, l)
- **Challenge:** Adversary ad outputs two equal length message set $\left(M^1_0, M^2_0, ..., M^m_0\right)$ and $\left(M^1_1, M^2_1, ...M^m_1\right)$ and sends these to ch, who chooses $b \in \{0, 1\}$ and runs EnC $\left(M^1_b, M^2_b, S^*_1, S^*_2, ..., S^*_m\right)$ and sends the result C^* to ad.
- **Query 2:** This phase is the same as Query 1 with the following restrictions.
  - The adversary cannot query ReEnC query(i, S^*_1, S^*_2, ..., S^*_m, S'_1, S'_2, ..., S^*_m, rk, C^*)
  - The adversary cannot query DecryptO (i, C^*, S^*_1, S^*_2, ..., S^*_m, l) for $i \in S^*_l$.
  - The adversary cannot query DecryptR (p, S^*_1, S^*_2, ..., S^*_m, C', S'_1, S'_2, ..., S'_m, l) for C' =ReEnC (i, S_1, S_2, ..., S_m, S'_1, S'_2, ..., S'_m, rk, C^*).
- **Guess:** The adversary guess b'. If $b' = b$, then ad wins the game. The advantage of ad is $Ad_{ad} = Pr\left[b' = b\right] - \frac{1}{2}$. MBP is in-sg-CCA secure if Ad_{ad} is negligible.

7.2. Methodology

We borrowed the MBE concept from Ref. [4] and the BPrE concept from Ref. [7] in MBP. We describe the full construction of MBP in this Section.

- **Setup:** This algorithm generates a bi-linear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. Here $\mathbb{G}_1$ and $\mathbb{G}_2$ are cyclic groups of prime order p. Let $a,\ b,\ c \in \mathbb{Z}_p$. It generates $g_i = g^{a^i}$ for $i = 1, 2, \ldots, n, n+2, \ldots, 2n$. It computes $v_1 = g^b$ and $v_2 = g^c$. It chooses a hash function $H : \mathbb{G}_2 \rightarrow \mathbb{G}_1$. Finally, it outputs the system secret key $SSK = \left(a, b, c\right)$ and system parameter $SP = \left(g, g_1, g_2, \ldots, g_n, g_{n+2}, \ldots, g_{2n}, v_1, v_2, H\right)$.

- **KeyG:** This algorithm takes system parameter SP, system secret key SSK, and identity i as inputs and calculates corresponding secret key $k_i = {g_i}^b$.

- **DataSelection:** This algorithm takes total number of available plaintext T_d, game time period T, constants A_1, A_2, B_1, B_2 as inputs. The algorithm works as following steps:
  1. if T is odd, then the offer $m_T = T_d$.
  2. if T is even then $m_T = 1$.
  3. for $\left(t = T - 1 \text{ to } 1\right)$
     (a) if t is odd, then offer m_t such that $Utl_s(T_d, m_t, t)$ is maximum among all $m_t = \{1, 2, ..., T_d\}$, but it should satisfy
     $Utl_d(T_d, m_t, t) \geq Utl_d(T_d, m_{t+1}, t)\Delta$.
     (b) if t is even, then offers m_t such that $Utl_d(T_d, m_t, t)$ is maximum among all $m_t = \{1, 2, ..., T_d\}$, but it should satisfy
     $Utl_s(T_d, m_t, t) \geq Utl_s(T_d, m_{t+1}, t+1)\Delta$
  4. The result is $m = m_1$.

- **EnC:** This algorithm takes plaintexts $M_1, M_2, ... M_m$, data owner group $S = \{S_1, S_2, ..., S_m\}$, system parameter SP as inputs. It randomly chooses $t_1, t_2, ..., t_m \in \mathbb{Z}_p$ and calculates $T_i = g^{t_i}$ $\forall i \in \{1, 2, ..., m\}$ and $C_{0i} = M_i e\left(g_1, g_n\right)^{t_i}$ $\forall i \in \{1, 2, ..., m\}$. It calculates $C_1 = \prod_{l=1}^{m} \left(v_1 \prod_{j \in S_l} g_{n+1-j}\right)^{t_l}$ and $C_{2i} = {v_2}^{t_i}$ $\forall i \in \{1, 2, ..., m\}$. Finally, it outputs the o-text C, where $C = \Big(\{T_1, T_2, ..., T_m\}\{C_{01}, C_{02}, ..., C_{0m}\}, C_1, \{C_{21}, C_{22}, ..., C_{2m}\}\Big)$.

- **RekeyG:** This algorithm takes secret key of data owner k_i of user $i \in S_i$ and $S_i \subset S$, system parameter SP, receiver groups $S' = \{S'_1, S'_2, ..., S'_m\}$ as inputs. It chooses random values $s_1, s_2, .., s_m \in \mathbb{Z}_p$ and $w_1, w_2, ... w_m \in \mathbb{Z}_p$. Then it computes $r_{0i} = k_i {v_2}^{s_i}$, $Q_i = e\left(g_1, g_n\right)^{w_i}$, $W_1 = g^{w_1}$, $r_{1i} = H\left(Q_1\right) g^{s_i}$ $\forall i \in \{1, 2, ..., m\}$.
  Then, it calculates $r_2 = \prod_{l=1}^{m} \left(v_1 \prod_{j \in S'_l} g_{n+1-j}\right)^{w_l}$.
  Finally, the algorithm outputs the r-key rk a $rk = \Big(\{r_{01}, r_{02}, ..r_{0m}\}, \{r_{11}, r_{12}, ... r_{1m}\}, \{W_1, W_2, ... W_m\}, r_2\Big)$.

- **ReEnC:** This algorithm takes the r-key rk, o-text C, system parameter SP, the sender group $S = \{S_1, S_2, ..., S_m\}$, and the data owner's identity i as inputs. It computes C_{Rq}
  $$= C_{0q} e\Big(r_{0q} \prod_{j \in S_q j \neq i} g_{n+1-j+i}, T_q\Big) \frac{\prod\limits_{l=1, l \neq m}^{k} e\Big(r_{0q} \prod\limits_{j \in S_l} g_{n+1-j+i}, T_l\Big)}{e\Big(g_i, C_1\Big)}$$
  $\forall q \in \{1, 2, ..., m\}$.
  Finally it outputs the r-text $C_R = \Big(\{C_{R1}, C_{R2}, ..., C_{Rm}\}, \{W_1, W_2, ..., W_m\}, \{r_{11}, r_{12}, ..., r_{1m}\}, \{C_{21}, C_{22}, ..., C_{2m}\}, r_2\Big)$.

- **DecryptO:** This algorithm takes o-text C, system parameter SP, user's identity i, secret key k_i, sender group $S = \{S_1, S_2, ..., S_m\}$ as inputs. If $i \in S_q$, then it calculates
  $$K_q = \frac{e\Big(g_i, C_1\Big)}{e\Big(k_i \prod_{j \in S_q, j \neq i} g_{n+1-j+i}, T_q\Big) \prod_{l=1, l \neq q}^{l=m} e\Big(k_i \prod_{j \in S_l} g_{n+1-j+i}, T_l\Big)}$$
  and then gets $M_q = \frac{C_{0q}}{K_q}$. If $i \notin S_q$, the algorithm aborts.

- **DecryptR:** This algorithm takes r-text C', system parameter SP, user's identity p, secret key k_p, receiver group $S' = \{S'_1, S'_2, ..., S'_m\}$ as inputs. If $p \in S'_q$, then it calculates K'_q
  $$= \frac{e\Big(g_p, r_2\Big)}{e\Big(k_p \prod_{j \in S'_q, j \neq p} g_{n+1-j+p}, W_m\Big) \prod_{l=1, l \neq q} e\Big(k_p \prod_{j \in S'_l} g_{n+1-j+p}, W_l\Big)}$$
  and then calculates $g^{s_q} = \frac{r_{1q}}{H\left(K'_q\right)}$, Then it gets $M_q = \frac{C_{Rq}}{e\Big(C_{2q}, g^{s_q}\Big) \prod_{l=1, l \neq q}^{l=m} e\Big(C_{2l}, g^{s_q}\Big)}$. If $p \notin S'_q$, the algorithm aborts.

7.3. Correctness

The following theorems prove the correctness of the o-text and the r-text of MBP.

Theorem 1. *If the o-text $C = \Big(\{T_1, T_2, ..., T_k\}\{C_{01}, C_{02}, ..., C_{0k}\}, C_1, \{C_{21}, C_{22}, ..., C_{2k}\}\Big)$ is calculated by EnC($\{M_1, M_2, ..., M_k\}\{S_1, S_2, ..., S_k\}, SP$), the secret key of user i is k_i is calculated by KeyG(i, SSK), where $i \in S_m$ and $m \in \{1, 2, ..., k\}$, then DecryptO(i, k_i, C, SP) algorithm always gives the correct plaintext M_m if $i \in S_m$.*

Proof. The correctness of the o-text of our scheme is proved as follows:
If $i \in G_m$, then it calculates the key
$$K_m = \frac{e\Big(g_i, C_1\Big)}{e\Big(k_i \prod_{j \in S_m, j \neq i} g_{n+1-j+i}, T_m\Big) \prod_{l=1, l \neq m}^{k} e\Big(k_i \prod_{j \in S_l} g_{n+1-j+i}, T_l\Big)}$$
$$= \frac{e\Big(g^{a^i}, \prod_{l=1}^{k} \left(v_1 \prod_{j \in S_l} g_{n+1-j}\right)^{t_l}\Big)}{e\Big({g_i}^b \prod_{j \in S_m, j \neq i} g_{n+1-j+i}, T_m\Big) \prod_{l=1, l \neq m}^{l=k} e\Big({g_i}^b \prod_{j \in S_l} g_{n+1-j+i}, T_l\Big)}$$
$$= \frac{e\Big(g^{a^i}, g^b\Big)^{\left(t_1 + \ldots + t_l\right)} \prod_{l=1}^{l=k} e\Big(g^{a^i}, \prod_{j \in S_l} g_{n+1-j}\Big)^{t_l}}{e\Big({g_i}^b \prod_{j \in S_m, j \neq i} g_{n+1-j+i}, g^{t_m}\Big) \prod_{l=1, l \neq m}^{l=k} e\Big({g_i}^b \prod_{j \in S_l} g_{n+1-j+i}, g^{t_l}\Big)}$$

$= e\left(g^{a^i}, g_{n+1-i}\right)^{t_m} = e\left(g, g_{n+1-i+i}\right)^{t_m}$
$= e\left(g, g_{n+1}\right)^{t_m}$.

Now, it calculates $\frac{C_{01}}{e\left(g,g_{n+1}\right)^{t_m}} = \frac{M_m e\left(g_1,g_n\right)^{t_m}}{e\left(g,g_{n+1}\right)^{t_m}} = M_m$.
Therefore, if user $i \in S_m$ and $S_m \subseteq S$, then s/he gets the corresponding plaintext M_m. □

Theorem 2. *If the r-text $C_R = \Big(\{C_{R1}, C_{R2}, ..., C_{Rk}\}, \{W_1, W_2, ..., W_k\}, \{r_{11}, r_{12}, ..., r_{1k}\}, \{C_{21}, C_{22}, ..., C_{2k}\}, r_2\Big)$ is calculated by ReEnC(rk, C, SP), the r-key rk is calculated by RekeyG(k_i, SP, S'), secret key k_i is calculated by KeyG(i, SSK), o-text C is calculated by EnC($\{M_1, M_2, ..., M_k\}\{S_1, S_2, ..., S_k\}$, SP), and secret key of user p is k_p, which is calculated by KeyG(p, SSK), the DecryptR(C', S', SP, p, k_p) gives the correct plaintext M_m, if $p \in S'_m$ and $S'_m \subset S'$.*

Proof. If $p \in S'_m$, then it calculates

$$K'_1 = \frac{e\left(g_p,r_2\right)}{e\left(k_p \prod_{j\in S'_m, j\neq p} g_{n+1-j+p}, W_m\right) \prod_{l=1,l\neq m}^{l=k} e\left(k_p \prod_{j\in S'_l} g_{n+1-j+p}, W_l\right)}$$

$$= \frac{e\left(g^{a^p}, \prod_{l=1}^{k} \left(v_1 \prod_{j\in S'_l} g_{n+1-j}\right)^{w_l}\right)}{e\left(k_p \prod_{j\in S'_m, j\neq p} g_{n+1-j+p}, g^{w_m}\right) \prod_{l=1,l\neq m}^{k} e\left(k_p \prod_{j\in S'_l} g_{n+1-j+p}, g^{w_l}\right)}$$

$$= \frac{e\left(g^{a^p}, \prod_{l=1}^{l=k} \left(v_1 \prod_{j\in S'_l} g_{n+1-j}\right)^{w_l}\right)}{e\left(k_p \prod_{j\in S'_m, j\neq p} g_{n+1-j+p}, g^{w_m}\right) \prod_{l=1,l\neq m}^{l=k} e\left(k_p \prod_{j\in S'_l} g_{n+1-j+p}, g^{w_l}\right)}$$

$= e\left(g^{a^p}, g_{n+1-p}\right)^{w_m} = e\left(g, g_{n+1}\right)^{w_m}$. Then, it calculates

$$\frac{r_{1m}}{H\left(K'_m\right)} = \frac{H\left(e\left(g_1,g_n\right)^{w_m}\right) g^{s_m}}{H\left(e\left(g,g_{n+1}\right)^{w_m}\right)} = g^{s_m}.$$

Next, it calculates $\frac{C_{Rm}}{e\left(C_{2m},g^{s_m}\right) \prod_{l=1,l\neq m}^{l=k} e\left(C_{2l},g^{s_m}\right)}$

$$= \frac{C_{0m} e\left(r_{0m} \prod_{j\in S_m j\neq i} g_{n+1-j+i}, T_m\right) \prod_{l=1,l\neq m}^{k} e\left(r_{0m} \prod_{j\in S_m} g_{n+1-j+i}, T_l\right)}{e\left(g_i,C_1\right) e\left(C_{2m},g^{s_m}\right) \prod_{l=1,l\neq m}^{l=k} e\left(C_{2l},g^{s_m}\right)}$$

$= M_m$.
If $p \in S'_m$, then the result is M_m; Otherwise, it outputs error. □

8. Security analysis

Theorem 3. *The proposed scheme MBP is ind-sg-CCA secure under DBDHE assumption.*

Proof. Challenger ch takes DBDHE parameters $\left(g, R_1, R_2, ..., R_m, g_1, \dots, g_n, g_{n+2}, \dots, g_{2n}, B_1, B_2, ..., B_m\right)$. Here B_l is either $e\left(g_{n+1}, R_l\right)$ or a random element of group $\mathbb{G}_2$. Here $g_i = g^{a^i}$ for $a \in \mathbb{Z}_p$. Adversary ad selects two groups $S^*_1, S^*_2, ..., S^*_m$, and an index k to challenge and sends these to challenger ch.

- **Setup:** The challenger ch generates system parameters SP. Challenger ch first chooses $c \in \mathbb{Z}_p$ and sets the system parameter $SP = \left(g_1, g_2, \dots, g_n, g_{n+2}, \dots, g_{2n}, v_1, g^c\right)$. Here $v_1 = g^u\left(\prod_{j\in S^*_k} g_{n+1-j}\right)^{-1}$, where ch chooses random $u \in \mathbb{Z}_p$. Challenger ch sends SP to ad. We do not include the hash function H in the SP, because H is used in **RekeyG** and **DecryptR** algorithms. In the security analysis, these algorithms act as random oracles. The adversary sends the inputs of RekeyG algorithm to RekeyG query and gets the output. Similarly, the adversary sends the inputs of DecryptR algorithm to DecryptR query and gets the output. Therefore, the adversary does not need the hash function H.

- **Query 1:** The adversary ad makes the following queries

  – *KeyG query(i)*: If $i \in S^*_k$, ch aborts, otherwise ch searches $Table_{key}$ whether k_i exists in $Table_{key}$ or not. If k_i exists in the table, then ch returns k_i as result, otherwise ch computes $k_i = {g_i}^u\left(\prod_{j\in S^*_k} g_{n+1-j+i}\right)^{-1} = g^{ua^i}\left(\prod_{j\in S^*_k} g_{n+1-j}\right)^{-a^i} = {v_1}^{a_i}$. It should be noted that v_1 is calculated as $v_1 = g^u\left(\prod_{j\in S^*_k} g_{n+1-j}\right)^{-1}$. Therefore, We can write $k_i = {g_i}^b$ as our original keyG algorithm. Here, the challenger ch does not need to know the a and b. Here, ch returns k_i to ad and stores the value in $Table_{key}$.

  – *RekeyG query(i, S'_1, S'_2, ..., S'_m)*: The challenger ch checks whether there is a tuple $\left(j, k_j\right)$ exists in $Table_{key}$ or not, where $i \in S^*_k$ and $j \in S'$, where $S' = \{S'_1, S'_2, ..., S'_m\}$. If it exists, ch aborts. Otherwise, ch searches whether there is any tuple $\left(i, S', rk\right)$ exists in $Table_{rekey}$ or not. If it presents, ch outputs rk as a result. Otherwise ch runs KeyG query(i) to get k_i and then calculates rk using RekeyG(i, S') algorithm and adds(i, k_i) to $Table_{key}$ and (i, S'_1, S'_2, ..., S'_m, rk) to $Table_{rekey}$.

  – *ReEnC query(i, S, S'_1, S'_2, ..., S'_m, C)*: The challenger searches whether the tuple (i, S, S'_1, S'_2, ..., S'_m, C, C_R) exists in $Table_{reenc}$ or not. If it exists, ch returns C_R as a result. Otherwise, ch searches whether the tuple (i, S'_1, S'_2, ..., S'_m, rk) exists in $Table_{rekey}$ or not. If it presents, ch generates C_R using ReEnC(i, S_1, S_2, S'_1, S'_2, ..., S'_m, C, rk) algorithm, sends C_R as result, and stores the tuple (i, S, S'_1, S'_2, ..., S'_m, C, C_R) to $Table_{reenc}$. Otherwise, ch first issues a query RekeyG query(i, S'), where $S' = \{S'_1, S'_2, ..., S'_m\}$, gets the r-key rk, and then runs ReEnC(i, S, S'_1, S'_2, ..., S'_m, C, rk) algorithm. The r-key rk is stored to $Table_{rekey}$, and the r-text C_R is stored to $Table_{reenc}$. Finally, the result C_R is sent to ad.

  – *DecryptO(i, C, S_1, S_2, ..., S_m, l)*: The challenger ch checks whether there is any tuple (i, k_i) exists in $Table_{key}$ or not. If it presents, ch uses k_i to recover the message. Otherwise, the challenger ch runs KeyG(i) algorithm to generate k_i and runs DecryptO(i, C, S_1, S_2, ..., S_m, l, k_i) algorithm, where l is the group index. Here, $l \in \{1, 2, ..., m\}$. Finally, ch returns the result to ad. Here the only restriction is that ad cannot query DecryptO(i, C, S_1^*, S_2^*, ..., S_m^*, l).

- *DecryptR(p, S, $C^{'}$, $S_1^{'}$, $S_2^{'}$, ..., $S_m^{'}$, l)*: The challenger ch checks whether there is any tuple (p, k_p) exists in $Table_{key}$ or not. If it presents, ch uses k_p to recover the message. Otherwise, the challenger ch runs KeyG(p) algorithm to generate k_p and runs DecryptR(p, $C^{'}$, $S_1^{'}$, $S_2^{'}$, ..., $S_m^{'}$, l, k_p) algorithm, where l is the group index. Finally, ch returns the result to ad. Here the restriction is that ad cannot query DecryptR(p, S_1^*, S_2^*, ..., S_m^*, $C^{'}$, $S_1^{'}$, $S_2^{'}$, ..., $S_m^{'}$, l)

- **Challenge:** Adversary ad outputs two equal length message set M_0 and M_1 and sends these to ch. The challenger chooses $b \in \{0, 1\}$. The ch calculates for $l \in \{1, 2, ..., m\} \setminus k$ as $C_{0l}^* = M_b B_l$. Here, $T_l^* = g^{t_l}$. Only T_k^* is calculated as $T_k^* = \frac{\prod_{l=1}^{l=m} R_1}{\prod_{l=1 l\neq k}^{l=m} T_l^*}$. It should be noted that to generate session keys, ch computes all values of T_l^* for all $l \in \{1, 2, ..., m\} \setminus k$. Then s/he calculates T_k^*. It should be noted that $R_l = g^{t^l}$. The challenger calculates $C_{1l}^* = {T_k}^{*u} \prod_{l=1,l\neq k}^{l=m} \left({T_l}^{*u} \frac{\prod_{j\in S_l^*} T_{n+1-j}^*}{\prod_{j\in S_k^*} T_{n+1-j}^*} \right)$ $= g^{t_k u} \prod_{l=1,l\neq k}^{l=m} \left(g^{t_l u} \frac{\prod_{j\in S_l^*} g_{n+1-j}}{\prod_{j\in S_k^*} g_{n+1-j}} \right)^{t_l}$ $= v_1 \left(\prod_{j\in S_k^*} g_{n+1-j} \right)^{t_k} \prod_{l=1,l\neq k}^{l=m} \left(v_1 \prod_{j\in S_l} g_{n+1-j} \right)^{t_l}$ $= \prod_{l=1}^{l=m} \left(v_1 \prod_{j\in S_l} g_{n+1-j} \right)^{t_l}$. Then ch calculates $C_{2l}^* = (g^c)^{t_l}$. Now, challenger ch returns $C^* = \left(C_{0l}^*, T_l^*, C_1^*, C_{2l}^* \ \forall l = \{1, 2, ..., m\} \right)$ to ad. If $B_k = e(g_{n+1}, R_k)$, then $C_{k0}^* = M_b e(g_{n+1}, R_k)$, otherwise, B_k is a random element.

- **Query 2:** This phase is the same as Query 1 with the following restrictions

  - The adversary cannot query ReEnC query(i, S_1^*, S_2^*, ..., S_m^*, $S_1^{'}$, $S_2^{'}$, ..., $S_m^{'}$, rk, C^*)
  - The adversary cannot query DecryptO(i, C^*, S_1^*, S_2^*, .., S_m^*, l) for $i \in S_l^*$ and $l \in \{1, 2, ..., m\}$.
  - The adversary cannot query DecryptR(p, S_1^*, S_2^*, ..., S_m^*, $C^{'}$, $S_1^{'}$, $S_2^{'}$, ..., $S_m^{'}$, l) for $C^{'}$ =ReEnC(i, S_1, S_2, ..., S_m, $S_1^{'}$, $S_2^{'}$, ..., $S_m^{'}$, rk, C^*).

- **Guess:** Adversary ad guesses $b^{'}$. If $b^{'} = b$, the challenger ch returns $\mathbb{B} = 0$, which means B_k is actual value. Challenger ch outputs $\mathbb{B} = 1$ means the B_k is random values. $Pr[\mathbb{B} = 0] = \frac{1}{2}$. If B_k is actual value, then $| \ Pr[\mathbb{B} = 1] - 1 \ | = Ad_{ad} \geqslant \epsilon$. Therefore, the challenger has atleast ϵ advantage to solve DBDHE problem in $\mathbb{G}$. MBP is ind-sg-CCA secure if Ad_{ad} is negligible.

□

Table 2
Experimental Setup

Hardware	Intel Core i3-10110U CPU@2.10GHz
OS	Ubuntu 16.04 LTS
Compiler	gcc-5.4.0
Program Library	pbc-0.5.14 [34]
Curve	$y^2 = x^3 + x$
Base Field	512 bits
Group Order	160 bits
DLog Security	1024 bits

Table 3
Simulation Parameter

Parameter	Value
Number of data	30 to 70 and 50 to 150
Number of receivers	1 to 50
Bandwidth	100 Mbps
Data rate	50 Mbps
Power consumption of receiver	22.2 MW
Δ_s	0.8
Δ_d	0.8
Constants A_1, A_2	$A_1 = 2$, $A_2 = 3$
Constants B_1, B_2	$B_1 = 3$, $B_2 = 4$
Time period T	15

9. Performance Analysis

9.1. Experimental Setup

Table 2 shows the experimental setup, including hardware, OS, compiler, and program library. We implement MBP to show its efficiency of the required time for different algorithms, the communication overhead, and the transmission delay than existing BPrE schemes.

9.2. Benchmarks

The performance of MBP is compared with two recent schemes — CIBPRE [8], RIB-BPRE [22], and PBRE [7] schemes. CIBPRE is an efficient conditional BPrE scheme for cloud email systems, where the size of r-key and r-text are constant. On the other hand, RIB-BPRE is another conditional BPrE scheme, where the proxy server has the power to revoke any existing recipient from the group of recipients. Here, the size of the r-key increases with the increase of the count of recipients in the group. PBRE is a CCA secure BPrE scheme to share cloud data.

9.3. Simulation parameter

The simulation parameters are shown in Table 3. We vary the number of data from 10 to 50 in Figs. 3, 4, 5, and

6. We fix the number of receivers in each receiver group as 40. We vary the number of data from 30 to 70 in Fig. 10. We consider a single data and vary the number of receivers from 10 to 50 in Figs. 8 and 9. The bandwidth of the network is considered as 100 Mbps as we consider a wide area network. The discount factors Δ_s and Δ_d are selected as 0.8. The constants A_1 and A_2 are selected as the number of group elements, which depend on the value m, and the number of group elements, which do not depend on m, respectively. On the other hand, the destination constants B_1 and B_2 are selected as the count of expensive operations in decryption, which depends on the value m and the count of expensive operations in decryption, which does not depend on the value m, respectively.

9.4. Performance metrics

The performance of MBP is shown based on the following metrics.

Communication Overhead from sender to third party The sender stores o-text to the third party. Later, s/he generates r-key and sends it to the third-party server. Hence, the communication overhead from the sender to the third party is measured by the size of the o-text and r-key.

Communication Overhead from third party to receiver The third party broadcasts the r-text. The receiver receives the r-text and decrypts it. Therefore, the communication overhead from the third party to the receiver depends on the size of the r-text.

RkeyG time The RkeyG time is measured by the required time to generate the required r-keys.

DecryptR time The DecryptR time is calculated by the required time to decrypt the corresponding r-text.

Transmission delay The transmission delay from the data owner to the cloud server is measured as the amount of time to push the o-text and r-key to the network link. Similarly, the transmission delay from the cloud server to the receiver is calculated as the amount of time to push the r-text to the network link.

9.5. Results and Discussion

Comparisons of communication overhead Fig. 3 shows the rate of increase of the communication overhead from the sender to the third party of the MBP scheme is less than the rate of increase of communication overhead of RIB-BPRE, CIBPRE, and PBRE schemes. This is because all the elements of the o-text and r-key of RIB-BPRE, CIBPRE, and PBRE depend on the number of data. It needs to generate separate o-texts and r-keys in these schemes. But in the case of MBP, all the elements of o-text and r-key do not depend on the number of data it needs to share.

Fig. 4 shows the comparisons of the communication overhead from the third party to the receiver in the MBP,

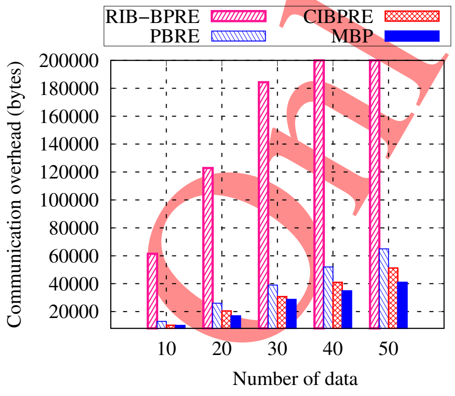

Figure 3: Communication overhead from sender to cloud server

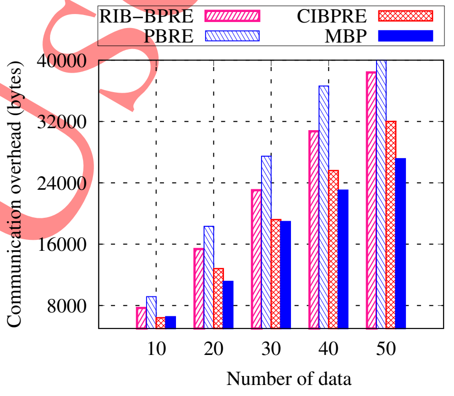

Figure 4: Communication Overhead from cloud server to receiver

RIB-BPRE, CIBPRE, and PBRE schemes. The rate of increase of the communication overhead in the MBP scheme is less than the rate of increase of the communication overhead in the RIB-BPRE, CIBPRE, and PBRE schemes. All the elements of the r-text in MBP do not depend on the number of data it needs to share, but in the case of RIB-BPRE, CIBPRE, and PBRE, all the elements of r-text depend on the number of data it needs to share.

Comparisons of transmission delay Fig. 5 shows the comparisons of the transmission delay from the data owner to the cloud server in the MBP, RIB-BPRE, CIBPRE, and PBRE schemes. The rate of increase of the transmission delay from the data owner to the cloud server of the MBP scheme is less than the rate of increase of transmission delay of RIB-BPRE, CIBPRE, and PBRE schemes. The elements of o-text and r-key of RIB-BPRE, CIBPRE, and PBRE depend on the number of data. It needs to generate separate o-texts and r-keys in these schemes. But in the case of MBP, all the elements of o-text and r-key do not depend on the number of data it needs to share. Fig. 6 shows the comparisons of the transmission delay from the cloud server to the receiver in the MBP, RIB-BPRE, CIBPRE, and PBRE schemes. The rate of increase of the transmission delay in the MBP scheme is less than the rate of increase of the

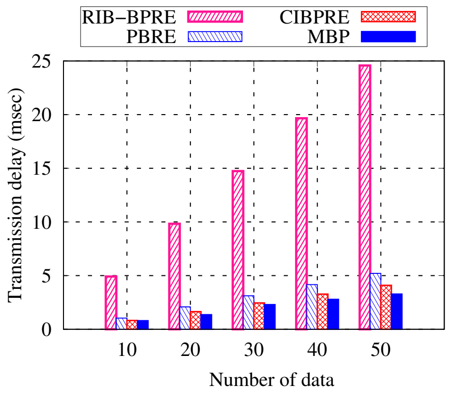

Figure 5: Transmission delay from the data owner to the cloud server

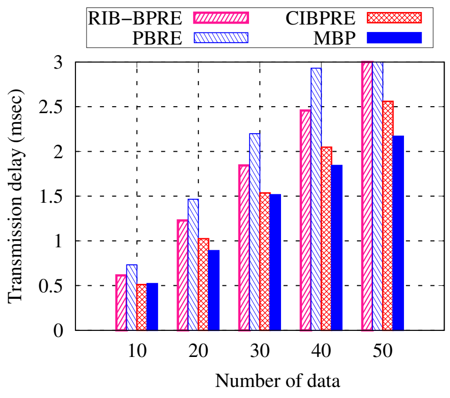

Figure 6: Transmission delay from the cloud server to the receiver

transmission delay in the RIB-BPRE, CIBPRE, and PBRE schemes. All the elements of the r-text in MBP do not depend on the number of data it needs to share, but in the case of RIB-BPRE, CIBPRE, and PBRE, all the elements of r-text depend on the number of data it needs to share.

Comparisons of energy consumption of IoT device Fig. 7 shows the comparisons of the energy consumption of the IoT device, which acts as a receiver in the MBP, RIB-BPRE, CIBPRE, and PBRE schemes. The rate of increase of the energy consumption in the MBP scheme is less than the rate of increase of the energy consumption in the RIB-BPRE, CIBPRE, and PBRE schemes. This is because the size of the r-text in MBP scheme is less than the other existing schemes.

Comparisons of computation cost Fig. 8 compares the time of the RekeyG algorithms of MBP, RIB-BPRE, CIBPRE, and PBRE schemes while varying the number of receivers. It is to be noted that a single data is considered. The RekeyG time of MBP is much less than RIB-BPRE, PBRE, and CIBPRE schemes because MBP requires less expensive operations than the other two schemes. Fig. 9 shows the comparisons of the decryption time of the r-text of the MBP scheme with the CIBPRE and RIB-BPRE schemes.

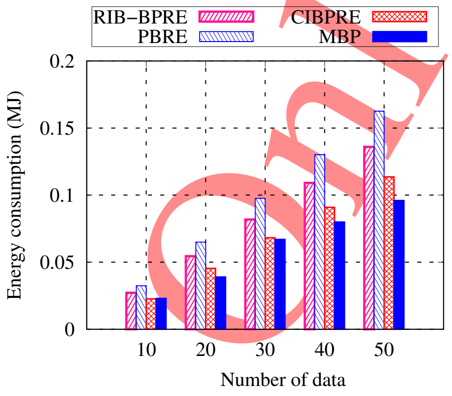

Figure 7: Energy consumption of IoT device

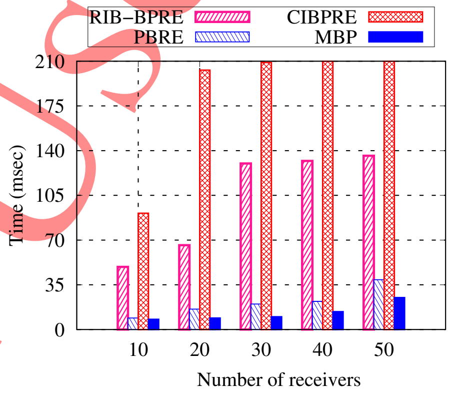

Figure 8: RekeyG time of MBP and existing schemes

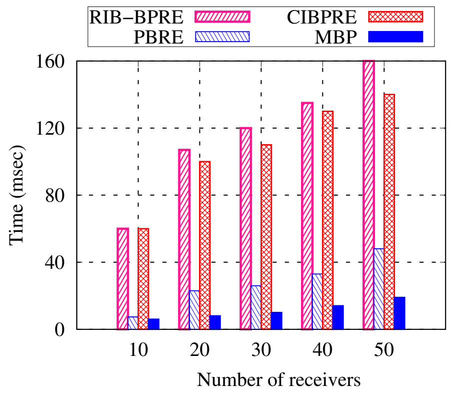

Figure 9: DecryptR time of MBP and existing schemes

It is to be noted that a single data is considered. MBP requires less number of expensive operations (e.g., bilinear pairing, modular exponentiation) than CIBPRE and RIB-BPRE schemes. Hence, the decryption time of MBP scheme is much less than the existing schemes.

Fig. 10 compares the time of DecryptR of the MBP scheme with the bargaining game and without the bargaining game. If the bargaining game is not used, the DecryptR algorithm needs to consider all the groups of all the data. If we use the bargaining game, the number of data, that

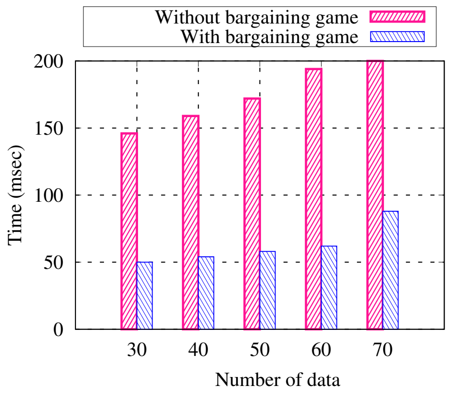

Figure 10: DecryptR time of MBP without bargaining game and with bargaining game

the receiver needs to consider, is reduced. Therefore, the required time of DecryptR is reduced if the bargaining game is used.

10. Conclusion

In this paper, we used the concept of MBE in a broadcast proxy re-encryption scheme. It allows the sender to share different ciphertexts to different groups of resource-constrained IoT devices at a time. We used the Rubinstein-Ståhl bargaining game to balance reducing the size of security components and decreasing the required time of decryption of re-encrypted ciphertext. We discussed the security of MBP and proved that MBP is selective id chosen-ciphertext secure in the random oracle model. Finally, we implemented MBP and compared its performance with the existing schemes to evaluate that the communication overhead of MBP from the data owner to the cloud server and from the cloud server to the recipient are more efficient than existing schemes.

In MBP, the size of security elements is reduced than other schemes if the number of data to be shared is more than one. In the future, the work can be extended as the size of security elements does not depend on the number of data.

References

[1] L. Ding, Z. Wang, X. Wang, D. Wu, Security information transmission algorithms for iot based on cloud computing, Computer Communications, Elsevier 155, doi: 10.1016/j.comcom.2020.03.010.
[2] C. Thirumalai, S. Mohan, G. Srivastava, An efficient public key secure scheme for cloud and iot security, Computer Communications, Elsevier 150, doi: 10.1016/j.comcom.2019.12.015.
[3] C. K. Chu, J. Weng, S. S. M. Chow, J. Zhou, R. H. Deng, Conditional Proxy Broadcast Re-Encryption, In: Boyd C., González Nieto J. (eds) Information Security and Privacy. ACISP, Lecture Notes in Computer Science, Springer 5594, doi: 10.1007/978-3-642-02620-1_23.
[4] V. C. T. Duong Hieu Phan, David Pointcheval, Multi-channel broadcast encryption, in: Proc. the 8th ACM SIGSAC symposium on Information, computer and communications security, ASIA CCS '13, no. 978-1-4503-1767-2, 2013, pp. 277–286, doi: 10.1145/2484313.2484348.
[5] S. Maiti, S. Misra, GROSE: Optimal group size estimation for broadcast proxy re-encryption, Computer Communications, Elsevier 157 (2020) 369–380, doi: 10.1016/j.comcom.2020.03.052.
[6] C. Ge, W. Susilo, Z. Liu, J. Baek, X. Luo, L. Fang, Attribute-Based Proxy Re-Encryption With Direct Revocation Mechanism for Data Sharing in Clouds, IEEE Transactions on Dependable and Secure ComputingDoi: 10.1109/TDSC.2023.3265979.
[7] M. Sun, C. Ge, L. Fang, J. Wang, A proxy broadcast re-encryption for cloud data sharing, Multimedia Tools and Applications, Springer 77 (9) (2018) 10455–10469, doi: 10.1007/s11042-017-4448-9.
[8] P. Xu, T. Jiao, Q. Wu, W. Wang, H. Jin, Conditional Identity-Based Broadcast Proxy Re-Encryption and Its Application to Cloud Email, IEEE Trans. on Computers 65 (1) (2016) 66–79, doi: 10.1109/TC.2015.2417544.
[9] L. Jiang, D. Guo, Dynamic Encrypted Data Sharing Scheme Based on Conditional Proxy Broadcast Re-Encryption for Cloud Storage, IEEE Access 5 (2017) 13336 – 13345, doi: 10.1109/ACCESS.2017.2726584.
[10] Q. Huang, Y. Yang, J. Fu, PRECISE: Identity-based private data sharing with conditional proxy re-encryption in online social networks, Future Generation Computer Systems (2018) 1523–1533.
[11] L. Yepeng, R. Yongjun, G. Chunpeng, J. Xia, Q. Wang, A CCA-secure multi-conditional proxy broadcast re-encryption scheme for cloud storage system, Journal of Information Security and Applications (2019) 125–131.
[12] J. M. M. Pérez, G. M. Pérez, A. F. S. Gomez, SecRBAC: Secure data in the Clouds, IEEE Trans. on Services Computing 10 (5) (2017) 726–740, doi: 10.1109/TSC.2016.2553668.
[13] S. Kim, I. Lee, Iot device security based on proxy re-encryption, Journal of Ambient Intelligence and Humanized ComputingDoi: 10.1007/s12652-017-0602-5.
[14] A. Manzoor, A. Braeken, S. S.Kanhere, M. Ylianttila, M. Liyanage, Proxy re-encryption enabled secure and anonymous iot data sharing platform based on blockchain, Journal of Network and Computer Applications, ElsevierDoi: 10.1016/j.jnca.2020.102917.
[15] O. A. Khashan, Hybrid lightweight proxy re-encryption scheme for secure fog-to-things environment, IEEE AccessDoi: 10.1109/ACCESS.2020.2984317.
[16] K. O.-B. O. Agyekum, Q. Xia, E. B. Sifah, C. N. A. Cobblah, H. Xia, A proxy re-encryption approach to secure data sharing in the internet of things based on blockchain, IEEE Systems JournalDoi: 0.1109/JSYST.2021.3076759.
[17] C.-I. Fan, J.-C. Chen, S.-Y. Huang, J.-J. Huang, W.-T. Chen, Provably secure timed-release proxy conditional reencryption, IEEE Systems JournalDoi: 10.1109/JSYST.2014.2385778.
[18] Y. Zhan, B. Wang, Z. Wang, T. Pei, Y. Chen, Q. Qu, Improved proxy re-encryption with delegatable verifiability, IEEE Systems Journal-Doi: 10.1109/JSYST.2019.2911556.
[19] J. Gao, H. Yu, X. Zhu, X. Li, Blockchain-based digital rights management scheme via multiauthority ciphertext-policy attribute-based encryption and proxy re-encryption, IEEE Systems JournalDoi: 10.1109/JSYST.2021.3064356.
[20] K. Liang, Q. Huang, R. Schlegel, D. S. Wong, C. Tang, A Conditional Proxy Broadcast Re-Encryption Scheme Supporting Timed-Release, in Proc. the 9th Int. Conf. Inf. Security Practice Experience (2013) 132–146Doi :10.1007/978-3-642-38033-4_10.
[21] S. Misra, A. Singh, S. Chatterjee, M. S. Obaidat, Mils-cloud: A sensor-cloud-based architecture for the integration of military tri-services operations and decision making, IEEE Systems JournalDoi: 10.1109/JSYST.2014.2316013.
[22] G. Chunpeng, Z. Liu, J. Xia, F. Liming, Revocable Identity-Based Broadcast Proxy Re-encryption for Data Sharing in Clouds, IEEE Trans. on Dependable and Secure ComputingDoi: 10.1109/TDSC.2019.2899300.
[23] S. Maiti, S. Misra, P2B: Privacy Preserving Identity-Based Broadcast Proxy Re-Encryption, IEEE Trans. on Vehicular Technology)Doi: 10.1109/TVT.2020.2982422.
[24] J. Zhang, S. Su, H. Zhong, J. Cui, D. He, Identity-Based Broadcast Proxy Re-Encryption for Flexible Data Sharing in VANETs, IEEE Transactions on Information Forensics and Security 18, doi: 10.1109/TIFS.2023.3299466.

[25] K. Acharya, R. Dutta, Constructions of Secure Multi-Channel Broadcast Encryption Schemes in Public Key Framework, in: International Conference on Cryptology and Network Security, Cryptology and Network Security, Vol. 11124, 2018, pp. 495–515, doi: 10.1007/978-3-030-00434-7_25.

[26] J. Kim, W. Susilo, M. H. Au, J. Seberry, Efficient Semi-static Secure Broadcast Encryption Scheme, in: Proc. Int. Conf. on Pairing-Based Cryptography, Vol. 8365, 2013, pp. 62–76, doi: 10.1007/978-3-319-04873-4_4.

[27] S.Canard, D. Phan, D. Pointcheval, V. Trinh, A new technique for compacting ciphertext in multi-channel broadcast encryption and attribute-based encryption, Theoretical Computer Science, Elsevier 723 (2018) 51–72, doi: 10.1016/j.tcs.2018.02.036.

[28] A. Rubinstein, Perfect Equilibrium in a Bargaining Model , Econometrica, 1982, 97–109.

[29] I. Ståhl, Bargaining Theory, Stockholm, Sweden: Stockholm School of Economics.

[30] S. Brahma, M. Chatterjee, Spectrum Bargaining: A Model for Competitive Sharing of Unlicensed Radio Spectrum, IEEE Trans. on Cognitive Communications and Networking, 2015, 257–272.

[31] E. E. Tsiropoulou, A. Kapoukakis, S. Papavassiliou, Energy-efficient subcarrier allocation in SC-FDMA wireless networks based on multilateral model of bargaining, in Proc. IFIP Networking Conference, 2013.

[32] C. Delerablée, Identity-Based Broadcast Encryption with Constant Size Ciphertexts and Private Keys, in Proc. the 13th Int. Conf. Theory Appl. Cryptol. Inf. Security: Adv. Cryptol. 4833 (2007) 200–215, doi: 10.1007/978-3-540-76900-2_12.

[33] S. Brahma, M. Chatterjee, Spectrum sharing in secondary networks: A bargain theoretic approach, in: Proc. the Wireless Communications and Networking Conf. (WCNC), IEEE, 2012, doi: 10.1109/WCNC.2012.6213986.

[34] B. Lynn, Pbc library (2013).
URL https://crypto.stanford.edu/pbc/