Hierarchical Conditional Proxy Re-Encryption: A New Insight of Fine-Grained Secure Data Sharing

Kai He[1], Xueqiao Liu[2], Huaqiang Yuan[1(✉)], Wenhong Wei[1], and Kaitai Liang[3]

[1] School of Computer and Network Security, Dongguan University of Technology, Guangdong 523808, China
kaihe1214@163.com, hyuan66@163.com, weiwh@dgut.edu.cn
[2] School of Computing and Information Technology, University of Wollongong, Wollongong, NSW 2512, Australia
xl691@uowmail.edu.au
[3] Department of Computer Science, University of Surrey, Guildford, UK
ktliang88@gmail.com

Abstract. Outsource local data to remote cloud has become prevalence for Internet users to date. While being unable to "handle" (outsourced) data at hand, Internet users may concern about the confidentiality of data but also further operations over remote data. This paper deals with the case where a secure data sharing mechanism is needed when data is encrypted and stored in remote cloud. Proxy re-encryption (PRE) is a promising cryptographic tool for secure data sharing. It allows a "honest-but-curious" third party (e.g., cloud server), which we call "proxy", to convert all ciphertexts encrypted for a delegator into those intended for a delegatee. The delegatee can further gain access to the plaintexts with private key, while the proxy learns nothing about the underlying plaintexts. Being regarded as a general extension of PRE, conditional PRE supports a fine-grained level of data sharing. In particular, condition is embedded into ciphertext that offers a chance for the delegator to generate conditional re-encryption key to control with which ciphertexts he wants to share. In this paper, for the first time, we introduce a new notion, called "hierarchical conditional" PRE. The new notion allows re-encryption rights to be "re-delegated" for "low-level" encrypted data. We propose the seminal scheme satisfying the notion in the context of identity-based encryption and further, prove it secure against chosen-ciphertext security.

Keywords: Hierarchical conditional proxy re-encryption
Fine-grained data sharing · Identity-based encryption
Chosen-ciphertext security

1 Introduction

To date cloud computing has been regarded as a successful and prevalent business model for many real-world applications due to its long-list features, such

J. K. Liu and P. Samarati (Eds.): ISPEC 2017, LNCS 10701, pp. 118–135, 2017.
https://doi.org/10.1007/978-3-319-72359-4_7

as considerable storage and computing power. Internet users have been "encouraged" to outsource their data to cloud in order to save the cost of local data maintenance and management but also to enjoy various cloud-based data services. To prevent their sensitive data from being compromised by cloud server, Internet users may choose to encrypt the data before outsourcing. However, the encryption may limit "out-of-physical" sharing. For example, a user $\mathcal{A}$ may share his data with another user, say $\mathcal{B}$. Assume the data of $\mathcal{A}$ is stored in a cloud server. A naive way for the sharing is to let $\mathcal{A}$ first download his encrypted data locally and decrypt it, then re-encrypt the data for $\mathcal{B}$. The solution, however, may require $\mathcal{A}$ to be on-line and meanwhile, bear all the workloads of decryption-and-re-encryption. To offload the workloads to the server, one may choose to allow the server to execute the decrypt-then-re-encrypt task. But this will compromise the confidentiality of the data.

Proxy re-encryption (PRE), which is a useful cryptographic primitive, has been introduced to tackle the above dilemma. By using PRE, $\mathcal{A}$ does not need to download, decrypt and re-encrypt the data. Instead, he is only required to generate a re-encrypted key, which supports ciphertext conversion, so that a semi-trust (i.e. honest-but-curious) cloud server (i.e. proxy) can use the re-encryption key to transform the ciphertext of $\mathcal{A}$ for $\mathcal{B}$. Even if the proxy obtains the re-encryption key, it cannot gain access to the underlying data. Since its introduction, PRE has been widely applied in many real-world applications, such as digital rights management systems [35], secure distributed files systems [1,9] and email forwarding systems [2].

In a traditional PRE mechanism, using a re-encryption key from $\mathcal{A}$ to $\mathcal{B}$, the proxy may transform all ciphertexts of $\mathcal{A}$ into those intended for $\mathcal{B}$. This "all-or-nothing" data sharing mode may not scale well in practice. What if some data is extremely sensitive to $\mathcal{A}$ so that he does not want it to be shared with others, even including $\mathcal{B}$? A fine-grained PRE may be desirable in this case. In 2009, Weng et al. [41] introduced the notion of conditional PRE (CPRE), in which the proxy who has a re-encryption key with a special condition can only convert the ciphertext of a delegator (e.g., $\mathcal{A}$) with the same special condition for a delegatee (e.g., $\mathcal{B}$). Due to its innate feature, CPRE, however, limits the data sharing in the sense that one re-encryption key only corresponds to the sharing of one ciphertext. This one-to-one sharing mode brings inconvenience for delegator. Specifically, if $\mathcal{A}$ plans to share 10,000 encrypted files (which are embedded with distinct conditions) with $\mathcal{B}$, he has to generate the same amount of re-encryption keys.

To address the above limitation, we introduce a new notion, which we call "hierarchical conditional" PRE (HCPRE). The new notion allows re-encryption rights to be "re-delegated" to lower level of encrypted data. It brings convenience and flexibility for delegator in the sense that a delegator may only need to generate a re-encryption key for high level data and further, the key can be "reformed" for the lower level data shoring. Below we use cloud data sharing as an example to illustrate the basic idea behind the notion to motivate our work.

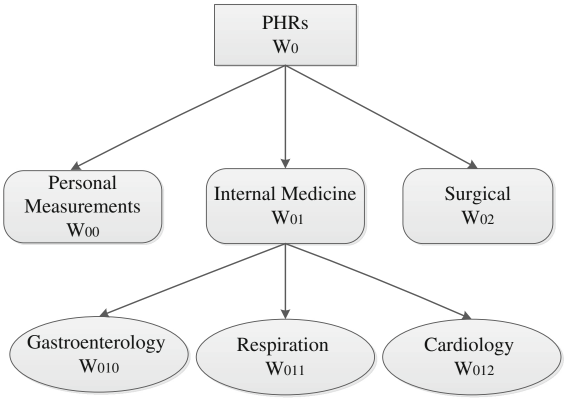

Fig. 1. Hierarchical conditional access structure

Assume outsourced data is under a specific data structure for some purposes, e.g., efficient retrieval. $\mathcal{A}$ first forms his data in a hierarchical structure as shown in Fig. 1, in which a data is tagged with a hierarchical condition set, for example, the data (related to) *Respiration* is with a condition set $W = \{W_0, W_{01}, W_{011}\}$. $\mathcal{A}$ further encrypts the data together with the corresponding hierarchical condition set before outsourcing to a cloud serer. Assume $\mathcal{B}$ is a Physician, who is allowed to access all of the *Internal Medicine* data of $\mathcal{A}$. To share the data with $\mathcal{B}$, $\mathcal{A}$ may generate a re-encryption key $RK_{\{W_0,W_{01}\}|A\to B}$, which is embedded with hierarchical conditions $\{W_0, W_{01}\}$, and sends it to the semi-trust cloud server. When $\mathcal{B}$ requests to access the *Internal Medicine* data, including *Gastroenterology*, *Respiration* and *Cardiology*, the proxy uses the re-encryption key $RK_{\{W_0,W_{01}\}|A\to B}$, which is for the conditions $\{W_0, W_{01}\}$, to "delegate" a new re-encryption key $RK_{\{W_0,W_{01},W_{01i}\}_{i\in\{0,1,2\}}|A\to B}$ for the "lower-level" hierarchical conditions $\{W_0, W_{01}, W_{01i}\}_{i\in\{0,1,2\}}$. The proxy further uses the resulting key $RK_{\{W_0,W_{01},W_{01i}\}_{i\in\{0,1,2\}}|A\to B}$ to convert the encrypted data for $\mathcal{B}$, so that $\mathcal{B}$ may use his private key to access the *Internal Medicine* data of $\mathcal{A}$. In particular, if $\mathcal{A}$ decides to share all of his data to $\mathcal{B}$, he only needs to generate a "root" re-encryption key for condition W_0 from $\mathcal{A}$ to $\mathcal{B}$; while $\mathcal{A}$ chooses to share one leaf data to $\mathcal{B}$, he generates a re-encryption key for one of the conditions $\{W_0, W_{01}, W_{01i}\}$ corresponding to the leaf of the structure.

1.1 Related Work

In 1998, Blaze et al. [2] constructed the first bidirectional PRE scheme. In 2005, Ateniese et al. [9] proposed the first unidirectional PRE scheme. Both of the schemes are secure only against chosen-plaintext attacks (CPA). In 2007, Canetti et al. [3] designed a bidirectional PRE scheme with chosen-ciphertext security. In 2008, Libert et al. [24] introduced a re-playable chosen ciphertext secure (RCCA) unidirectional PRE scheme. Since then, various PRE schemes have been proposed in the literature (e.g., [7,11,25,29,34,37,40]).

PRE can be extended in the context of identity-based encryption. In 2007, Green and Ateniese [10] proposed the first identity-based proxy re-encryption (IBPRE) scheme, which is CCA secure in the random oracle model, where hash functions are assumed to be fully random. Chu and Tzeng [6] constructed a CCA secure IBPRE scheme in the standard model. After that, many identity-based proxy re-encryption (IBPRE) schemes have been proposed, such as [6,10,18,20,28,30,31,33,38].

However, among all of the aforementioned schemes, the semi-trust proxy can use a given re-encryption key to transform all the ciphertexts of a delegator into those of a delegatee. But in reality, the delegator does not want to transform all of his data for the delegatee. Therefore, type-based PRE [36] and conditional PRE (CPRE) [41,42] were proposed, in which the proxy can only fulfill ciphertext conversion "conditionally". Later, Liang et al. [16,19] proposed two IBCPRE schemes with CCA secure in the standard model. However, He et al. [12] presented the security analysis to show that their schemes only achieve CPA security. In 2016, He et al. [13] proposed an efficient identity-based conditional proxy re-encryption (IBCPRE) scheme with CCA secure in the random oracle model.

PRE can be extended in the attribute-based setting. Attribute-based proxy re-encryption (ABPRE) can effectively increase the flexibility of data sharing. In 2009, Liang et al. [23] first defined the notion of ciphertext-policy ABPRE (CP-ABPRE), where each ciphertext is labeled with a set of descriptive conditions and each re-encryption key is associated with an access tree that specifies which type of ciphertexts the proxy can re-encrypt, and they presented a concrete scheme supporting AND gates with positive and negative attributes. After that, several CP-ABPRE schemes (e.g., [27]) with more expressive access policy were proposed. In 2011, Fang et al. [8] proposed a key-policy ABPRE (KP-ABPRE) scheme in the random oracle model, whereby ciphertext encrypted with conditions W can be re-encrypted by the proxy using the CPRE key under the access structure $\mathcal{T}$ if and only if $\mathcal{T}(W) = 1$. More recent ABPRE systems can be seen in [15,17,21,22].

In 2016, Lee et al. [14] proposed a searchable hierarchical CPRE (HCPRE) scheme for cloud storage services, and cloud service provider is able to generate a hierarchical key, but the re-encryption key generation algorithm also requires the private keys of the delegator and delegatee.

So far, the proxy re-encryption scheme [13] is the only one which is conditional and chosen-ciphertext secure scheme in the identity-based setting. Therefore, based on the scheme [13], we propose a HCPRE scheme with more scalability and flexibility in controlling data sharing and which is in identity-based setting and further achieves CCA security. Note that secure access control have also been proposed in the literature for fine-grained data sharing (e.g., [4,5,32]).

We here compare our scheme with other related PRE schemes, namely CPRE, IB-PRE and AB-PRE, in terms of computation, communication, features as well as security in the following tables. We state that AB-PRE allows proxy to convert a group of ciphertext satisfying attribute description embedded into re-encryption key. This is somewhat similar to our scheme. But the distinct feature of our scheme is that we can support re-encryption key re-delegation in a secure and scalable way. Let C_e, C_p, C_S and C_E be the computational cost of an exponentiation, a bilinear pairing, a signature and a symmetric encryption, respectively. u is the total number of attributes used in system, w is the number of conditions in the ciphertext and d is the size of an access formula. $|G_1|$ and $|G_T|$ denote the bit-length of an element in $\mathbb{G}_1$ and $\mathbb{G}_T$, respectively. $|Sym|$ and $|Sign|$ denote the bit-length of a symmetric encryption and a signature, respectively.

From Table 1, it can be seen that our scheme achieves constant pairing cost in all metrics, much like others, except for the re-encryption phase. We state that this will not bring heavy computational burden to system user because this phase is handled by cloud server. Since our scheme supports flexible condition control, the number of condition used in ciphertext and sharing/re-encryption is based on the preference of user. If a user chooses to use only one condition (i.e. $w = 1$), our scheme also achieves constant computational cost in all metrics.

Table 2 shows the communication cost comparison. Much like the analysis mentioned previously, our scheme would achieve constant communication cost while $w = 1$. We note that $w = 1$ may indicate that a delegator delegates the decryption rights of a "root" data to a delegatee.

Table 1. Computation cost comparison

Schemes	Enc	Re-Enc	Dec_1	Dec_2	Rekey
[16]	$8C_e + C_p + C_S$	$6C_e + 7C_p$	$5C_e + 6C_p$	$5C_e + 6C_p$	$16C_e$
[41]	$4C_e + 2C_p$	$8C_p$	$2C_e + 2C_p$	$C_e + C_p$	$2C_e$
[10]	$4C_e + C_p + C_S$	$2C_e$	$2C_e + 3C_p$	$2C_e + 10C_p$	$4C_e + C_p + C_S$
[26]	$3C_e + C_p$	$4C_p$	$2C_p$	$2C_p$	$4C_e$
[23]	$(2 + u)C_p$	$(1 + u)C_p$	$(1 + u)C_p$	$2C_p$	$(2u + 1)C_e$
Ours	$(2 + w)C_e + C_p$	$(3 + w)C_p$	$C_e + 2C_p$	$2C_e + 2C_p$	$(2w + 1)C_e + C_p$

Table 2. Communication complexity comparison

Schemes	RKey	Original ciphertext	Re-encryption ciphertext
[16]	$6\|G_1\|$	$3\|G_1\| + \|G_T\| + \|Sign\|$	$3\|G_1\| + \|G_T\| + \|Sign\|$
[41]	$2\|G_1\|$	$4\|G_1\|$	$2\|G_1\| + \|G_T\|$
[10]	$3\|G_1\| + \|G_T\| + \|Sign\|$	$9\|G_1\| + 2\|G_T\| + 2\|Sign\|$	$5\|G_1\| + \|G_T\| + \|Sign\|$
[26]	$2\|G_1\|$	$3\|G_1\| + \|G_T\|$	$2\|G_1\| + \|G_T\|$
[23]	$(3 + 3u)\|G_1\| + \|G_T\|$	$(2 + u)\|G_1\| + \|G_T\|$	$(3 + u)\|G_1\| + (4 + u)\|G_T\|$
Ours	$(3 + w)\|G_1\| + \|G_T\|$	$(3 + w)\|G_1\| + \|G_T\|$	$3\|G_1\| + 2\|G_T\|$

Table 3. Feature and security comparison

Schemes	Conditional sharing RKey number	Complexity	Security	Adaptivity	RKey re-delegation
[16]	$O(d)$	ℓ-wBDHI*	CCA	×	×
[41]	$O(d)$	3-QBDH	CCA	√	×
[10]	$O(d)$	DBDH	CPA	√	×
[26]	$O(d)$	DBDH	CPA	×	×
[23]	O(1)	ADBDH	CPA	×	×
Ours	O(1)	DBDH	CCA	√	√

The comparison of feature and security is shown in Table 3. We can see that our scheme is the first and only achieving all features. Like [23], our scheme only needs constant number of re-encryption key while the others need the number of $O(d)$. It also achieves adaptively CCA security under well-study complexity assumption, DBDH. A re-encryption key in our scheme can be further re-delegated by proxy (for re-encryption key recycle purpose) without jeopardizing security.

1.2 Contributions

The contributions of this paper are described as follows.

- Taking into account structured data, we introduce the new notion, hierarchical conditional PRE. The new notion allows a proxy to "re-formed" a given re-encryption key, so that the resulting key can be used to re-encrypt "lower-level" encrypted data. In other words, a re-encryption key in our notion may be "recycled".
- We concretely explore the notion in the context of identity-based encryption, and further define the corresponding system and security notion. We present a concrete construction satisfying the notion, which is the first of its type. Specifically, the construction is inspired by [13].
- The premise of our construction is quite similar to the hierarchical identity-based secret key re-delegation technique. A semi-trust proxy is allowed to delegate an "upper level" re-encryption key generation to lower-level "conditions". Therefore, a delegator can control which specific data blocks located in the structure can be accessed by others without generating a huge amount of re-encryption key.
- Our scheme is proved secure against chosen-ciphertext attacks in the random oracle model.

1.3 Organization

The rest of this paper is organized as follows. Some necessary preliminaries, system definition and security notion are given in Sect. 2. The concrete construction is introduced in Sect. 3 and the security analysis are described in Sect. 4. The conclusion is presented in Sect. 5.

2 Preliminaries

2.1 Bilinear Map

Two multiplicative cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ whose orders are prime p and a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ has following three properties:

- Bilinearity: $e(u^a, v^b) = e(u, v)^{ab}$ given $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$.
- Non-degeneracy: $e(g, g) \rightarrow \mathbb{G}$ given a generator g of $\mathbb{G}$.
- Computability: There exists a probabilistic algorithm to compute $e(u, v)$ given $u, v \in \mathbb{G}$.

2.2 Decisional Bilinear Diffie-Hellman (DBDH) Assumption

The definition of DBDH assumption [39] in a bilinear group $(p, \mathbb{G}, \mathbb{G}_T, e)$ is given as follows: A challenger takes as input (g, g^a, g^b, g^c, Z) for the unknown $a, b, c \leftarrow_R \mathbb{Z}_p$. A probabilistic polynomial time (PPT) adversary needs to decide whether $Z = e(g, g)^{abc}$ or Z is a random chosen from $\mathbb{G}_T$. The advantage of the PPT adversary $\mathcal{A}$ solving the DBDH assumption is defined like this:

$$Adv_{\mathcal{A}}^{\mathrm{DBDH}} = |\Pr[\mathcal{A}(g, g^a, g^b, g^c, e(g, g)^{abc}) = 1] - \Pr[\mathcal{A}(g, g^a, g^b, g^c, Z) = 1]|.$$

If the advantage is negligible, it means that the DBDH assumption holds.

2.3 Identity-Based Hierarchical Conditional Proxy Re-encryption (IBHCPRE)

We here define the algorithms and security notion for IBHCPRE. An IBHCPRE scheme includes the following algorithms:

- Setup(1^λ): Intake a security parameter 1^λ, output a public parameter $params$ and a master secret key msk.
- Extract(msk, ID): Intake the master secret key msk and an identity ID, output a private key sk_{ID}.
- Enc($params, ID_i, W_n, m$): Intake the public parameter $params$, an identity ID_i, a condition vector $W_n = \{w_1, w_2, \cdots, w_n\}$ of depth n and a plaintext $m \in \mathcal{M}$, output an initial ciphertext $CT_{(ID_i, W_n)}$.
- ReKeyGen(sk_{ID_i}, ID_j, W_n): Intake a private key sk_{ID_i}, an identity ID_j, and a condition vector $W_n = \{w_1, w_2, \cdots, w_n\}$ of depth n, output a re-encryption key $rk_{W_n|ID_i \rightarrow ID_j}$ from ID_i to ID_j associated with the condition vector W_n.
- HCReKeyGen($rk_{W_n|ID_i \rightarrow ID_j}, W_{n+1}$): Intake a re-encryption key $rk_{W_n|ID_i \rightarrow ID_j}$ for the parent condition $W_n = \{w_1, \cdots, w_n\}$ of depth n and a condition vector $W_{n+1} = \{W_n, w_{n+1}\}$ of depth $n+1$, output the re-encryption key $rk_{W_{n+1}|ID_i \rightarrow ID_j}$ from ID_i to ID_j for condition $W_{n+1} = \{w_1, \cdots, w_n, w_{n+1}\}$.

- ReEnc($rk_{W_n|ID_i \to ID_j}, CT_{(ID_i,W_n)}$): Intake a re-encryption key $rk_{W_n|ID_i \to ID_j}$ and an initial ciphertext $CT_{(ID_i,W_n)}$, output a transformed ciphertext $CT_{(ID_j,W_n)}$.
- Dec$_2$(sk_{ID_i}, $CT_{(ID_i,W_n)}$): Intake a private key sk_{ID_i} and an initial ciphertext $CT_{(ID_i,W_n)}$, output a plaintext m or an invalid symbol $\perp$.
- Dec$_1$(sk_{ID_j}, $CT_{(ID_j,W_n)}$): Intake a private key sk_{ID_j} and a transformed ciphertext $CT_{(ID_j,W_n)}$, output a plaintext m or an invalid symbol $\perp$.

Correctness: For any $m \in \mathcal{M}$, sk_{ID_i} and sk_{ID_j} are generated from $\mathsf{Extract}$ algorithm, it holds that $\mathsf{Dec}_2(sk_{ID_i}, CT_{(ID_i,W_n)}) = M$ and $\mathsf{Dec}_1(sk_{ID_j}$, $\mathsf{ReEnc}(\mathsf{ReKeyGen}(sk_{ID_i}, ID_j, W_n), CT_{(ID_i,W_n)}) = M$.

Next, we give the security definition for IBHCPRE in the sense of indistinguishability under chosen-ciphertext attacks (IND-CCA), which is described by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. Adversary $\mathcal{A}$ is able to obtain a series of queries. In spite of this, an adversary $\mathcal{A}$ cannot distinguish which message is encrypted from the challenge ciphertext.

- Setup: Challenger $\mathcal{C}$ runs $(params, msk) \leftarrow \mathsf{Setup}(1^\lambda)$, it sends $params$ to $\mathcal{A}$ and keeps msk itself.
- Phase 1: Adversary $\mathcal{A}$ adaptively issues a polynomial number of queries:
  - *Extraction query* $\langle ID_i \rangle$: Challenger $\mathcal{C}$ runs $\mathsf{Extract}(msk, ID_i)$ to obtain a private key sk_{ID_i} and returns it to adversary $\mathcal{A}$.
  - *Re-encryption key query* $\langle ID_i, ID_j, W_n \rangle$: Challenger $\mathcal{C}$ first gets the private key $sk_{ID_i} \leftarrow \mathsf{Extract}\ (msk, ID_i)$ and runs $rk_{W_n|ID_i \to ID_j} \leftarrow \mathsf{ReKeyGen}(sk_{ID_i}, ID_j, W_n)$, and then it returns $rk_{W_n|ID_i \to ID_j}$ to adversary $\mathcal{A}$.
  - *Hierarchical condition re-encryption key query* $\langle rk_{W_n|ID_i \to ID_j}, W_{n+1} \rangle$: Challenger $\mathcal{C}$ gets the re-encryption key for parent condition vector W_n of depth n and runs $rk_{W_{n+1}|ID_i \to ID_j} \leftarrow \mathsf{HCReKeyGen}(rk_{W_n|ID_i \to ID_j}, W_{n+1})$.
  - *Re-encryption query* $\langle ID_i, ID_j, CT_{(ID_i,W_n)} \rangle$: Challenger $\mathcal{C}$ first gets the re-encryption key $rk_{W_n|ID_i \to ID_j} \leftarrow \mathsf{ReKeyGen}(sk_{ID_i}, ID_j, W_n)$ and runs $CT_{(ID_j,W_n)} \leftarrow \mathsf{ReEnc}(rk_{W_n|ID_i \to ID_j}, CT_{(ID_i,W_n)})$, and then it returns $CT_{(ID_j,W_n)}$ to adversary $\mathcal{A}$.
  - *Decryption query* $\langle ID, CT_{(ID,W_n)} \rangle$: Challenger $\mathcal{C}$ first gets the private key $sk_{ID} \leftarrow \mathsf{Extract}(msk, ID)$ and runs the decryption algorithm and returns the result $\mathsf{Dec}_1(sk_{ID}, CT_{(ID,W_n)})$ or $\mathsf{Dec}_2(sk_{ID}, CT_{(ID,W_n)})$ to adversary $\mathcal{A}$.
- Challenge: Adversary $\mathcal{A}$ outputs a target identity ID^* and condition W_n^* as well as two distinct plaintexts $m_0, m_1 \in \mathcal{M}$. Challenger $\mathcal{C}$ picks $\beta \in_R \{0,1\}$ and returns $CT^*_{(ID^*,W_n^*)} = \mathsf{Enc}\ (params, ID^*, W_n^*, m_\beta)$ to adversary $\mathcal{A}$.
- Phase 2: Adversary $\mathcal{A}$ keeps on issuing all queries as in Phase 1, challenger $\mathcal{C}$ responds the queries as in Phase 1. But the difference is that Phase 2 needs to satisfy the following conditions:
  - Adversary $\mathcal{A}$ cannot issue *Extraction query* on ID^*.
  - Adversary $\mathcal{A}$ cannot issue *Decryption query* on neither $\langle ID^*, CT^*_{(ID^*,W_n^*)} \rangle$ nor $\langle ID_j$, ReEnc$(rk_{W_n^*|ID^* \to ID_j}, CT^*_{(ID^*,W_n^*)}) \rangle$.

  - If adversary $\mathcal{A}$ gets sk_{ID_j} on ID_j, it cannot issue *Re-encryption query* on $\langle ID^*, ID_j, CT^*_{(ID^*,W_n^*)}\rangle$ and *Re-encryption key query* on $\langle ID^*, ID_j, W_k^*\rangle$, where $W_k^* = \{w_1, \cdots, w_k\}$ and $k \in [1, n]$.
- Guess: Adversary $\mathcal{A}$ makes a guess $\beta' \in \{0, 1\}$ and wins the game if $\beta' = \beta$.

We define adversary $\mathcal{A}$'s advantage in the above game as

$$Adv_{\mathcal{A}}^{\text{IND-IBHCPRE-CCA}} = |\Pr[\beta' = \beta] - 1/2|.$$

Definition 1 *(IND-IBHCPRE-CCA Security). We say that an IBHCPRE scheme is IND-CCA secure, if for any PPT adversary $\mathcal{A}$, the advantage in the above security game is negligible, that is $Adv_{\mathcal{A}}^{IND\text{-}IBHCPRE\text{-}CCA} \leq \epsilon$.*

3 Construction

- Setup(1^λ): Given a security parameter 1^λ, first output a bilinear group $(p, \mathbb{G}, \mathbb{G}_T, e)$, and then choose a generator $g \in_R \mathbb{G}$, $\alpha \in_R \mathbb{Z}_p$ and compute $g_1 = g^\alpha$. Finally, choose six hash functions H_1, H_2, H_3, H_4, H_5 and H_6, where $H_1 : \{0,1\}^* \to \mathbb{G}$, $H_2 : G_T \times \mathcal{M} \to \mathbb{Z}_p$, $H_3 : \mathbb{G}_T \to \mathcal{M}$, $H_4 : \{0,1\}^* \times \mathbb{G} \times \mathbb{G}_T \times \mathcal{M} \times \mathbb{G}^n \to \mathbb{G}$, $H_5 : \{0,1\}^* \to \mathbb{G}$ and $H_6 : \mathcal{M} \to \mathbb{G}$, where $\mathcal{M}$ is the massage space. The public parameter is
$$PPs = ((p, \mathbb{G}, \mathbb{G}_T, e), g, g_1, H_1, H_2, H_3, H_4, H_5, H_6)$$
and the master secret key is $msk = \alpha$.
- Extract(msk, ID): Given the master secret key msk and an identity ID, it computes $Q_{ID} = H_1(ID)$ and sets the private key as
$$sk_{ID} = Q_{ID}^\alpha.$$
- Enc($PPs, ID_i, W_n = \{w_1, \cdots, w_n\}, M$): Given the public parameter PPs, an identity ID_i, a condition vector $W_n = \{w_1, \cdots, w_n\}$ and a message $M \in \mathcal{M}$, pick $\delta \in_R \mathbb{G}_T$ and set $r = H_2(\delta||M)$,
$$\begin{aligned} A &= g^r \\ B &= \delta \cdot e(g_1, H_1(ID_i))^r \\ C &= H_3(\delta) \oplus M \\ D_1 &= H_5(ID_i||w_1)^r \\ D_2 &= H_5(ID_i||w_1||w_2)^r \\ &\cdots \\ D_n &= H_5(ID_i||w_1||\cdots||w_n)^r \\ S &= H_4(ID_i||A||B||C||D_1||\cdots||D_n)^r \end{aligned}$$
Then output an initial ciphertext
$$CT_{(ID_i,W_n)} = (A, B, C, D_1, \cdots, D_n, S, W_n).$$

– ReKeyGen($sk_{ID_i}, ID_j, W'_n = \{w'_1, \cdots, w'_n\}$): Given the private key sk_{ID_i}, an identity ID_j and a condition vector W'_n, first pick $\theta \in_R \mathcal{M}$, $\delta' \in_R \mathbb{G}_T$ and set $r' = H_2(\delta'||\theta)$ and pick $s_1, \cdots, s_n \in_R \mathbb{Z}_p^n$

$$\begin{aligned}
rk_1 &= g^{r'} \\
rk_2 &= \delta' \cdot e(g_1, H_1(ID_j))^{r'} \\
rk_3 &= H_3(\delta') \oplus \theta \\
RK_1 &= sk_{ID_i} \cdot H_5(ID_i||w'_1)^{s_1} \cdots H_5(ID_i||w'_1||\cdots||w'_n)^{s_n} \cdot H_6(\theta) \\
RK_2^1 &= g^{s_1} \\
&\cdots \\
RK_2^n &= g^{s_n}
\end{aligned}$$

Finally, output the re-encryption key

$$rk_{W'_n|ID_i \to ID_j} = (rk_1, rk_2, rk_3, RK_1, RK_2^1, \cdots, RK_2^n).$$

– HCReKeyGen($rk_{W'_n|ID_i \to ID_j}, W'_{n+1}$): Give the re-encryption key $rk_{W'_n|ID_i \to ID_j}$ for a parent condition vector $W'_n = \{w'_1, \cdots, w'_n\}$, compute a hierarchical conditional re-encryption key $rk_{W'_{n+1}|ID_i \to ID_j}$ for a condition vector $W'_{n+1} = \{w'_1, \cdots, w'_n, w'_{n+1}\}$ as follows:
Choose $r'', s'_1, s'_2, \cdots, s'_n, s_{n+1} \in_R \mathbb{Z}_p$ and compute

$$\begin{aligned}
rk'_1 &= rk_1 \cdot g^{r''} \\
rk'_2 &= rk_2 \cdot e(g_1, H_1(ID_j))^{r''} \\
rk'_3 &= rk_3 \\
RK'_1 &= RK_1 \cdot H_5(ID_i||w'_1||\cdots||w'_{n+1})^{s_{n+1}} \cdot H_5(ID_i||w'_1)^{s'_1} \cdots \\
&\quad H_5(ID_i||w'_1||\cdots||w'_n)^{s'_n} \\
RK_2^{'1} &= RK_2^1 \cdot g^{s'_1} \\
&\cdots \\
RK_2^{'n} &= RK_2^n \cdot g^{s'_n} \\
RK_2^{'n+1} &= g^{s_{n+1}}
\end{aligned}$$

Finally, output the re-encryption key

$$rk_{W_{n+1}|ID_i \to ID_j} = (rk'_1, rk'_2, rk'_3, RK'_1, RK_2^{'1}, \cdots, RK_2^{'n+1})$$

which is a valid re-encryption key, as the distribution of the re-encryption key is the same as the distribution of keys generated by ReKeyGen.

– ReEnc($rk_{W_n|ID_i \to ID_j}, CT_{(ID_i, W_n)}$): Given a re-encryption key $rk_{W_n|ID_i \to ID_j}$ and an initial ciphertext $CT_{(ID_i, W_n)}$, check whether

$$\begin{aligned}
&e(SD_1 \cdots D_n, g) = \\
&e(H_4(ID_i||A||B||C||D_1||\cdots||D_n)H_5(ID_i||w_1) \cdots H_5(ID_i||w_1||\cdots||w_n), A).
\end{aligned}$$

If not, output $\perp$; otherwise compute

$$B' = B \cdot \frac{e(D_1, RK_2^1) \cdots e(D_n, RK_2^n)}{e(A, RK_1)} = \delta / e(A, H_6(\theta)).$$

Then output the transformed ciphertext

$$CT_{(ID_j, W_n)} = (A, B', C, rk_1, rk_2, rk_3).$$

– $\mathrm{Dec}_2(sk_{ID_i}, CT_{(ID_i, W_n)})$: Given the private key sk_{ID_i} and the initial ciphertext $CT_{(ID_i, W_n)}$, first check whether

$$e(SD_1 \cdots D_n, g) =$$
$$e(H_4(ID_i||A||B||C||D_1|| \cdots ||D_n) H_5(ID_i||w_1) \cdots H_5(ID_i||w_1|| \cdots ||w_n), A).$$

If not, output $\perp$; otherwise, compute

$$\begin{aligned} \delta &= B / e(A, sk_{ID_i}) \\ M &= H_3(\delta) \oplus C. \end{aligned}$$

Then check whether

$$A = g^{H_2(\delta||M)}.$$

If not, output $\perp$; otherwise output M.

– $\mathrm{Dec}_1(sk_{ID_j}, CT_{(ID_j, W_n)})$: Given the private key sk_{ID_j} and the transformed ciphertext $CT_{(ID_j, W_n)}$, first compute

$$\begin{aligned} \delta' &= rk_2 / e(rk_1, sk_{ID_j}) \\ \theta &= H_3(\delta') \oplus rk_3. \end{aligned}$$

Then it checks whether

$$rk_1 = g^{H_2(\delta'||\theta)}.$$

If not, output $\perp$; else compute

$$\begin{aligned} \delta &= B' \cdot e(A, H_6(\theta)) \\ M &= H_3(\delta) \oplus C. \end{aligned}$$

Finally, check whether

$$A = g^{H_2(\delta||M)}.$$

If not, output $\perp$; otherwise output M.

4 Security Analysis

In the following, we prove that our construction is IND-IBHCPRE-CCA secure in the random oracle model.

Theorem 1. *Suppose that the DBDH assumption holds in a bilinear group (p, G, G_T, e), then the above IBHCPRE scheme is IND-CCA secure in the random oracle model.*

Concretely, if adversary $\mathcal{A}$ with a non-negligible advantage against the above IBHCPRE scheme, then there exists a challenger $\mathcal{C}$ to solve the DBDH assumption with a non-negligible advantage.

Proof. Suppose that adversary $\mathcal{A}$ has a non-negligible advantage to attack the above IBHCPRE scheme. We can build a PPT challenger $\mathcal{C}$ that makes use of adversary $\mathcal{A}$ to solve the DBDH problem. Challenger $\mathcal{C}$ is given a DBDH instance (g, g^a, g^b, g^c, Z) with unknown $a, b, c \in \mathbb{Z}_p$, challenger $\mathcal{C}$'s goal is to decide $Z = e(g, g)^{abc}$ or Z is a random value. Challenger $\mathcal{C}$ works by interacting with $\mathcal{A}$ in the above security game as follows:

- Setup: Adversary $\mathcal{A}$ is given the public parameter $params = ((p, \mathbb{G}, \mathbb{G}_T, e), g, g_1, H_1, H_2, H_3, H_4, H_5, H_6)$ where $g_1 = g^a$ and $H_1, H_2, H_3, H_3, H_4, H_5, H_6$ are random oracles managed by challenger $\mathcal{C}$. The master secret key a is unknown to challenger $\mathcal{C}$.
- Phase 1: Adversary $\mathcal{A}$ adaptively asks the following queries:
  - *Hash Oracle Queries*. Adversary $\mathcal{A}$ freely queries H_i with $i \in \{1, 2, 3, 4, 5, 6\}$. Challenger $\mathcal{C}$ maintains six hash tables H_i-list with $i \in \{1, 2, 3, 4, 5, 6\}$. At the beginning, all of the tables are empty. Challenger $\mathcal{C}$ replies the queries as follows:

    $Hash_1$ Query (ID_j):

    If ID_j is on the H_1-list in the form of $\langle ID_j, Q_j, q_j, \varpi_j \rangle$, challenger $\mathcal{C}$ returns the predefined value Q_j; otherwise, it chooses $q_j \in_R \mathbb{Z}_p$ and generates a random $\varpi_j \in \{0, 1\}$, if $\varpi_j = 0$, challenger $\mathcal{C}$ computes $Q_j = g^{q_j}$; else it computes $Q_j = g^{bq_j}$ and adds $\langle ID_j, Q_j, q_j, \varpi_j \rangle$ into the H_1-list, and then it returns Q_j.

    $Hash_2$ Query $(\delta || M)$:

    If $\langle \delta || M \rangle$ is on the H_2-list in the form of $\langle \delta || M, r, g^r \rangle$, return r; otherwise, challenger $\mathcal{C}$ selects $r \in_R Z_p^*$ and adds $\langle \delta || M, r, g^r \rangle$ into the H_2-list, then it returns r.

    $Hash_3$ Query $(\delta \in \mathbb{G}_T)$:

    If δ is on the H_3-list in the form of $\langle \delta, X \rangle$, challenger $\mathcal{C}$ returns X; otherwise, it chooses $X \in_R \mathcal{M}$ and adds $\langle \delta, X \rangle$ into the H_3-list, then it returns X.

    $Hash_4$ Query $(ID_j || A || B || C || D_1 || \cdots || D_n)$:

    If $\langle ID_j || A || B || C || D_1 || \cdots || D_n \rangle$ is on the H_4-list in the form of $\langle ID_j || A || B || C || D_1 || \cdots || D_n, T_j, t_j \rangle$, challenger $\mathcal{C}$ returns the value T_j; otherwise, it chooses $t_j \in_R \mathbb{Z}_p$, computes $T_j = g^{t_j}$ and adds $\langle ID_j || A || B || C || D_1 || \cdots || D_n, T_j, t_j \rangle$ into the H_4-list, and then $\mathcal{C}$ returns T_j.

    $Hash_5$ Query $(ID_j, W_n = \{w_1, \cdots, w_k\})$:

    1. If $k = 1$, that is while $\langle ID_j, w_1 \rangle$ is on the H_5-list in the form of $\langle ID_j || w_1, \widehat{Q_1}, \widehat{q_1}, \widehat{\varpi_1} \rangle$, challenger $\mathcal{C}$ returns the value $\widehat{Q_1}$; otherwise, it picks $\widehat{q_1} \in_R \mathbb{Z}_p$ and $\widehat{\varpi_1} \in_R \{0, 1\}$. If $\widehat{\varpi_1} = 0$, it computes $Q_1 = g^{\widehat{q_1}}$;

else it computes $Q_1 = g^{b\widehat{q_1}}$. It adds $\langle ID_j||w_1, \widehat{Q_1}, \widehat{q_1}, \widehat{\varpi_1}\rangle$ into the H_5-list and responds with $\widehat{Q_1}$.

2. If $k \neq 1$, that is while $\langle ID_j, w_1, \cdots, w_k\rangle$ is on the H_5-list in the form of $\langle ID_j||w_1||\cdots||w_k, \widehat{Q_k}, \widehat{q_k}\rangle$, challenger $\mathcal{C}$ returns the value $\widehat{Q_k}$; otherwise, it picks $\widehat{q_k} \in_R \mathbb{Z}_p$ and computes $Q_k = g^{\widehat{q_k}}$. It adds $\langle ID_j||w_1||\cdots||w_k, \widehat{Q_k}, \widehat{q_k}\rangle$ into the H_5-list and then responds with $\widehat{Q_k}$.

$Hash_6$ Query ($\theta \in \mathcal{M}$):
If θ is on the H_6-list in the form of $\langle \theta, Y\rangle$, challenger $\mathcal{C}$ returns the value Y; otherwise, it chooses $Y \in_R \mathbb{G}$ and adds $\langle \theta, Y\rangle$ into the H_6-list, and then challenger $\mathcal{C}$ returns Y.

- *Extraction query(ID_j):* Challenger $\mathcal{C}$ recovers the tuple $\langle ID_j, Q_j, q_j, \varpi_j\rangle$ from the H_1-list. If $\varpi_j = 1$, challenger $\mathcal{C}$ outputs $\perp$ and aborts; otherwise, challenger $\mathcal{C}$ returns $sk_{ID_j} = g_1^{q_j}$ to adversary $\mathcal{A}$. (Note that $sk_{ID_j} = g_1^{q_j} = g^{aq_j} = Q_j^a = H_1(ID_j)^\alpha$, so that this is a proper private key for the identity ID_j).
- *Re-encryption key query(ID_i, ID_j, W_n):* Challenger $\mathcal{C}$ first picks $\delta' \in_R \mathbb{G}_T$, $\theta \in_R \mathcal{M}$ and recovers $\langle ID_i, Q_i, q_i, \varpi_i\rangle$ and $\langle ID_j, Q_j, q_j, \varpi_j\rangle$ from the H_1-list and $\langle \delta'||\theta, r', g^{r'}\rangle$ from the H_2-list, $\langle \delta', X\rangle$ from the H_3-list, $\langle ID_i||w_1, \widehat{Q_1}, \widehat{q_1}, \widehat{\varpi_1}\rangle$ and $\langle ID_i||w_1||\cdots||w_n, \widehat{Q_n}, \widehat{q_n}\rangle$ from the H_5-list and $\langle \theta, Y\rangle$ from the H_6-list. Lets $rk_1 = g^{r'}$, $rk_2 = \delta' \cdot e(g_1, Q_j)^{r'}$, $rk_3 = X \oplus \theta$. Then challenger $\mathcal{C}$ constructs $RK_1, RK_2^1, RK_2^2, \cdots, RK_2^n$ as follows:
  1. If $\varpi_i = 0$, challenger $\mathcal{C}$ picks $s_1, \cdots, s_n \in_R \mathbb{Z}_p$ and lets $RK_1 = g_1^{q_i} \cdot \widehat{Q_1}^{s_1} \cdots \widehat{Q_n}^{s_n} \cdot Y$, $RK_2^1 = g^{s_1}, \cdots, RK_2^n = g^{s_n}$.
  2. If $\varpi_i = 1$ and $\widehat{\varpi_1} = 1$: challenger $\mathcal{C}$ picks $s', s_2, \cdots, s_n \in_R \mathbb{Z}_p$ and sets $RK_1 = g^{b\widehat{q_1}s'}\widehat{Q_2}^{s_2} \cdots \widehat{Q_n}^{s_n} \cdot Y$, $RK_2^1 = g_1^{-q_i/\widehat{q_1}} g^{s'}$, $RK_2^2 = g^{s_2}, \cdots, RK_2^n = g^{s_n}$, where $s_1 = -aq_i/\widehat{q_1} + s'$.
  3. If $\varpi_i = 1$ and $\widehat{\varpi_1} = 0$: challenger $\mathcal{C}$ outputs $\perp$ and aborts.

  Finally, challenger $\mathcal{C}$ returns the re-encryption key $rk_{w|ID_i \to ID_j} = (rk_1, rk_2, rk_3, RK_1, RK_2^1, \cdots, RK_2^n)$ to adversary $\mathcal{A}$.
- *Hierarchical condition Re-encryption key query*$\langle rk_{W_n|ID_i \to ID_j}, W_{n+1}\rangle$: Challenger $\mathcal{C}$ first gets the re-encryption key $rk_{W_n|ID_i \to ID_j}$ for a condition $W_n = \{w_1, \cdots, w_n\}$, it first chooses $r', s_1, s_2, \cdots, s_n, s_{n+1} \in_R \mathbb{Z}_p$ and computes $rk_1' = rk_1 \cdot g^{r'}$, $rk_2' = rk_2 \cdot e(g_1, H_1(ID_j))^{r'}$, $rk_3' = rk_3$, $RK_1' = RK_1 \cdot H_5(ID_i||w_1||\cdots||w_{n+1})^{s_{n+1}} \cdot H_5(ID_i||w_1)^{s_1} \cdots H_5(ID_i||w_1||\cdots||w_n)^{s_n} \cdot H_6(\theta)$, $RK_2'^1 = RK_2^1 \cdot g^{s_1}, \cdots, RK_2'^n = RK_2^n \cdot g^{s_n}$, $RK_2'^{n+1} = g^{s_{n+1}}$. Challenger $\mathcal{C}$ returns the hierarchical conditional re-encryption key $rk_{W_{n+1}|ID_i \to ID_j}$ for the conditional $W_{n+1} = \{w_1, \cdots, w_n, w_{n+1}\}$ to adversary $\mathcal{A}$.
- *Re-encryption query*$(ID_i, ID_j, CT_{(ID_i, W_n)})$: Their exists the following two cases to generate the re-encrypted ciphertext:
  1. If $\varpi_i = 1$ and $\widehat{\varpi_1} = 0$, challenger $\mathcal{C}$ first parses the ciphertext $CT_{(ID_i, W_n)}$ as $(A, B, C, D_1, \cdots, D_n, S, W_n)$ and checks whether $e(SD_1 \cdots D_n, g) = e(H_4(ID_i||A||B||C||D_1||\cdots||D_n)H_5(ID_i||w_1) \cdots H_5(ID_i||w_1||\cdots||w_n), A)$. If not, it returns $\perp$; otherwise, challenger $\mathcal{C}$ checks whether there exists a tuple $\langle \delta||M, r, g^r\rangle$ from the

H_2-list such that $A = g^r$. If no, it returns $\perp$; otherwise, $\mathcal{C}$ recovers the tuple $\langle ID_j, Q_j, q_j, \varpi_j\rangle$ from the H_1-list and then it picks $\theta \in_R \mathcal{M}$, $\delta' \in_R \mathbb{G}_T$, $\mathcal{C}$ recovers the tuple $\langle \delta', X\rangle$ from the H_3-list and sets $r' = H_2(\delta'||\theta), rk_1 = g^{r'}, rk_2 = \delta' \cdot e(g_1, Q_j)^{r'}$, $rk_3 = X \oplus \theta$. Next, $\mathcal{C}$ recovers the tuple $\langle\theta, Y\rangle$ from the H_6-list and sets $B' = \delta / e(A, Y)$. Finally, $\mathcal{C}$ outputs the transformed ciphertext $CT_{(ID_j, W_n)} = (A, B', C, rk_1, rk_2, rk_3)$ to adversary $\mathcal{A}$.

2. Otherwise, challenger $\mathcal{C}$ first queries the re-encryption key to get $rk_{W_n|ID_i \to ID_j}$, and then it runs ReEnc $(rk_{W_n|ID_i \to ID_j}, CT_{(ID_i, W_n)})$ algorithm to obtain the transformed ciphertext $CT_{(ID_j, W_n)}$. Finally challenger $\mathcal{C}$ returns the transformed ciphertext $CT_{(ID_j, W_n)}$ to adversary $\mathcal{A}$.

- *Decryption query*$(ID, CT_{(ID,W_n)})$: Challenger $\mathcal{C}$ checks whether $CT_{(ID,W_n)}$ is an initial or a transformed ciphertext.
  1. For an initial ciphertext, challenger $\mathcal{C}$ first extracts $CT_{(ID,W_n)}$ as $(A, B, C, D_1, \cdots, D_n, S, W_n)$. Then it recovers a tuple $\langle ID, Q, q, \varpi\rangle$ from the H_1-list. If $\varpi = 0$ (meaning $sk_{ID} = g_1^q$), challenger $\mathcal{C}$ decrypts the ciphertext $CT_{(ID,W_n)}$ using sk_{ID}; otherwise, challenger $\mathcal{C}$ first checks whether $e(SD_1 \cdots D_n, g) = e(H_4(ID_i||A||B||C|| D_1|| \cdots ||D_n) H_5(ID_i||w_1) \cdots H_5(ID_i||w_1|| \cdots ||w_n), A)$ holds. If no, it returns $\perp$; else challenger $\mathcal{C}$ searches the tuple $\langle \delta||M, r, g^r\rangle$ from the H_2-list such that $A = g^r$. If it cannot find such tuple, it returns $\perp$; else it searches whether there exists a tuple $\langle\delta, X\rangle$ from the H_3-list such that $M \oplus X = C$, a tuple $\langle ID||w_1, \widehat{Q_1}, \widehat{q_1}, \widehat{\varpi_1}\rangle$ and some tuples $\{\langle ID||w_1|| \cdots ||w_k, \widehat{Q_k}, \widehat{q_k}\rangle\}_{1 \le k \le n}$ from the H_5-list and a tuple $\langle ID||A||B||C||D_1|| \cdots ||D_n, T, t\rangle$ from the H_4-list, such that $\widehat{Q_1}^r = D_1, \cdots, \widehat{Q_k}^r = D_k$ and $T^r = S$. If not, it returns $\perp$; otherwise, challenger $\mathcal{C}$ returns $M = C \oplus X$ to adversary $\mathcal{A}$.
  2. For a transformed ciphertext, challenger $\mathcal{C}$ first parses $CT_{(ID,W_n)}$ as $(A, B', C, rk_1, rk_2, rk_3)$. Then challenger $\mathcal{C}$ recovers tuple $\langle ID, Q, q, \varpi\rangle$ from the H_1-list. If $\varpi = 0$ (meaning $sk_{ID} = g_1^q$), challenger $\mathcal{C}$ decrypts the ciphertext $CT_{(ID,W_n)}$ using sk_{ID}; otherwise, challenger $\mathcal{C}$ searches whether there exists a tuple $\langle\delta'||\theta, r', g^{r'}\rangle$ from the H_2-list such that $rk_1 = g^{r'}$. If not, it returns $\perp$; else searches whether there exists a tuple $\langle\delta', X\rangle$ from the H_3-list and a tuple $\langle ID, Q, q, 1\rangle$ from the H_1-list such that $\theta \oplus X = C$ and $\delta' \cdot e(g_1, Q)^{r'} = rk_2$. If not, it returns $\perp$; otherwise, challenger $\mathcal{C}$ recovers $\langle\theta, Y\rangle$ from the H_6-list, and it computes $\delta = B' \cdot e(A, Y)$ and $M = H_3(\delta) \oplus C$. Finally, challenger $\mathcal{C}$ returns M to adversary $\mathcal{A}$.

– Challenge: Adversary $\mathcal{A}$ outputs an identity ID^*, a condition W_n^* of depth n and two different plaintexts $M_0, M_1 \in \mathcal{M}$. Challenger $\mathcal{C}$ recovers the tuple $\langle ID^*, Q^*, q^*, \varpi^*\rangle$ from the H_1-list, a tuple $\langle ID^*||w_1^*, \widehat{Q^*}, \widehat{q^*}, \widehat{\varpi^*}\rangle$ and several tuples $\{\langle ID^*||w_1^*|| \cdots ||w_k^*, \widehat{Q_k^*}, \widehat{q_k^*}\rangle\}_{1 \le k \le n}$ from the H_5-list. If $\varpi^* = 0$ or $\widehat{\varpi^*} = 1$, challenger $\mathcal{C}$ outputs $\perp$ and aborts; else challenger $\mathcal{C}$ first picks $\beta \in_R \{0, 1\}$, $\delta^* \in_R \mathbb{G}_T$, $X^* \in_R \{0, 1\}^n$, and then it inserts the tuple $\langle\delta^*, X^*\rangle$ into the

H_3-list and the tuple $\langle \delta^*, M_\beta, \cdot, g^c \rangle$ into the H_2-list. Next challenger $\mathcal{C}$ sets $A^* = g^c$, $B^* = \delta^* \cdot T^{q^*}$, $C^* = X^* \oplus M_\beta, D_1^* = g^{c\widehat{q^*}}, \cdots, D_n^* = g^{c\widehat{q_n^*}}$ and selects $t^* \in_R \mathbb{Z}_p$, and then it inserts the tuple $\langle ID^* || A^* || B^* || C^* || D_1^* || \cdots || D_n^*, g^{t^*}, t^* \rangle$ into the H_4-list, and sets $S^* = g^{ct^*}$. Finally, challenger $\mathcal{C}$ sends the challenge ciphertext $CT^*_{(ID^*, W_n^*)} = (A^*, B^*, C^*, D_1^*, \cdots, D_n^*, S^*)$ to adversary $\mathcal{A}$.

- Phase 2: Adversary $\mathcal{A}$ continues to adaptively issue queries as in Phase 1. But it needs to satisfy the conditions which are described in the above security model.
- Guess: Adversary $\mathcal{A}$ outputs a guess $\beta' \in \{0, 1\}$.

5 Conclusion

In this paper, we propose an identity-based hierarchical conditional proxy re-encryption scheme, which is the first of its type. The new scheme allows delegator to achieve more flexibly encrypted data sharing. The scheme is proved secure against chosen-ciphertext attacks in the random oracle model. Via comparison, we show the flexibility and scalability of our scheme. This paper leaves some interesting open problems, for example, how could we prove the security in the standard model, and how to reduce the re-encryption key size to constant.

Acknowledgment. This work was supported by National Science Foundation of China (No. 61572131), Guangdong Provincial Science and Technology Plan Projects (No. 2016A010101034) and Project of Internation as well as Hongkong, Macao & Taiwan Science and Technology Cooperation Innovation Platform in Universities in Guangdong Province (No. 2015KGJHZ027).

References

1. Ateniese, G., Kevin, F., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Trans. Inf. Syst. Secur. **9**(1), 1–30 (2006)
2. Blaze, M., Bleumer, G., Strauss, M.: Divertible protocols and atomic proxy cryptography. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 127–144. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054122
3. Canetti, R., Hohenberger, S.: Chosen-ciphertext secure proxy re-encryption. In: Proceedings of the 2007 ACM Conference on Computer and Communications Security (CCS 2007), Alexandria, Virginia, USA, 28–31 October 2007, pp. 185–194 (2007)
4. Castiglione, A., De Santis, A., Masucci, B., Palmieri, F., Castiglione, A., Huang, X.: Cryptographic hierarchical access control for dynamic structures. IEEE Trans. Inf. Forensics Secur. **11**(10), 2349–2364 (2016)
5. Castiglione, A., De Santis, A., Masucci, B., Palmieri, F., Castiglione, A., Li, J., Huang, X.: Hierarchical and shared access control. IEEE Trans. Inf. Forensics Secur. **11**(4), 850–865 (2016)
6. Chu, C.-K., Tzeng, W.-G.: Identity-based proxy re-encryption without random oracles. In: Garay, J.A., Lenstra, A.K., Mambo, M., Peralta, R. (eds.) ISC 2007. LNCS, vol. 4779, pp. 189–202. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75496-1_13

7. Deng, R.H., Weng, J., Liu, S., Chen, K.: Chosen-ciphertext secure proxy re-encryption without pairings. In: Franklin, M.K., Hui, L.C.K., Wong, D.S. (eds.) CANS 2008. LNCS, vol. 5339, pp. 1–17. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89641-8_1
8. Fang, L., Susilo, W., Ge, C., Wang, J.: Interactive conditional proxy re-encryption with fine grain policy. J. Syst. Softw. **84**(12), 2293–2302 (2011)
9. Giuseppe, A., Kevin., Matthew, G., Susan, H.: Improved proxy re-encryption schemes with applications to secure distributed storage. In: Proceedings of the Network and Distributed System Security Symposium (NDSS 2005), San Diego, California, USA (2005)
10. Green, M., Ateniese, G.: Identity-based proxy re-encryption. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 288–306. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72738-5_19
11. Hanaoka, G., Kawai, Y., Kunihiro, N., Matsuda, T., Weng, J., Zhang, R., Zhao, Y.: Generic construction of chosen ciphertext secure proxy re-encryption. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 349–364. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27954-6_22
12. He, K., Weng, J., Deng, R.H., Liu, J.K.: On the security of two identity-based conditional proxy re-encryption schemes. Theor. Comput. Sci. **652**, 18–27 (2016)
13. He, K., Weng, J., Liu, J.K., Zhou, W., Liu, J.-N.: Efficient fine-grained access control for secure personal health records in cloud computing. In: Chen, J., Piuri, V., Su, C., Yung, M. (eds.) NSS 2016. LNCS, vol. 9955, pp. 65–79. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46298-1_5
14. Lee, C.-C., Li, C.-T., Chen, C.-L., Chiu, S.-T.: A searchable hierarchical conditional proxy re-encryption scheme for cloud storage services. ITC **45**(3), 289–299 (2016)
15. Liang, K., Au, M.H., Liu, J.K., Susilo, W., Wong, D.S., Yang, G., Yu, Y., Yang, A.: A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing. Future Gener. Comput. Syst. **52**, 95–108 (2015)
16. Liang, K., Chu, C.-K., Tan, X., Wong, D.S., Tang, C., Zhou, J.: Chosen-ciphertext secure multi-hop identity-based conditional proxy re-encryption with constant-size ciphertexts. Theor. Comput. Sci. **539**, 87–105 (2014)
17. Liang, K., Fang, L., Wong, D.S., Susilo, W.: A ciphertext-policy attribute-based proxy re-encryption scheme for data sharing in public clouds. Concurr. Comput. Pract. Exp. **27**(8), 2004–2027 (2015)
18. Liang, K., Liu, J.K., Wong, D.S., Susilo, W.: An efficient cloud-based revocable identity-based proxy re-encryption scheme for public clouds data sharing. In: Kutyłowski, M., Vaidya, J. (eds.) ESORICS 2014. LNCS, vol. 8712, pp. 257–272. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11203-9_15
19. Liang, K., Liu, Z., Tan, X., Wong, D.S., Tang, C.: A CCA-secure identity-based conditional proxy re-encryption without random oracles. In: Kwon, T., Lee, M.-K., Kwon, D. (eds.) ICISC 2012. LNCS, vol. 7839, pp. 231–246. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37682-5_17
20. Liang, K., Su, C., Chen, J., Liu, J.K.: Efficient multi-function data sharing and searching mechanism for cloud-based encrypted data. In: Chen, X., Wang, X., Huang, X. (eds.) Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security (AsiaCCS 2016), Xi'an, China, May 30 - June 3, 2016, pp. 83–94. ACM (2016)
21. Liang, K., Susilo, W.: Searchable attribute-based mechanism with efficient data sharing for secure cloud storage. IEEE Trans. Inf. Forensics Secur. **10**(9), 1981–1992 (2015)

22. Liang, K., Susilo, W., Liu, J.K.: Privacy-preserving ciphertext multi-sharing control for big data storage. IEEE Trans. Inf. Forensics Secur. **10**(8), 1578–1589 (2015)
23. Liang, X., Cao, Z., Lin, H., Shao, J.: Attribute based proxy re-encryption with delegating capabilities. In: Proceedings of the 2009 ACM Symposium on Information, Computer and Communications Security (ASIACCS 2009), Sydney, Australia, 10–12 March 2009, pp. 276–286 (2009)
24. Libert, B., Vergnaud, D.: Unidirectional chosen-ciphertext secure proxy re-encryption. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 360–379. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78440-1_21
25. Libert, B., Vergnaud, D.: Unidirectional chosen-ciphertext secure proxy re-encryption. IEEE Trans. Inf. Theory **57**(3), 1786–1802 (2011)
26. Wang, L., Wang, L., Mambo, M., Okamoto, E.: New identity-based proxy re-encryption schemes to prevent collusion attacks. In: Joye, M., Miyaji, A., Otsuka, A. (eds.) Pairing 2010. LNCS, vol. 6487, pp. 327–346. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17455-1_21
27. Lin, S., Zhang, R., Wang, M.: Verifiable attribute-based proxy re-encryption for secure public cloud data sharing. Secur. Commun. Netw. **9**(12), 1748–1758 (2016)
28. Luo, S., Shen, Q., Chen, Z.: Fully secure unidirectional identity-based proxy re-encryption. In: Kim, H. (ed.) ICISC 2011. LNCS, vol. 7259, pp. 109–126. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31912-9_8
29. Matsuda, T., Nishimaki, R., Tanaka, K.: CCA proxy re-encryption without bilinear maps in the standard model. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 261–278. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13013-7_16
30. Matsuo, T.: Proxy re-encryption systems for identity-based encryption. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (eds.) Pairing 2007. LNCS, vol. 4575, pp. 247–267. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-73489-5_13
31. Mizuno, T., Doi, H.: Secure and efficient IBE-PKE proxy re-encryption. IEICE Trans. **94–A**(1), 36–44 (2011)
32. Nabeel, M., Bertino, E.: Privacy preserving delegated access control in public clouds. IEEE Trans. Knowl. Data Eng. **26**(9), 2268–2280 (2014)
33. Shao, J., Cao, Z.: Multi-use unidirectional identity-based proxy re-encryption from hierarchical identity-based encryption. Inf. Sci. **206**, 83–95 (2012)
34. Shao, J., Rongxing, L., Lin, X., Liang, K.: Secure bidirectional proxy re-encryption for cryptographic cloud storage. Pervasive Mobile Comput. **28**, 113–121 (2016)
35. Smith, T.: DVD jon: Buy DRM-less tracks from Apple iTunes, January 2005. http://www.theregister.co.uk/2005/03/18/itunespymusique
36. Tang, Q.: Type-based proxy re-encryption and its construction. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) INDOCRYPT 2008. LNCS, vol. 5365, pp. 130–144. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89754-5_11
37. Isshiki, T., Nguyen, M.H., Tanaka, K.: Proxy re-encryption in a stronger security model extended from CT-RSA2012. In: Dawson, E. (ed.) CT-RSA 2013. LNCS, vol. 7779, pp. 277–292. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36095-4_18
38. Wang, L., Wang, L., Mambo, M., Okamoto, E.: Identity-based proxy cryptosystems with revocability and hierarchical confidentialities. IEICE Trans. **95–A**(1), 70–88 (2012)
39. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_7

40. Weng, J., Chen, M.-R., Yang, Y., Deng, R.H., Chen, K., Bao, F.: CCA-secure unidirectional proxy re-encryption in the adaptive corruption model without random oracles. Sci. China Inf. Sci. **53**(3), 593–606 (2010)
41. Weng, J., Deng, R.H., Ding, X., Chu, C.-K., Lai, J.: Conditional proxy re-encryption secure against chosen-ciphertext attack. In: Proceedings of the 2009 ACM Symposium on Information, Computer and Communications Security (ASIACCS 2009), Sydney, Australia, 10–12 March 2009, pp. 322–332 (2009)
42. Weng, J., Yang, Y., Tang, Q., Deng, R.H., Bao, F.: Efficient conditional proxy re-encryption with chosen-ciphertext security. In: Samarati, P., Yung, M., Martinelli, F., Ardagna, C.A. (eds.) ISC 2009. LNCS, vol. 5735, pp. 151–166. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04474-8_13