2013

Identity-based data storage in cloud computing

Jinguang Han
University of Wollongong, jh843@uowmail.edu.au

Willy Susilo
University of Wollongong, wsusilo@uow.edu.au

Yi Mu
University of Wollongong, ymu@uow.edu.au

Publication Details

Han, J., Susilo, W. & Mu, Y. (2013). Identity-based data storage in cloud computing. Future Generation Computer Systems: the international journal of grid computing: theory, methods and applications, 29 (3), 673-681.

Identity-based data storage in cloud computing

Abstract
Identity-based proxy re-encryption schemes have been proposed to shift the burden of managing numerous files from the owner to a proxy server. Nevertheless, the existing solutions suffer from several drawbacks. First, the access permission is determined by the central authority, which makes the scheme impractical. Second, they are insecure against collusion attacks. Finally, only queries from the same domain (intra-domain) are considered. We note that one of the main applications of identity-based proxy re-encryption schemes is in the cloud computing scenario. Nevertheless, in this scenario, users in different domains can share files with each other. Therefore, the existing solutions do not actually solve the motivating scenario, when the scheme is applicable for cloud computing. Hence, it remains an interesting and challenging research problem to design an identity-based data storage scheme which is secure against collusion attacks and supports intra-domain and inter-domain queries. In this paper, we propose an identity-based data storage scheme where both queries from the intra-domain and inter-domain are considered and collusion attacks can be resisted. Furthermore, the access permission can be determined by the owner independently.

Keywords
era2015

Disciplines
Engineering | Science and Technology Studies

Publication Details
Han, J., Susilo, W. & Mu, Y. (2013). Identity-based data storage in cloud computing. Future Generation Computer Systems: the international journal of grid computing: theory, methods and applications, 29 (3), 673-681.

Identity-Based Data Storage in Cloud Computing

Jinguang Han[a,b,*], Willy Susilo[a], Yi Mu[a]

[a] *Centre for Computer and Information Security Research, School of Computer Science and Software Engineering, University of Wollongong, NSW2522, Australia.*
[b] *College of Sciences, Hohai University, Nanjing 210098, China*

Abstract

Identity-based proxy re-encryption schemes have been proposed to shift the burden of managing numerous files from the owner to a proxy server. Nevertheless, the existing solutions suffer from several drawbacks. First, the access permission is determined by the central authority, which makes the scheme impractical. Second, they are insecure against the collusion attacks. Finally, only queries from the same domain (intra-domain) are considered. We note that one of the main applications of identity-based proxy re-encryption schemes is in the cloud computing scenario. Nevertheless, in this scenario, users in different domains can share files with each other. Therefore, the existing solutions do not actually solve the motivating scenario, when the scheme is applicable for cloud computing. Hence, it remains an interesting and challenging research problem to design an identity-based data storage scheme which is secure against the collusion attacks and supports intra-domain and inter-domain queries. In this paper, we propose an identity-based data storage scheme where both queries from the intra-domain and inter-domain are considered and collusion attacks can be resisted. Furthermore, the access permission can be determined by the owner independently.

Keywords: Identity-based System, Data Storage, Query, Security

*Corresponding author.Tel.: +61 242214824; fax: +61 242215550
Email addresses: jh843@uowmail.edu.au (Jinguang Han), wsusilo@uow.edu.au (Willy Susilo), ymu@uow.edu.au (Yi Mu)

1. Introduction

In cloud computing, users can utilize powerful computing resources and obtain ample storage spaces. This is called database-as-a-service (DaaS)[1,2], software-as-a-service (SaaS)[3,4,5] or infrastructure as a service (IaaS)[5,6,7]. Although it brings users with great convenience, the security issues in cloud computing are the primary concerns of users[3,8].

Armbrust *et al.*[3] gave a view of cloud computing and listed top ten obstacles and opportunities for cloud computing. The first obstacle is availability/business continuity where users in cloud computing are concerned about whether the computing services have adequate availability. The approach to overcome this obstacle is to use multiple cloud servers. The second obstacle is data lock-in, where users cannot extract their data from one site to run on another site. The potential solution for this obstacle can be hybridised: cloud computing and standardized APIs. The third obstacles is data confidentiality and audibility. Security issues in cloud computing include threats from inside and outside the cloud. This obstacle could be overcome by applying encryption and firewall techniques. The cloud server should be responsible for physical layer security; while the user should be responsible for the application layer security. The fourth obstacle is data transfer bottlenecks as users want to transfer data with minimized cost. The method to reduce the high cost of internet transfer is shipping disk. The fifth obstacle is performance unpredictability as network and disk I/O sharing are problematic. This obstacle can be handled by improving the architectures and operating systems. The sixth obstacle is scalable storage. The method to resolve this obstacle is to create a storage system which would not only satisfy existing programmer expectations in regard to durability, high availability, and the ability to manage and query data, but also combine them with the cloud advantages of scaling arbitrarily up and down on demand. The seventh obstacle is bugs in large-scale distributed systems, namely how to remove the errors in distributed systems. This obstacle can be overcome by reliance on virtual machines. The eighth obstacle is scaling quickly as the cost is computed differently, depending on the virtualization level. The method to handle this obstacle is to scale automatically so that users can save their money. The ninth obstacle is reputation rate sharing where one user's bad behavior can affect the reputation of other users using the same cloud. The opportunity to resolve this obstacle is to create reputation-guarding services. The tenth obstacle is software licensing, namely the licensing model of commercial soft-

ware cannot match well with the utility computing. This obstacle can be overcome by providing open source or changing the licensing structures. In these obstacles, the first three affect the adoption of cloud computing, the next five affect the growth of cloud computing, and the last two are policy and business obstacles. Therefore, to improve the adoption of cloud computing, a desirable scheme should provide the following properties. First, it can provide confidentiality to the sensitive data. Second, multiple service providers can co-exist. Third, it can make services available across domains.

In data storage systems[9,10], users can store their data to an external proxy servers to reduce the maintenance cost and enhance the access and availability. To protect the confidentiality of the outsourced files, the owner encrypts them prior to outsourcing them to an untrusted proxy server. The proxy server can perform some functions on the ciphertexts, such that an authorized user can access the desired sensitive files. Samarati and Vimercati[11] addressed the privacy issues in data outsourcing expanding from the data confidentiality to data utility, and pointed out some research topics in the protection of the outsourced data. Kher and Kim[12] surveyed the data storage systems comprehensively and divided them into three types based on their security services: networked file systems (NFS), storage-based intrusion detection systems (SBIDS) and cryptographic file systems (CFS).

In this paper, we propose an identity-based data storage scheme which is applicable to cloud computing.

1.1. Related Work

Proposed by Mambo and Okamoto[13], a proxy cryptosystem is a system where a user can delegate his/her decryption right to a designated decrypter. Subsequently, Blaze, Bleumer and Strauss[14] extended this notion by introducing the concept of proxy re-encryption (PRE). In this new cryptographic primitive, a proxy server can transfer a ciphetext designated for one user to another ciphertext designated for another user without the need to have the knowledge on the plaintext. Since then, some useful PRE schemes have been proposed accordingly[15,16,17,18,19]. Weng, *et al.*[20] proposed a new PRE scheme called conditional PRE (C-PRE). In this scheme, only the ciphertexts which satisfy the condition given by the original decryptor can be transferred to the ciphertexts for a designated decryptor, instead of all ciphertexts. Subsequently, Fang, *et al.*[21] extended the notion of C-PRE to be hierarchical C-PRE (HC-PRE). In this scheme, a proxy server can delegate his re-encryption right to other proxy servers under a specified condition. Furthermore, they

pointed out some application scenarios, such as ZigBee security for visitors in home automaton, privacy-preserving location sharing protocol *etc.*

Ateniese, *et al.*[22] improved the concept of PRE and employed it to data storage. In their scheme, the owner encrypts his/her files and outsources them to a proxy server. The proxy server can transfer a ciphertext for the owner to a ciphetext for the requester if and only if he has obtained a re-encryption key from the owner.

Introduced by Shamir[23], identity-based encryption (IBE) is an efficient cryptographic system where the public key can be any arbitrary string and the secret key is extracted from a trusted party called private key generator (PKG). Boneh and Franklin[24] proposed the first practical IBE scheme based on the bilinear group. Since its seminal introduction, IBE schemes[25,26] have been discussed extensively as in this new cryptographic notion, the need for public key infrastructure (PKI) has been eliminated efficiently.

Ivan and Dodis[27] proposed two identity-based proxy encryption schemes where the master secret key held by the PKG is split into two parts. One is for the user and the other is for the proxy server. Then, the user can cooperate with the proxy server to decrypt a ciphertext. Unfortunately, these schemes are not secure against the collusion attacks[22] as the user and the proxy server can collaborate to compute the master secret key.

Green and Ateniese[28] introduced the concept of identity-based proxy re-encryption (IBPRE). In an IBPRE scheme, a proxy server can transfer a ciphertext encrypted under one identity to a ciphertext encrypted under another identity without learning the contents of the plaintext.

Subsequently, Matsuo[29] proposed two IBPRE schemes. In the first scheme, a ciphertext encrypted under traditional PKI can be transferred to a ciphertext encrypted under an identity in IBE schemes. Meanwhile, the second scheme is proposed to transfer a ciphetext encrypted under the identity of the original decyprter to a ciphertext encrypted under the identity of the designated decrypter.

Wang, *et al.*[30,31] proposed two new IBPRE schemes. In[30], they discussed the relationships between the IBPRE secure against chosen plaintext attacks and the PRE properties: unidirectional, nontransferable and collusion safe. It was proposed[31] that the proxy server can transfer a ciphertext for the original decrypter to a ciphertext for the designated decrypter, and decrypt the ciphertext for the original decrypter. Additionally, the original decrypter can revoke the decryption and re-encryption rights of the proxy server. In the schemes due to[29,30,31], the re-encryption key must be computed with the

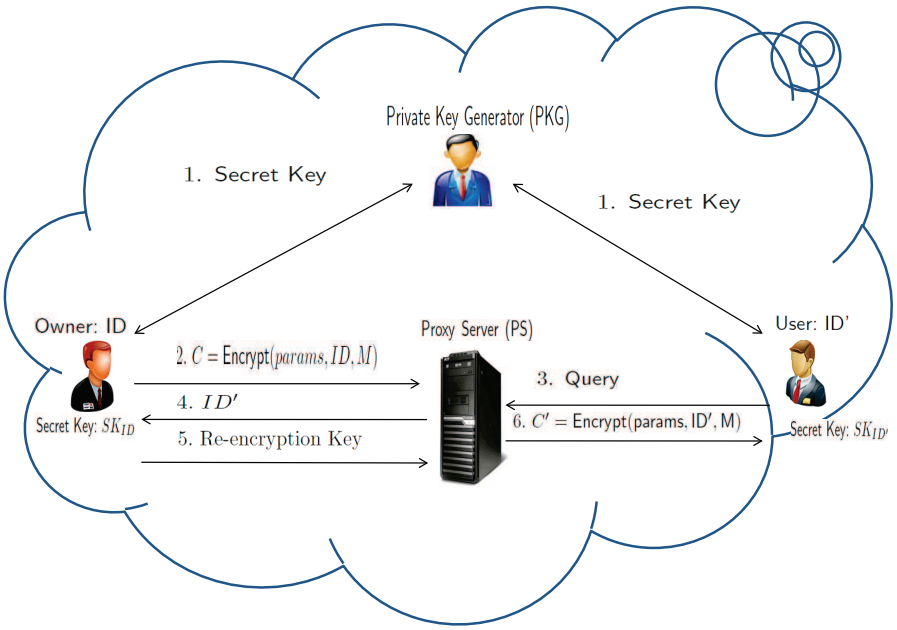

Figure 1: Identity-based Data Storage Supporting Intra-Domain Query

help of the PKG.

Chu and Tzeng[32] proposed two IBPRE schemes in the standard model. Unfortunately, these schemes[28,32] are not secure against the collusion attacks. If the designated decrypter can compromise the proxy server, he can obtain the secret key of the original decrypter.

In all the above IBPRE schemes, only the *intra-domain setting* is considered, namely both the original decrypter and the designated decrypter should come from the *same domain*. Tang, Hartel and Jonker[33] made an important step by firstly proposing the IBPRE scheme where the *inter-domain setting* is considered, namely the proxy server can transfer a ciphertext for the original decrypter in a domain to a ciphertext for the designated decrypter in another domain. Although this scheme is not secure against the collusion attacks, they made an important step from intra-domain IBPRE to inter-domain IBPRE. We review this scheme in section 2.4.

To clarify, we depict IBPRE schemes which support intra-domain query and inter-domain query in Figure 1 and Figure 2, respectively.

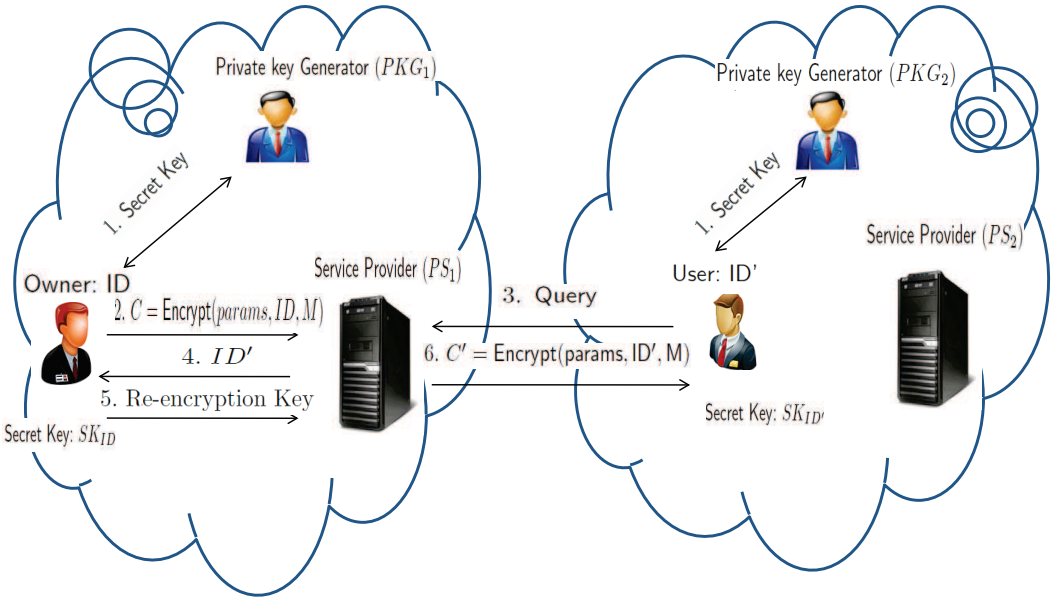

Figure 2: Identity-based Data Storage Supporting Inter-Domain Query

1.2. Our Contribution

Cloud computing is a distributed system where multiple domains co-exist together. It is desirable that users in different domains can share sensitive data with others. Therefore, a sound identity-based data storage scheme in cloud computing should support not only the intra-domain query but also the inter-domain query. However, current identity-based proxy re-encryption schemes cannot be employed in cloud computing as they cannot support inter-domain query and resist collusion attacks. In this paper, we propose an identity-based data storage scheme which support both intra-domain and inter-domain queries. In our scheme, the re-encryption key is computed by the data owner independently without the help of the PKG. For one query, the requester can only access one file of the owner, while the requester and the proxy server can cooperatively access all the files of the owner in previous schemes as the access permission (re-encryption key) is not bound to the ciphertext in these schemes. Furthermore, our scheme is secure against the collusion attacks and selective-identity secure in the standard model. Therefore, our scheme can improve the adoption of cloud computing as it can overcome the first three obstacles[3].

1.3. Roadmap

In Section 2, we introduce the preliminaries used throughout this paper. An identity-based data storage scheme supporting intr-domain and inter-domain queries is proposed and proven in Section 3. We implement our scheme in the PBC library in Section 4. Section 5 concludes this paper.

2. Preliminaries

In the rest of this paper, by $s \xleftarrow{R} S$, we denote s is selected from S at random. If S is a finite set, by $s \xleftarrow{R} S$, we denote s is selected uniformly from S. By $F(x) \rightarrow y$, we denote y is obtained by running the algorithm F on input x. A function $\epsilon : \mathbb{Z} \rightarrow \mathbb{R}$ is negligible if, for all $z \in \mathbb{Z}$, there exists an integer $k \in \mathbb{Z}$ such that $\epsilon(x) \leq \frac{1}{x^z}$ when $x > k$.

2.1. Identity-based Data Storage

There are four entities in an identity-based data storage scheme: the private key generator (PKG), the data owner, the proxy server (PS) and the requester. The PKG validates the users' identities and issues secret keys to them. The data owner encrypts his files and outsources them to the proxy server. He validates the requesters and issues access permissions to the proxy sever. The proxy server stores the ciphertexts and can transfer them to ciphertexts for the requester when he obtains corresponding re-encryption keys from the owner. The requester can decrypt the re-encrypted ciphertext. An identity-based data storage scheme supporting intra-domain and inter-domain queries consists of the following algorithms:

$\mathsf{Setup}(1^\ell) \rightarrow (params, (MSK_1, PK_1), (MSK_2, PK_2))$. This algorithm takes as inputs a security parameters 1^ℓ and outputs the public parameters $params$, master secret-public key pairs (MSK_1, PK_1) and (MSK_2, PK_2) for PKG_1 in domain $\mathfrak{D}_1$ and PKG_2 in domain $\mathfrak{D}_2$, respectively.

$\mathsf{KeyGen}(params, ID, MSK_i) \rightarrow SK_{ID}$. This algorithm takes as inputs the public parameters $params$, an identity ID in the domain $\mathfrak{D}_i$ and the master secret key MSK_i, and outputs a secret key SK_{ID} for the identity ID, where $i \in \{1, 2\}$.

Encryption$(params, ID, M) \rightarrow CT$. This algorithm takes as inputs the public parameters $params$, the identity ID and the message M, and outputs the ciphertext $CT = \mathsf{Encrypt}(params, ID, M)$. It sends the ciphertext CT to the proxy server PS_i in the domain $\mathfrak{D}_i$, where $i = \{1, 2\}$.

Query$(ID', SK_{ID'}, CT) \rightarrow \Theta$. The requester R with identity ID' queries the proxy server on the ciphertext CT. This algorithm takes as input the requester's identity ID', secrete key $SK_{ID'}$ and the ciphertext CT, and outputs an authentication information Θ. The requester sends Θ to the proxy server PS_i. PS_i works as follows:

1. Both the owner and the requester are in the same domain. The proxy server PS_i sends (ID', Θ, CT) to the owner.
2. The owner and the requester are in different domains. Suppose that the owner is in the domain $\mathfrak{D}_i$ and the requester is in the domain $\mathfrak{D}_{3-i}$, where $i = \{1, 2\}$. The proxy server PS_i sends $(ID', PK_{3-i}, \Theta, CT)$ to the owner.

Permission$(params, ID', CT, SK_{ID}) \rightarrow AK$. The owner validates the requester by verifying the authentication information Θ. If the requester is legal, this algorithm takes as inputs the public parameters $params$, the requeseter's identity ID', the intended ciphertext CT and the owner's secret key SK_{ID}, and outputs an access key (re-encryption key) AK. It sends AK to the proxy server PS_i.

Re-encryption$(params, ID', AK, CT) \rightarrow CT'$. This algorithm takes as inputs the public parameters $params$, the requester's identity ID', the access key AK and the ciphertext CT, and outputs the re-encrypted ciphertext $CT' = \mathsf{Encrypt}(params, ID', M)$.

Decryption. There are two algorithms. One is for the owner and the other is for the requester.

1. $\mathsf{Decryption}_1(params, SK_{ID}, CT) \rightarrow M$. This algorithm takes as inputs the public parameters $params$, the owner's secrete key SK_{ID} and the ciphertext CT, and outputs the message M.

2. $\mathsf{Decryption}_2(params, SK_{ID'}, CT') \rightarrow M$. This algorithm takes as inputs the public parameters $params$, the requester's secrete key $SK_{ID'}$ and the re-encrypted ciphertext CT', and outputs the message M.

Definition 1. *We say an identity-based data storage scheme supporting intra-domain and inter-domain queries is correct if*

$$\Pr\left[\begin{array}{l|l} \mathsf{Decryption}_1(params, & \mathsf{Setup}(1^\ell) \rightarrow (params, MSK, PK); \\ SK_{ID}, CT) \rightarrow M & \mathsf{KeyGen}(params, ID, MSK) \rightarrow SK_{ID}; \\ & \mathsf{Encryption}(params, ID, M) \rightarrow CT \end{array}\right] = 1$$

and

$$\Pr\left[\begin{array}{l|l} & \mathsf{Setup}(1^\ell) \rightarrow (params, MSK, PK); \\ \mathsf{Decryption}_2(params & \mathsf{KeyGen}(params, ID, MSK) \rightarrow SK_{ID}; \\ SK_{ID'}, CT') \rightarrow M & \mathsf{KeyGen}(params, ID', MSK) \rightarrow SK_{ID'}; \\ & \mathsf{Permission}(params, ID', CT, SK_{ID}) \\ & \rightarrow AK; \\ & \textit{Re-encryption}(params, ID', AK, CT) \\ & \rightarrow CT' \end{array}\right] = 1$$

where the probability is taken over the random coins which all the algorithms in the scheme consumes.

2.2. Security Model

The following game is used to formalize the security model of identity-based data storage scheme supporting intra-domain and inter-domain queries. This model is derived from the selective-identity secure IBE scheme[34]. The game is run between a challenger $\mathfrak{C}$ and an adversary $\mathfrak{A}$ as follows:

Initialization. $\mathfrak{A}$ submits an identity ID^* with which he wants to be challenged to $\mathfrak{C}$. Let ID^* be in the domain $\mathfrak{D}_i$ where $i \in \{1, 2\}$.

Setup. $\mathfrak{C}$ runs $\mathsf{Setup}(1^\ell)$ to generate the public parameters $params$ and the secret-public key pairs (MSK_1, PK_1) for the PKG_1 in $\mathfrak{D}_1$ and (MSK_2, PK_2) for the PKG_2 in $\mathfrak{D}_2$. It sends $(params, PK_1, PK_2)$ to $\mathfrak{A}$.

Phase 1. $\mathfrak{A}$ can adaptively make the following queries:

1. Secret Key Query. $\mathfrak{A}$ can query secret key for an identity ID in $\mathfrak{D}_i$ or $\mathfrak{D}_{3-i}$, where the only constrains is $ID \neq ID^*$ and $i \in \{1, 2\}$. $\mathfrak{C}$ runs KeyGen $(params, ID, MSK_i)$ to generate a secret key SK_{ID} for ID. $\mathfrak{C}$ responds $\mathfrak{A}$ with SK_{ID}. This query can be made multiple times.

2. Permission Query. $\mathfrak{A}$ can query permission on (ID, ID') where the restrict is $ID \neq ID^*$ and $ID' \neq ID^*$. $\mathfrak{C}$ runs KeyGen$(params, ID, MSK_i)$ to extract a secret key SK_{ID}, then runs Permission $(params, ID', SK_{ID})$ to obtain AK. $\mathfrak{C}$ responds $\mathfrak{A}$ with AK. This query can be made multiple times.

Challenge. $\mathfrak{A}$ submits two messages M_1 and M_2 with equal length. $\mathcal{C}$ flips an unbiased coin with $\{0, 1\}$ and obtains $b \in \{0, 1\}$. He computes $CT^* =$ Encryption$(params, ID^*, M_b)$ and sends CT^* to $\mathfrak{A}$.

Phase 2. $\mathfrak{A}$ can adaptively make the following additional queries:

1. Secret key Query. $\mathfrak{A}$ can query secret key for an identity ID, where $ID \neq ID^*$. $\mathfrak{C}$ responds as in Phase 1.

2. Permission Query. $\mathfrak{A}$ can query permission on (ID, ID') where the restriction is $ID \neq ID^*$ and $ID' \neq ID^*$. $\mathfrak{C}$ responds as in Phase 1.

Guess. $\mathfrak{A}$ outputs his guess b' on b. $\mathfrak{A}$ wins the game if $b' = b$.

Definition 2. *An identity-based data storage scheme supporting intra-domain and inter-domain queries is $(T, q_1, q_2, \epsilon(\ell))$-selective identity and chosen plaintext (IND-sID-CPA) secure if no probabilistic polynomial-time adversary $\mathfrak{A}$ making at most q_1 secrete key queries and q_2 permission queries can win the game with the advantage*

$$Adv_{\mathfrak{A}}^{IND\text{-}sID\text{-}CPA} = |\Pr[b' = b] - \frac{1}{2}| \geq \epsilon(\ell)$$

in the above model.

2.3. Complexity Assumption

Let $\mathbb{G}$ and $\mathbb{G}_\tau$ be two multiplicative groups with prime order p and g be a generator of the group $\mathbb{G}$. A bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_\tau$ is a map between the groups $\mathbb{G}$ and $\mathbb{G}_\tau$ with the following properties.

1. Bilinearity. For all $x, y \in \mathbb{Z}_p$ and $u, v \in \mathbb{G}$, $e(u^x, v^y) = e(x, y)^{xy}$.
2. Non-degeneracy. $e(g, g) \neq 1$ where 1 is the identity element of the group $\mathbb{G}_\tau$.
3. Computability. For all $u, v \in \mathbb{G}$, there exists an efficient algorithm to compute $e(u, v)$.

Let $\mathcal{GG}(1^\ell)$ be a generator of bilinear group, which takes as input a security parameter 1^ℓ and outputs a bilinear group $(e, p, \mathbb{G}, \mathbb{G}_\tau)$ with prime order p.

Definition 3. (Decisional Bilinear Diffie-Hellman (DBDH) Assumption[24].) *Let* $a, b, c, z \xleftarrow{R} \mathbb{Z}_p$, $\mathcal{GG}(1^\ell) \to (e, p, \mathbb{G}, \mathbb{G}_\tau)$ *and* g *be a generator of the group* $\mathbb{G}$. *We say that the DBDH assumption holds in* $(e, p, \mathbb{G}, \mathbb{G}_\tau)$ *if no probabilistic polynomial-time adversary* $\mathfrak{A}$ *can distinguish* $(A, B, C, Z) = (g^a, g^b, g^c, e(g, g)^z)$ *from* $(A, B, C, Z) = (g^a, g^b, g^c, e(g, g)^{abc})$ *with the advantage*

$$Adv_{\mathfrak{A}}^{DBDH} = |\Pr[\mathfrak{A}(A, B, C, e(g, g)^z) = 1] - \Pr[\mathfrak{A}(A, B, C, e(g, g)^{abc} = 1]| \geq \epsilon(\ell)$$

where the probability is token over the random choices of a, d, c, z *and the bits consumed by* $\mathfrak{A}$.

2.4. Inter-domain identity-based proxy re-encryption in[33]

In this section, we demonstrate a collusion attack against Tang, Hartel and Jonker's scheme[33]. For completeness, we will first describe Tang, Hartel and Jonker's scheme and subsequently, the collusion attack against it.

Tang, Hartel and Jonker's scheme[33] works as follows:

$\mathsf{Setup}_1(1^\ell)$. This algorithm takes as input a security parameter 1^ℓ and outputs a bilinear group $(e, p, \mathbb{G}, \mathbb{G}_\tau)$ and hash functions $H_1 : \{0,1\}^* \to \mathbb{G}$ and $H_2 : \{0,1\}^* \to \mathbb{G}$. Let g be a generator of $\mathbb{G}$. PKG_1 generates his master secret $\alpha_1 \xleftarrow{R} \mathbb{Z}_p$ and public key $y_1 = g^{\alpha_1}$. The public parameters are $(e, p, \mathbb{G}, \mathbb{G}_\tau, g, y_1, H_1, H_2)$ and the master secret key is α_1.

$\mathsf{KeyGen}_1(\alpha_1, ID)$. This algorithm takes as input the master secret key α_1 and an identity ID, and computes $SK_{ID} = H_1(ID)^{\alpha_1}$. The secret key for the identity ID is SK_{ID}.

$\mathsf{Encryption}_1(M, ID)$. To encrypt a message M, this algorithm selects $s \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and computes

$$C_0 = M \cdot e(y_1, H_1(ID))^s \quad \text{and} \quad C_1 = g^s.$$

The ciphertext is $CT = (C_0, C_1)$.

$\mathsf{Decryption}_1(CT, SK_{ID})$. This algorithm takes as input the secret key of original decrypter SK_{ID} and the ciphertext CT, and computes

$$M = \frac{C_0}{e(C_1, SK_{ID})}.$$

Suppose that PKG_2 generates master secret and pubic key pair (α_2, y_2) and setups another IBE scheme with algorithms $(\mathsf{Setup}_2, \mathsf{KeyGen}_2, \mathsf{Encryption}_2, \mathsf{Decryption}_2)$, where $y_2 = g^{\alpha_2}$. Suppose that the designated decrypter with identity ID' is registered with PKG_2 and obtains secret key $SK_{ID'}$.

$\mathsf{RKeyGen}(SK_{ID}, ID')$. To generate a re-encryption key for ID', this algorithm selects $X \stackrel{R}{\leftarrow} \{0,1\}^*$, and computes

$$R_1 = H_1(ID)^{-\alpha_1} \cdot H_2(X) \text{ and } R_2 = \mathsf{Encryption}_2(X, ID').$$

The re-encryption key is (R_1, R_2).

$\mathsf{Re\text{-}encryption}(CT, R_1, R_2, ID')$. To encrypt the ciphertext CT, this algorithm takes as input the ciphertext CT, the re-encryption key (R_1, R_2) and the identity ID' and computes

$$C_1' = \mathsf{Encryption}_2(\mathsf{C}_1, \mathsf{ID}'), \; C_2' = R_2 \quad \text{and} \quad C_3' = C_0 \cdot e(C_1, R_1 \cdot H_2(C_1)).$$

The re-encrypted ciphertext is $CT' = (C_1', C_2', C_3')$.

$\mathsf{Decryption}_3(CT', SK_{ID'})$. The designated decrypter uses his secret key $SK_{ID'}$ to compute $W_1 = \mathsf{Decryption}_2(\mathsf{C}'_1, \mathsf{SK}_{\mathsf{ID}'})$, $W_2 = \mathsf{Decryption}_2(C'_2, SK_{ID'})$ and

$$M = \frac{C'_3}{e(W_1, H_2(W_2) \cdot H_2(W_1))}$$

Collusion Attacks. If the designated decrypter can compromise the proxy server, he can obtain the re-encryption key (R_1, R_2). Then, he can use his secret key $SK_{ID'}$ to compute $X = \mathsf{Decryption}_2(\mathsf{R}_2, SK_{ID'})$. Therefore, he can compute the secret key of the original decrypter $SK_{ID} = \frac{H_2(X)}{R_1}$. □

3. Identity-Based Data Storage Scheme in Cloud Computing

In this section, we propose an identity-based data storage scheme supporting intra-domain and inter-domain queries and prove its security. In our scheme, the access permission can be determined by the data owner independently without the need of the PKG. Especially, the access permission is bound to the requested ciphertext. Furthermore, our scheme is secure against the collusion attacks.

Overview. Our aim is to design an identity-based data storage scheme where multiple domains can co-exist and users can share files across domains. For simplicity, suppose that there are two domains: $\mathfrak{D}_1$ and $\mathfrak{D}_2$. At first, the private key generator PKG_i in the domain $\mathfrak{D}_i$ generates his secret-public pair $(\xi_i, (g_i, h_i))$ where $i \in \{1, 2\}$. Then, users in the domain $\mathfrak{D}_i$ authenticate themselves to the PKG_i and obtain secret keys from PKG_i. Prior to outsourcing the data, the data owner encrypts it under his identity ID. Then, the owner sends the ciphertext to the proxy server PS. PS validates the ciphertext. If it is computed correctly, PS stores it for the owner; otherwise, he rejects the ciphertext. Suppose that the PS can detect which domain the requester is from and the owner can know which file the requester wants to access from the partial ciphertext. If the requester wants to access a file, he can use his secret key $K_{ID'}$ to compute an authentication information $(Q, F, K_{ID',3})$ and sends it to the PS. If the requester and the owner are in the same domain, the PS sends $(ID', Q, F, K_{ID',3}, C_2)$ to the owner, where C_2 is the partial ciphertext. If the requester and the owner are in different domains, the PS sends $(ID', Q, F, K_{ID',3}, (g_i, h_i))$ to the owner. To resist the illegal requesters outside the cloud access the file, the owner validates the

requester by verifying $(Q, F, K_{ID',3})$. If the authentication is successful, the owner creates an access key $(P_1, P_2, P_3, K_{ID,2})$ and sends it to the PS. PS re-encrypts the ciphertext using the access key and sends the re-encrypted ciphertext to the requester. At the end, the requester can use his secret key to decrypt the re-encrypted ciphertext.

In the inter-domain query, suppose that the owner is in the domain $\mathcal{D}_i$ and the requester is in the domain $\mathcal{D}_{3-i}$, where $i \in \{1, 2\}$. Actually, in our scheme, the owner in $\mathcal{D}_i$ can use his secret key to generate an access key[1] for the requester in $\mathfrak{D}_{3-i}$. Furthermore, the proxy server PS_i can use the access key to transfer a ciphertext for the owner to a ciphertext for the requester.

Our scheme is based on the IBE scheme[35]. The protocol is described in Figure 3.

Correctness. The following equations hold.

$$\begin{aligned} C_1 \cdot \frac{e(K_{ID,2}, C_3)}{e(K_{ID,1}, C_2)} &= M \cdot e(g_i, \eta)^s \frac{e(g^{r_{ID}}, (g_i^{ID} h)^s)}{e(\eta^{\alpha_i} (g_i^{ID} h)^{r_{ID}}, g^s)} \\ &= M \cdot e(g_i, \eta)^s \frac{e(g^{r_{ID}}, (g_i^{ID} h)^s)}{e(g_i, \eta)^s \cdot e(g^{r_{ID}}, (g_i^{ID} h)^s)} \\ &= M \cdot e(g_i, \eta)^s \cdot \frac{1}{e(g_i, \eta)^s} \\ &= M, \end{aligned}$$

$$\begin{aligned} P_1 &= \frac{K_{ID,1}}{Q \cdot F^{\nu}} \cdot g^{ID'\beta} \\ &= \frac{\eta^{\alpha_i} (g_i^{ID} h)^{r_{ID}}}{K_{ID',1} \mathfrak{h}^k \cdot \mathfrak{g}^{k\nu}} \cdot g^{ID'\beta}, \end{aligned}$$

$$P_3 = e(C_2, g)^{ID'\beta} = e(g, g)^{s\beta ID'},$$

$$C_1' = P_3 \cdot C_1 = M \cdot e(g_i, \eta)^s \cdot e(g, g)^{s\beta ID'},$$

[1]This key maybe not identical to that generated by the PKG_i for the requester with identity ID'. Here, we just mean that the requester can use it to decrypt the re-encrypted ciphertext.

Setup. This algorithm takes as input a security parameter 1^ℓ, and outputs a bilinear group $\mathcal{GG}(1^\ell) \rightarrow (e, p, \mathbb{G}, \mathbb{G}_\tau)$ with prime order p, where $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_\tau$. Let $g, h, \eta, \mathfrak{g}, \mathfrak{h}$ be the generators of $\mathbb{G}$.

1. PKG_1 chooses $\alpha_1 \overset{R}{\leftarrow} \mathbb{Z}_p$ and sets $g_1 = g^{\alpha_1}$, $h_1 = \mathfrak{g}^{\alpha_1}$ and $\xi_1 = \eta^{\alpha_1}$. The master secret key is ξ_1 and the public key is $(g, h, \eta, \mathfrak{g}, \mathfrak{h}, g_1, h_1)$.

2. PKG_2 chooses $\alpha_2 \overset{R}{\leftarrow} \mathbb{Z}_p$ and sets $g_2 = g^{\alpha_2}$, $h_2 = \mathfrak{g}^{\alpha_2}$ and $\xi_2 = \eta^{\alpha_2}$. The master secret key is ξ_2 and the public key is $(g, h, \eta, \mathfrak{g}, \mathfrak{h}, g_2, h_2)$.

KeyGen. This algorithm takes as input the master secret key ξ_i of PKG_i, and an identity $ID \in \mathbb{Z}_p$ in the domain $\mathfrak{D}_i$, and computes

$$K_{ID,1} = \eta^{\alpha_i}(g_i^{ID}h)^{r_{ID}},\ K_{ID,2} = g^{r_{ID}} \text{ and } K_{ID,3} = \mathfrak{g}^{r_{ID}}$$

where $r_{ID} \overset{R}{\leftarrow} \mathbb{Z}_p$ and $i \in \{1, 2\}$.

The secret key for the user U with identity ID is $K_{ID} = (K_{ID,1}, K_{ID,2}, K_{ID,3})$. This secret key can be verified by

$$e(K_{ID,1}, g) \overset{?}{=} e(\eta, g_i) \cdot e(g_i^{ID}h, K_{ID,2}) \text{ and } e(K_{ID,2}, \mathfrak{g}) \overset{?}{=} e(g, K_{ID,3}).$$

Encryption. Suppose that the owner O with identity ID in the domain $\mathfrak{D}_i$ obtains secret key $SK_{ID} = \{K_{ID,1}, K_{ID,2}, K_{ID,3}\}$. To encrypt a message M, he chooses $s \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes

$$C_1 = M \cdot e(g_i, \eta)^s,\ C_2 = g^s \text{ and } C_3 = (g_i^{ID}h)^s$$

where $i \in \{1, 2\}$. The ciphertext for the message M is $CT = (C_1, C_2, C_3)$. The owner sends CT to the proxy server PS_i in the domain $\mathfrak{D}_i$. PS_i validates the ciphertext by checking

$$e((g_i^{ID}h), C_2) \overset{?}{=} e(C_3, g)$$

where $i \in \{1, 2\}$. If the equation holds, PS_i stores the ciphertext $CT = (C_1, C_2, C_3)$ for the owner. Otherwise, he rejects the ciphertext.

Query. A requester R with identity ID' queries the data outsourced by the owner with identity ID. Suppose that the requester with identity ID' has secret key $SK_{ID'} = \{K_{ID',1}, K_{ID',2}, K_{ID',3}\}$. He chooses $k \overset{R}{\leftarrow} \mathbb{Z}_p$, and computes $Q = K_{ID',1}\mathfrak{h}^k$ and $F = \mathfrak{g}^k$. He sends $(Q, F, K_{ID',3})$ to the proxy server PS_i. There are two scenarios:

1. Both the owner and the requester are in the same domain $\mathfrak{D}_i$. The PS_i sends $(ID', Q, F, K_{ID',3}, C_2)$ to the owner.
2. The owner and the requester are in different domains. Suppose that the owner is in $\mathfrak{D}_i$ and the requester is in $\mathfrak{D}_{3-i}$ where $i \in \{1, 2\}$. The PS_i sends $(ID', Q, F, K_{ID',3}, (g_{3-i}, h_{3-i}), C_2)$ to the owner.

Permission. There are two scenarios:

1. Both the owner and the requester are in the same domain $\mathfrak{D}_i$. The owner checks

$$e(Q, \mathfrak{g}) \overset{?}{=} e(\eta, h_i) \cdot e(g_i^{ID'} h, K_{ID',3}) \cdot e(\mathfrak{h}, F).$$

2. The owner and the requester are in different domains. The owner checks

$$e(Q, \mathfrak{g}) \overset{?}{=} e(\eta, h_{3-i}) \cdot e(g_{3-i}^{ID'} h, K_{ID',3}) \cdot e(\mathfrak{h}, F).$$

If it holds, the owner chooses $\beta, \nu \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes

$$P_1 = \frac{K_{ID,1}}{Q \cdot F^{\nu}} \cdot g^{ID'\beta}, \ P_2 = \mathfrak{g}^{\nu} \ \text{ and } P_3 = e(C_2, g)^{ID'\beta}.$$

The owner sends the access key $(P_1, P_2, P_3, K_{ID,2})$ to PS_i.

Re-encryption. Receiving $(P_1, P_2, P_3, K_{ID,2})$ from the owner, the proxy server PS_i computes the re-encrypted ciphterxt as

$C'_1 = P_3 \cdot C_1,\ C'_2 = C_2,\ C'_3 = C_3,\ C'_4 = P_1,\ C'_5 = P_2$ and $C'_6 = K_{ID,2}$.

The proxy server responds the requester with $CT' = (C'_1, C'_2, C'_3, C'_4, C'_5, C'_6)$.

Decryption.

1. The owner can compute $M = C_1 \cdot \frac{e(K_{ID,2}, C_3)}{e(K_{ID,1}, C_2)}$.
2. The requester computes $E = K_{ID',1} \cdot C'_4 \cdot \mathfrak{h}^k \cdot C'^k_5$. Then, he can compute $M = C'_1 \cdot \frac{e(C'_6, C'_3)}{e(E, C'_2)}$.

Figure 3: Identity-based Data Storage Scheme Supporting Intra-Domain and Inter-Domain Queries

and

$$\begin{aligned} E &= K_{ID',1} \cdot C'_4 \cdot \mathfrak{h}^k \cdot C'^k_5 \\ &= K_{ID',1} \cdot \frac{K_{ID,1}}{K_{ID',1}\mathfrak{h}^k \cdot \mathfrak{g}^{k\nu}} \cdot g^{ID'\beta} \cdot \mathfrak{h}^k \cdot \mathfrak{g}^{k\nu} \\ &= K_{ID,1} \cdot g^{ID'\beta} \\ &= \eta^{\alpha_i} \cdot (g_i^{ID} h)^{r_{ID}} \cdot g^{ID'\beta}. \end{aligned}$$

Therefore, we have

$$\begin{aligned} & C'_1 \cdot \frac{e(C'_6, C'_3)}{e(E, C_2)'} \\ = \ & C'_1 \cdot \frac{e(g^{r_{ID}}, (g_i^{ID} h)^s)}{e(\eta^{\alpha_i}(g_i^{ID} h)^{r_{ID}} g^{ID'\beta}, g^s)} \\ = \ & C'_1 \cdot \frac{e(g, g_i^{ID} h)^{sr_{ID}}}{e(g_i, \eta)^s \cdot e(g, g_i^{ID} h)^{sr_{ID}} \cdot e(g, g)^{s\beta ID'}} \\ = \ & C'_1 \cdot \frac{1}{e(g_i, \eta)^s \cdot e(g, g)^{s\beta ID'}} \\ = \ & M \cdot e(g_i, \eta)^s \cdot e(g, g)^{s\beta ID'} \cdot \frac{1}{e(g_i, \eta)^s \cdot e(g, g)^{s\beta ID'}} \\ = \ & M. \end{aligned}$$

Theorem 1. *Our identity-based data storage scheme supporting intra-domain and inter-domain queries is $(T, q_1, q_2, \epsilon(\ell))$-selective identity and chosen plaintext (IND-sID-CPA) secure if the $(T', \epsilon(\ell)')$ decisional bilinear Diffie-Hellman (DBDH) assumption holds in the bilinear group $(e, p, \mathbb{G}, \mathbb{G}_\tau)$ where*

$$T' = T + \mathcal{O}(T) \quad \textit{and} \quad \epsilon(\ell)' = \frac{1}{2}\epsilon(\ell).$$

PROOF. Suppose that there exists an adversary $\mathfrak{A}$ who can $(T, q_1, q_2, \epsilon(\ell))$ break our scheme, we can construct an algorithm $\mathfrak{B}$ that can use $\mathfrak{A}$ to break the DBDH assumption as follows. The challenger $\mathfrak{C}$ generates the bilinear group $\mathcal{GG}(1^\ell) \rightarrow (e, p, \mathbb{G}, \mathbb{G}_\tau)$. Let g be a generator of the group $\mathbb{G}$. It flips an unbiased coin μ with $\{0, 1\}$. If $\mu = 0$, he sends $(A, B, C, Z) = (g^a, g^b, b^c, e(g, g)^{abc})$ to $\mathfrak{B}$. Otherwise, he sends $(A, B, C, Z) = (g^a, g^b, g^c, e(g, g)^z)$ to $\mathfrak{B}$, where $z \overset{R}{\leftarrow} \mathbb{Z}_p$. $\mathfrak{B}$ will outputs his guess μ' on μ.

Initialization. $\mathfrak{A}$ submits an identity ID^* with which he wants to be challenged to $\mathfrak{B}$. Let ID^* be in the domain $\mathfrak{D}_i$ where $i \in \{1, 2\}$.

Setup. $\mathfrak{B}$ selects $\upsilon, \gamma, \theta \overset{R}{\leftarrow} \mathbb{Z}_p$ and sets $g_i = A$, $g_{3-i} = g^\upsilon$, $\eta = B$, $\mathfrak{g} = g^\theta$, $h_i = A^\theta$, $h_{3-i} = g^{\upsilon\theta}$ and $h = g_i^{-ID^*} g^\gamma$. He chooses $\mathfrak{h} \overset{R}{\leftarrow} \mathbb{G}$. The public parameters are $(g, h, \eta, \mathfrak{g}, \mathfrak{h})$. The public keys for the PKG_i in $\mathfrak{D}_i$ and PKG_{3-i} in $\mathfrak{D}_{3-i}$ are (g_i, h_i) and (g_{3-i}, h_{3-i}), respectively. $\mathfrak{B}$ sends $\{(e, p, \mathbb{G}, \mathbb{G}_\tau), g, h, \eta, \mathfrak{g}, \mathfrak{h}, g_i, h_i, g_{3-i}, h_{3-i}\}$ to $\mathfrak{A}$. The master secret keys for PKG_i and PKG_{3-i} are g^{ab} and $g^{\upsilon b}$, respectively.

Phase 1.

1. Secret Key Query. $\mathfrak{A}$ queries secret key for an identity ID, where the only restricts is $ID \neq ID^*$.
   (a) If ID is in $\mathfrak{D}_{3-i}$, $\mathfrak{B}$ chooses $r \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes
   $$K_{ID,1} = B^\upsilon (g_{3-i}^{ID} h)^r, \; K_{ID,2} = g^r \text{ and } K_{ID,3} = K_{ID,2}^\theta;$$
   (b) If ID is in $\mathfrak{D}_i$, $\mathfrak{B}$ chooses $r \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes
   $$K_{ID,1} = B^{\frac{-\gamma}{ID - ID^*}} (g_i^{ID} h)^r, K_{ID,2} = g^r B^{\frac{-1}{ID - ID^*}}$$
   and
   $$K_{ID,3} = K_{ID,2}^\theta.$$

We claim that the secret key is computed correctly. We have

$$\begin{aligned} K_{ID,1} &= B^{\frac{-\gamma}{ID-ID^*}}(g_i^{ID}h)^r \\ &= g^{\frac{-b\gamma}{ID-ID^*}}(g^{a(ID-ID^*)+\gamma})^r \\ &= g^{ab}g^{-ab}g^{\frac{-b\gamma}{ID-ID^*}}(g^{a(ID-ID^*)+\gamma})^r \\ &= g^{ab}(g^{a(ID-ID^*)+\gamma})^{\frac{-b}{ID-ID^*}}(g^{a(ID-ID^*)+\gamma})^r \\ &= g^{ab}(g^{a(ID-ID^*)+\gamma})^{r-\frac{b}{ID-ID^*}} \\ &= g^{ab}(g_i^{ID}h)^{r-\frac{b}{ID-ID^*}}. \end{aligned}$$

Let $\hat{r} = r - \frac{b}{ID-ID^*}$, we have $K_{ID,1} = g^{ab}(g_i^{ID}h)^{\hat{r}}$, $K_{ID,2} = g^r B^{\frac{-1}{ID-ID^*}}$ $= g^{r-\frac{b}{ID-ID^*}} = g^{\hat{r}}$ and $K_{ID,3} = K_{ID,2}^{\theta} = g^{\hat{r}\theta} = \mathfrak{g}^{\hat{r}}$. So, the distribution of the secret key created in the simulation paradigm is identical to that in the real protocol.

2. Permission Query. $\mathfrak{A}$ queries permission on (ID, ID', C_2), where the only restrict is $ID \neq ID^*$ and $ID' \neq ID^*$.
   (a) If ID' is in $\mathfrak{D}_{3-i}$, $\mathfrak{B}$ chooses $r' \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes

$$K_{ID',1} = B^{\upsilon}(g_{3-i}^{ID}h)^{r'}.$$

   (b) If ID' is in $\mathfrak{D}_i$, $\mathfrak{B}$ chooses $r' \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes

$$K_{ID',1} = B^{\frac{-\gamma}{ID'-ID^*}}(g_i^{ID'}h)^{r'}.$$

   $\mathfrak{B}$ chooses $t, k, \beta, \nu \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes $Q = K_{ID',1}\mathfrak{h}^t$, $F = \mathfrak{g}^k$,

$$P_1 = \frac{K_{ID,1}}{Q \cdot F^{\nu}} \cdot g^{ID'\beta},\ P_2 = \mathfrak{g}^{\nu} \text{ and } P_3 = e(C_2, g)^{ID'\beta}.$$

   $\mathfrak{B}$ responds with $(P_1, P_2, P_3, K_{ID,2})$.

Challenge. $\mathfrak{A}$ submits two messages M_0 and M_1 with the equal length. $\mathfrak{B}$ flips an unbiased coin with $\{0, 1\}$ and obtains $\omega \in \{0, 1\}$. $\mathfrak{B}$ computes

$$C_1^* = M_{\omega} \cdot Z,\ C_2^* = C \text{ and } C_3^* = C^{\gamma}.$$

$\mathfrak{B}$ responds $\mathfrak{A}$ with the ciphertext $CT^* = (C_1^*, C_2^*, C_3^*)$.

Phase 2. Phase 1. is repeated.

Guess. The adversary $\mathfrak{A}$ outputs his guess ω' on ω. If $\omega' = \omega$, $\mathfrak{B}$ outputs $\mu' = 0$. If $\omega' \neq \omega$, $\mathfrak{B}$ outputs $\mu' = 1$.

As shown above, the public parameters, public keys and the secret keys created in the simulation paradigm are identical to those created in the real protocol. Thereafter, we can compute the advantage with which $\mathcal{B}$ can break the DBDH assumption as follows.

If $\mu = 0$, $CT^* = (C_1^*, C_2^*, C_3^*)$ is a legal ciphertext of the message M_ω. Hence, $\mathcal{A}$ can output $\omega' = \omega$ with advantage at least $\epsilon(\ell)$, namely $\Pr[\omega' = \omega|\mu = 0] \geq \frac{1}{2} + \epsilon(\ell)$. Since $\mathcal{B}$ outputs $\mu' = 0$ when $\omega' = \omega$, we have $\Pr[\mu' = \mu|\mu = 0] \geq \frac{1}{2} + \epsilon(\ell)$.

In the case $\mu = 1$, $CT^* = (C_1^*, C_2^*, C_3^*)$ is not a legal ciphertext of the message M_ω. Hence, $\mathcal{A}$ can output $\omega' \neq \omega$ without any advantage, namely $\Pr[\omega' \neq \omega|\mu' = 1] = \frac{1}{2}$. Since $\mathcal{B}$ outputs $\mu' = 1$ when $\omega' \neq \omega$, we have $\Pr[\mu' = \mu|\mu = 1] = \frac{1}{2}$.

Therefore, the advantage with which $\mathcal{B}$ can break the DBDH assumption is

$$|\frac{1}{2}\Pr[\mu' = \mu|\mu = 0] + \frac{1}{2}\Pr[\mu' = \mu|\mu = 1] - \frac{1}{2}| \geq \frac{1}{2} \times (\frac{1}{2} + \epsilon(\ell)) + \frac{1}{2} \times \frac{1}{2} - \frac{1}{2} \geq \frac{1}{2}\epsilon(\ell).$$

□

Collusion Attacks. In our scheme, when computing an access permission, the data owner chooses a random number $\beta \overset{R}{\leftarrow} \mathbb{Z}_p$, randomizes his secret key $K_{ID,1}$ by g^β and computes $P_2 = \mathfrak{g}^\nu$ and $P_3 = e(C_2, g)^{ID'\beta}$. If the requester can compromise the proxy server, they can obtain $V = K_{ID,1} \cdot g^{ID'\beta}$ by $K_{ID',1}$ and $P_1 = \frac{K_{ID,1}}{Q \cdot F^\nu} \cdot g^{ID'\beta}$. If he can compute $K_{ID,1}$ from V, he can compute $\Psi = g^\beta = (\frac{V}{K_{ID,1}})^{\frac{1}{ID'}}$. However, this is intractable since the random number β is unknown to the adversary $\mathfrak{A}$.

Chosen Ciphertext Security. We can employ the technique introduced in[36] to our scheme to obtain a CCA secure identity-based data storage scheme supporting intra-domain and inter-domain queries.

Computation and Communication Costs.

Table 1: The Computation Cost in Our Scheme

Scheme	Setup	KeyGen	Encryption	Query	Permission	Decryption	
						O	R
Our scheme	6E	5E+5P	4E+3P	2E	5E+5P	2P	2P+2E

Table 2: The Communication Cost in Our Scheme

Scheme	Setup	KeyGen	Encryption	Query			Permission	Re-encryption
					Intra	Inter		
		$PKG \rightarrow U$	$O \rightarrow PS$	$R \rightarrow PS$	$PS \rightarrow O$	$PS \rightarrow O$	$O \rightarrow PS$	$PS \rightarrow R$
Our Scheme	$9E_{\mathbb{G}}$	$3E_{\mathbb{G}}$	$2E_{\mathbb{G}} + E_{\mathbb{G}_\tau}$	$3E_{\mathbb{G}}$	$3E_{\mathbb{G}} + E_{\mathbb{Z}_p}$	$5E_{\mathbb{G}} + E_{\mathbb{Z}_p}$	$3E_{\mathbb{G}} + E_{\mathbb{G}_\tau}$	$5E_{\mathbb{G}} + E_{\mathbb{G}_\tau}$

We demonstrate the computation cost and communication cost in our scheme in Table 1 and Table 2, respectively. By E and P, we denote one exponential and one paring computation. By $E_{\mathbb{G}}$ and $E_{\mathbb{G}_\tau}$, we denote one element in the group $\mathbb{G}$ and $\mathbb{G}_\tau$. We denote PKG, U, O, PS and R as the private key generator, the user, the data owner, the proxy server and the requester, respectively.

4. Performance Evaluation

The efficiency of pairing-based schemes is related to the selected elliptic curve. Literatures[37,38,39] suggested that how to select elliptic curves for efficient cryptographic systems. In order to select a secure elliptic curve, two important factors must be considered: the group size l of the elliptic curve and the embedding degree d. To achieve the security of 1,024-bit RSA, the group size and the embedding degree should satisfy $l \times d \geq 1024$. Although the pairing operations on the elliptic curve with high embedding degree is expensive, the length of the elements in this curve is short. Most of pairing-based schemes are implemented on the elliptic curves: Type A and Type D[39]. Type A is a supersingular curve $y^2 = x^3 + \lambda x$, where $\mathbb{G}_1 = \mathbb{G}_2$ and the group order is a Solinas prime. On a Type A curve, the pairing operation is fastest, but the length of the elements in this curve is longer. Meanwhile, a Type D curve is an MNT curve $y^2 = x^3 + \lambda_1 x + \lambda_0$, where $\mathbb{G}_1 \neq \mathbb{G}_2$. On a Type D curve, the length of the elements in this curve can be shorter, but the pairing operation is more expensive.

4.1. Benchmark Time

The running time of different operations on the bilinear group is obtained on a DELL E630 with Intel(R) Core™ 2 Duo CPU (T8100@2.10GHz) and 2GB RAM running Ubuntu 9.10. The running time is calculated by computing the average of running the operation 10 times with random inputs

Table 3: Benchmark Time of Different Operations on Type A Curve (ms)

Curve	Pairing		$\mathbb{G}$		$\mathbb{G}_\tau$	
	Normal	PP	EP	MBE	EP_τ	MBE_τ
Type A	5.883	2.565	0.736	6.278	0.142	1.242

Table 4: Running Time of Our scheme on Type A Curve (ms)

Curve	Setup	KeyGen	Encryption	Query	Permission	Decryption	
						Owner	Requester
Type A	4.416	16.505	35.509	1.4720	16.505	5.130	6.602

using the text code from the PBC library[39]. The running times of different operations on the bilinear group from Type A curve are described in Table 3. By PP, EP, EP_τ, we denote the running time of a pairing operation with preprocessing, an exponential operation on group $\mathbb{G}$ with preprocessing and an exponential operation on group $\mathbb{G}_\tau$ with preprocessing, respectively. By MBE and MBE_τ, we denote the running time of executing exponential operations on multiple bases on the group $\mathbb{G}$ and $\mathbb{G}_\tau$, respectively, such as $\zeta = g^{x_1}h^{x_2}$.

4.2. Implementations of Our Scheme

We implement our scheme on Type A curve: $y^2 = x^3 + x$, where $\mathbb{G}_1 = \mathbb{G}_2$, the order p is 160 bits, the embedding degree $d = 2$ and the group size l is 512 bits. Compared with the curves with $d > 2$, the expensive paring operation is fastest on Type A curve.

We describe the running times consumed by different algorithms in our scheme in Table 4. We observe that it takes 4.416 ms and 16.505 ms to setup the system and generate a secret key, respectively. To encrypt a message, it takes 35.509 ms. It takes 1.4720 ms and 16.505 ms to compute a query information and an access permission, respectively. To decrypt a ciphertext for the data owner and the requester, it takes 5.130 ms and 6.602 ms, respectively.

The communication cost of our scheme on Type A curve is described in Table 5.

Table 5: The Communication Cost of Our Scheme in Type A Curve (byte)

Curve	Setup	KeyGen	Encryption	Query			Permission	Re-encryption
					Intra	Inter		
		$PKG \to U$	$O \to PS$	$R \to PS$	$PS \to O$	$PS \to O$	$O \to PS$	$PS \to R$
Type A	576	192	192	192	212	340	256	384

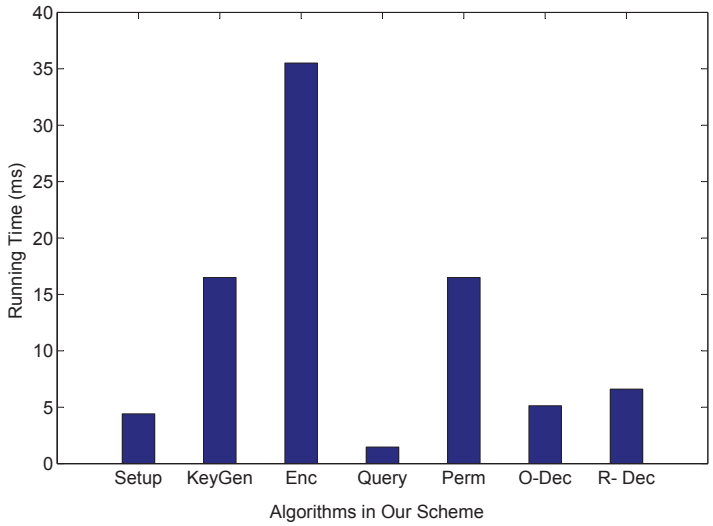

Figure 4: Comparison of The Running Time Consumed by the Algorithms: Setup, KeyGen, Encryption, Query, Permission, Owner Decryption and Requester Decryption

5. Conclusion

Cloud computing is a distributed system where users in different domains can share data among each other. Identity-based proxy re-encryption schemes have been proposed to outsource sensitive data from the owner to an external party. Nevertheless, they cannot be employed in cloud computing. For example, they can only support the intra-domain query and the access key is computed with the help of the private key generator (PKG). Additionally, the proxy server must be trusted. In this paper, we proposed an identity-based data storage scheme which is suitable to the cloud computing scenario as it supports both intra-domain and inter-domain queries. In our scheme, the access key is bound to not only the requester's identity but also the requested ciphertext, and can be computed by the owner independently without the help of the PKG. For one query, the requester can only access one file of the owner, instead of all files. Furthermore, our scheme is secure against the collusion attacks. We proved the security of the proposed scheme in the selective-identity model.

Acknowledge

The first author was supported by PhD scholarships of Smart Services Cooperative Research Centre (CRC) and University of Wollongong. The second author was supported by ARC Future Fellowship FT0991397.

References

[1] Bouganim L, Pucheral P. Chip-secured data access: Confidential data on untrusted servers. In: Proceedings: International Conference on Very Large Data Bases - VLDB 2002. Hong Kong, China: Morgan Kaufmann; 2002: 131-142.

[2] Maheshwari U, Vingralek R, ShapiroW. How to build a trusted database system on untrusted storage. In: Proceedings: Symposium on Operating System Design and Implementation - OSDI 2000. San Diego, California, USA: USENIX; 2000: 135-150.

[3] Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, Zaharia M. A view of cloud computing. Communications of the ACM 2010; 53(4): 50-58.

[4] Fernando N, Loke SW, Rahayu W. Mobile cloud computing: A survey. Future Generation Computer Systems 2013; 29(1): 84-106.

[5] Aoun R, Abosi CE, Doumith EA, Nejabati R, Gagnaire M, Simeonidou D. Towards an optimized abstracted topology design in cloud environment. Future Generation Computer Systems 2013; 29(1): 46-60.

[6] Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation Computer Systems 2009; 25(6): 599-616.

[7] Abrishami S, Naghibzadeh M, Epema DH. Deadline-constrained workflow scheduling algorithms for infrastructure as a service clouds. Future Generation Computer Systems 2013; 29(1): 158-169.

[8] Anthes G. Security in the Cloud. Communications of the ACM 2010; 53(11): 16-18.

[9] Gu Y, Grossman RL. Sector: A high performance wide area community data storage and sharing system. Future Generation Computer Systems 2010; 26(5): 720-728.

[10] Spillner J, Muller J, Schill A. Creating optimal cloud storage systems. Future Generation Computer Systems. http://dx.doi.org/10.1016/j.future.2012.06.004.

[11] Samarati P, di Vimercati SDC. Data protection in outsourcing scenarios: issues and direction. In: Feng D, Basin DA, Liu P, eds. Proceedings: ACM Symposium on Information, Computer and Communications Security- ASIACCS 2010. Beijing, China: ACM; 2010: 1-14.

[12] Kher V, Kim Y. Securing distributed storage: challenges, techniques, and systems. In: Atluri V, Samarati P, Yurcik W, Brumbaugh L, Zhou Y, eds. Proceedings: ACM Workshop On Storage Security And Survivability- StorageSS 2005. Fairfax, VA, USA: ACM; 2005: 9-25.

[13] Mambo M, Okamoto E. Proxy cryptosystems: Delegation of the power to decyrpt ciphertexts. IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences 1997; E80A(1): 54-63.

[14] Blaze M, Bleumer G, Strauss M. Divertible protocols and atomic proxy cryptography. In: Nyberg K, ed. Proceedings: Advances in Cryptology - EUROCRYPT 1998; vol. 1403 of Lecture Notes in Computer Science. Espoo, Finland: Springer-Verlag; 1998: 127-144.

[15] Ateniese G, Benson K, Hohenberger S. Key-private proxy re-encryption. In: Fischlin M, ed. Proceedings: The Cryptographers Track at the RSA Conference - CT-RSA 2009; vol. 5743 of Lecture Notes in Computer Science. San Francisco, CA, USA: Springer-Verlag; 2009: 279-294.

[16] Matsuda T, Nishimaki R, Tanaka K. CCA proxy re-encryption without bilinear maps in the standard model. In: Nguyen PQ, Pointcheval D, eds. Proceedings: Public Key Cryptography - PKC 2010; vol. 6056 of Lecture Notes in Computer Science. Paris, France: Springer-Verlag; 2010: 261-278.

[17] Shao J, Cao Z. CCA-secure proxy re-encryption without pairings. In: Jarecki S, Tsudik G, eds. Proceedings: Public Key Cryptography - PKC

2009; vol. 5443 357376 of Lecture Notes in Computer Science. Irvine, CA, USA: Springer-Verlag; 2009: 357-376.

[18] Chow SSM, Weng J, Yang Y, Deng: RH. Efficient unidirectional proxy re-encryption. In: Bernstein DJ, Lange T, eds. Proceedings: Progress in Cryptology - AFRICACRYPT 2010; vol. 6055 of Lecture Notes in Computer Science. Stellenbosch, South Africa: Springer-Verlag; 2010: 316-332.

[19] Weng J, Zhao Y, Hanaoka G. On the security of a bidirectional proxy re-encryption scheme from PKC 2010. In: Catalano D, Fazio N, Gennaro R, Nicolosi A, eds. Proceedings: Public Key Cryptography - PKC 2011; vol. 6571 of Lecture Notes in Computer Science. Taormina, Italy: Springer-Verla; 2011: 284-295.

[20] Weng J, Deng RH, Ding X, Chu CK, Lai J. Conditional proxy re- encryption secure against chosen-ciphertext attack. In: Li W, Susilo W, Tupakula UK, Safavi-Naini R, Varadharajan V, eds. Proceedings: ACM Symposium on Information, Computer and Communications Security - ASIACCS 2009. Sydney, Australia: ACM; 2009: 322-332.

[21] Fang L, Susilo W, Ge C, Wang J. Hierarchical conditional proxy re-encryption. Computer Standards & Interfaces 2012; 34(4): 380-389.

[22] Ateniese G, Fu K, Green M, Hohenberger S. Improved proxy re- encryption schemes with applications to secure distributed storage. In: Proceedings: Network and Distributed System Security Symposium - NDSS 2005. San Diego, California, USA: The Internet Society; 2005: 1-15.

[23] Shamir A. Identity-based cryptosystems and signature scheme. In: Blak- ley GR, Chaum D, eds. Proceedings: Advances in Cryptology - CRYPTO 1984; vol. 196 of Lecture Notes in Computer Science. Santa Barbara, California, USA: Springer-Verlag; 1984: 47-53.

[24] Boneh D, Franklin M. Identity-based encryption from the weil pairing. In: Kilian J, ed. Proceedings: Advances in Cryptology - CRYPTO 2001; vol. 2139 of Lecture Notes in Computer Science. Santa Barbara, California, USA: Springer-Verlag; 2001: 213-229.

[25] Waters B. Efficient identity-based encryption without random oracles. In: Cramer R, ed. Proceedings: Advances in Cryptology -EUROCRYPT 2005; vol. 3494 of Lecture Notes in Computer Science. Aarhus, Denmark: Springer-Verlag; 2005: 114-127.

[26] Gentry C. Practical identity-based encryption without random ora- cles. In: Vaudenay S, ed. Proceedings: Advances in Cryptology - EUROCRYPT 2006; vol. 4004 of Lecture Notes in Computer Science. St. Petersburg, Russia: Springer-Verlag; 2006: 445-464.

[27] Ivan A, Dodis Y. Proxy cryptography revisited. In: Proceedings: Network and Distributed System Security Symposium - NDSS 2003. San Diego, California, USA: The Internet Society; 2003: 1-20.

[28] Green M, Ateniese G. Identity-based proxy re-encryption. In: Katz J, Yung M, eds. Proceedings: Applied Cryptography and Network Security - ACNS 2007; vol. 4521 of Lecture Notes in Computer Science. Zhuhai, China: Springer-Verlag; 2007: 288-306.

[29] Matsuo T. Proxy re-encryption systems for identity-based encryption. In: Takagi T, Okamoto T, Okamoto E, Okamoto T, eds. Proceedings: Pairing-Based Cryptography - Pairing 2007; vol. 4575 of Lecture Notes in Computer Science. Tokyo, Japan: Springer-Verlag; 2007: 247-267.

[30] Wang L, Wang L, Mambo M, Okamoto E. Identity-based proxy cryptosystems with revocability and hierarchical confidentialities. In: Soriano M, Qing S, Lopez J, eds. Proceedings: International Conference on Information and Communications Security - ICICS 2010; vol. 6476 of Lecture Notes in Computer Science. Barcelona, Spain: Springer-Verlag; 2010: 383-440.

[31] 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. Proceedings: Pairing-Based Cryptography - Pairing 2010 ; vol. 6487 of Lecture Notes in Computer Science. Yamanaka Hot Spring, Japan: Springer-Verlag; 2010: 327-346.

[32] Chu CK, Tzeng WG. Identity-based proxy re-encryption without random oracles. In: Garay JA, Lenstra AK, Mambo M, Peralta R, eds. Proceedings: Information Security Conference - ISC 2007; vol. 4779 of

Lecture Notes in Computer Science. Valparaso, Chile: Springer-Verlag; 2007: 189-202.

[33] Tang Q, Hartel PH, Jonker W. Inter-domain identity-based proxy re-encryption. In: Yung M, Liu P, Lin D, eds. Proceedings: Information Security and Cryptology - Inscrypt 2008; vol. 5487 of Lecture Notes in Computer Science. Beijing, China: Springer-Verlag; 2008: 332-347.

[34] Boneh D, Boyen X. Efficient selective-id secure identity-based encryption without random oracles. In: Cachin C, Camenisch J, eds. Proceedings: Advances in Cryptology - EUROCRYPT 2004; vol. 3027 of Lecture Notes in Computer Science. Interlaken, Switzerland: Springer-Verlag; 2004: 223-238.

[35] Boneh D, Boyen X, Halevi S. Chosen ciphertext secure public key threshold encryption without random oracles. In: Pointcheval D, ed. Proceedings: The Cryptographers Track at the RSA Conference - CT- RSA 2006; vol. 3860 of Lecture Notes in Computer Science. San Jose, CA, USA: Springer-Verlag; 2006: 226-243.

[36] Canetti R, Halevi S, Katz J. Chosen-ciphertext security from identity-based encryption. In: Cachin C, Camenisch J, eds. Proceedings: Advances in Cryptology - EUROCRYPT 2004; vol. 3027 of Lecture Notes in Computer Science. Interlaken, Switzerland: Springer-Verlag; 2004: 207-222.

[37] National Institute of Standards and Technology, Recomeeded Elliptic Curves for Federal Government Use. http://csrc.nist.gov/groups/ST/toolkit/documents/dss /NISTReCur.pdf July.

[38] Certicom Research, Standards for Efficient Cryptography SEC 2: Recommended El- liptic Curve Domain Parameters. http://www.secg.org/collateral/sec2 final.pdf September.

[39] PBC library, the pairing-based cryptography library. http://crypto.stanford.edu/pbc/.