Identity-Based Secure Distributed Data Storage Schemes

Jinguang Han, *Student Member, IEEE,* Willy Susilo, *Senior Member, IEEE,* and Yi Mu, *Senior Member, IEEE*

Abstract—Secure distributed data storage can shift the burden of maintaining a large number of files from the owner to proxy servers. Proxy servers can convert encrypted files for the owner to encrypted files for the receiver *without* the necessity of knowing the content of the original files. In practice, the original files will be removed by the owner for the sake of space efficiency. Hence, the issues on confidentiality and integrity of the outsourced data must be addressed carefully. In this paper, we propose two identity-based secure distributed data storage (IBSDDS) schemes. Our schemes can capture the following properties: (1) The file owner can decide the access permission independently without the help of the private key generator (PKG); (2) For one query, a receiver can only access one file, instead of all files of the owner; (3) Our schemes are secure against the collusion attacks, namely even if the receiver can compromise the proxy servers, he cannot obtain the owner's secret key. Although the first scheme is only secure against the chosen plaintext attacks (CPA), the second scheme is secure against the chosen ciphertext attacks (CCA). To the best of our knowledge, it is the *first* IBSDDS schemes where an access permissions is made by the owner for an exact file and collusion attacks can be protected in the standard model.

Index Terms—Distributed Data Storage, Identity-based System, Access Control, Security

1 Introduction

CLOUD computing provides users with a convenient mechanism to manage their personal files with the notion called database-as-a-service (DAS) [1], [2], [3]. In DAS schemes, a user can outsource his encrypted files to untrusted proxy servers. Proxy servers can perform some functions on the outsourced ciphertexts without knowing anything about the original files. Unfortunately, this technique has not been employed extensively. The main reason lies in that users are especially concerned on the confidentiality, integrity and query of the outsourced files as cloud computing is a lot more complicated than the local data storage systems, as the cloud is managed by an untrusted third party. After outsorcing the files to proxy servers, the user will remove them from his local machine. Therefore, how to guarantee the outsoured files are not accessed by the unauthorized users and not modified by proxy servers is an important problem that has been considered in the data storage research community. Furthermore, how to guarantee that an authorized user can query the outsourced files from proxy servers is another concern as the proxy server only maintains the outsourced ciphertexts. Consequently, research around these topics grows significantly.

- *J. Han, is with Centre for Computer and Information Security Research, School of Computer Science and Software Engineering, University of Wollongong, NSW2522, Australia. College of Science, Hohai University, Nanjing 210098, China. E-mail: jh843@uow.edu.au*
- *W. Susilo and Y. Mu are with Centre for Computer and Information Security Research, School of Computer Science and Software Engineering, University of Wollongong, NSW2522, Australia. E-mail:wsusilo@uow.edu.au; ymu@uow.edu.au*

Confidentiality is proposed to prevent unauthorized users from accessing the sensitive data as it is subject to unauthorized disclose and access after being outsourced. Since the introduction of DAS, the confidentiality of outsourced data has been the primary focus among the research community. To provide confidentiality to the outsourced data, encryption schemes are deployed [4], [5], [6], [7], [8].

Integrity can prevent outsourced data from being replaced and modified. Some schemes have been proposed to protect the integrity of the outsourced data, such as proof of retrievability [9], [10], [11], [12], [13] and provable data possession [14], [15], [16]. In these schemes, digital signature schemes and message authentication codes (MAC) are deployed.

Query in data storage is executed between a receiver and a proxy server. The proxy server can perform some functions on the outsourced ciphertexts and convert them to those for the receiver. As a result, the receiver can obtain the data outsourced by the owner without the proxy server knowing the content of the data [17], [18], [19], [20].

1.1 Related Work

In this section, we review schemes related to identity-based secure distributed data storage (IBSDDS) schemes.

1.1.1 Data Storage Systems

Data storage systems enable users to store their data to external proxy servers to enhance the access and availability, and reduce the maintainance cost. Samarati and Vimercati [21] addressed the privacy issues in data

Digital Object Indentifier 10.1109/TC.2013.26

outsourcing expanding from the data confidentiality to data utility, and pointed out the main research directions in the protection of the externally stored data. Kher and Kim [22] surveyed the data storage systems comprehensively and classified them into three kinds based on their security services: networked file systems (NFS), storage-based intrusion detection systems (SBIDS) and cryptographic file systems (CFS).

Networked File Systems. In these systems, proxy servers are assumed to be trusted. They authenticate receivers and validate access permissions. The interactions between the proxy servers and receivers are executed in a secure channel. Therefore, these systems cannot provide an end-to-end data security, namely they cannot ensure the confidentiality of the data stored at the proxy server [23], [24], [25]. In these schemes, a receiver authenticates himself to the proxy server using his password. Then, the proxy sever passes the authentication result to the file owner. The owner will make an access permission according to the received information.

Storage-based Intrusion Detection Systems. In these systems, an intrusion detection scheme is embedded in proxy servers or the file owner to detect the intruder's behaviors, such as adding backdoors, inserting Trojan horses and tampering with audit logs. These schemes can be classified into two types: host-based system and network-based system. In the host-based systems, an intrusion detection scheme is embedded in the host to detect the local intrusion actions [26]. On the contrary, in network-based systems, an intrusion detection scheme is embedded in the proxy servers to detect the external intruder's actions. The main advantage of these systems is that proxy servers can still detect the intrusion actions even if the host is compromised as the proxy server are independent from the host [27], [28], [29].

Cryptographic File System. In these systems, an end-to-end security is provided by cryptographic protocols which are executed by the file owner to prevent proxy servers and unauthorized users from modifying and accessing the sensitive files. These systems can be divided into two types: shared file system and non-shared system. In shared file systems [30], [31], the owner can share his files with a group of users. Cryptographic techniques deployed in these systems are key sharing, key agreement and key revocation. In non-shared file systems [32], [33], in order to share a file with another user, the owner can compute an access key for the user using his secret key. In these two systems, the integrity of the sensitive files is provided by digital signature schemes and message authentication codes (MAC).

1.1.2 Identity-based Proxy Re-encryption

Proxy cryptosystem was introduced by Mambo and Okamoto [34] to delegate the decryption power to a designated decryptor. Then, Blaze, Bleumer, and Strauss [35] proposed an atomic proxy cryptosystem where a semi-trusted proxy server can transfer a ciphertext for the original decryptor to a ciphertext for the designated decryptor without knowing the plaintext. Proxy cryptosystem as an efficient primitive has been used in email forwarding, law enforcement and data storage. Identity-based cryptosystem introduced by Shamir [36] is a system where the public key can be any arbitrary string and the secret key is issued by a trusted party called the private key generator (PKG). Being different from public key infrastructure (PKI), two parties can communicate directly without verifying their public key certificates in identity-based systems. The first secure and practical identity-base encryption (IBE) was proposed by Boneh and Franklin [37] based on pairing.

Identity-based proxy encryption (IBPE) was first proposed by Ivan and Dodis [4] where the formal definitions and security models for both unidirectional and bidirectional IBPE schemes were formalized. In their schemes, the master secret key which is used to extract secret keys for users is split into two parts. One is sent to the proxy server and the other is sent to the user. The user can decrypt a ciphertext for him with the help of the proxy server. Consequently, Ateniese, Fu, Green and Hohenberger [5] pointed out that these schemes are not secure against the collusion attacks, namely the master secret key can be exposed if the user can compromise the proxy server. The first identity-based proxy re-encryption (IBPRE) was proposed by Green and Ateniese [38] where the proxy sever can transfer a ciphertext for the original decryptor to a ciphertext for the designated decryptor after he gets a re-encryption key from the former. We divide the IBPRE schemes into the following two types based on the generation of the re-encryption key:

The re-encryption key can be computed by the original decryptor [38], [39], [40]. In these schemes, for a decryption request, the original decryptor selects a random number and computes a re-encryption key by randomizing his secret key. Then, he encrypts the selected random number under the receiver's identity. Finally, he sends the re-encryption key and the ciphertext to the proxy server. Using the re-encryption key, the proxy server can transfer a ciphertext for the original decryptor to a ciphertext for the designated decryptor. The designated decryptor decrypts the ciphertext using his secret key and obtains the random number selected by the original decrptor. Then, he can decrypt the re-encrypted ciphertext by the random number. Unfortunately, these schemes are vulnerable to the collusion attacks. If the designated decryptor can compromise the proxy server, they can decrypt the ciphertext, obtain the random number selected by the original decryptor and compute the secret key of the original decryptor.

The re-encryption key must be computed by the PKG [41], [42], [43]. In these schemes, the PKG computes the re-encryption key by checking the secret keys of the original decryptor and the designated decryptor.

1.1.3 Identity-based Secure Distributed Data Storage

In an identity-based secure distributed data storage (IBSDDS) scheme, a user's identity can be an arbitrary string and two parties can communicate with each other without checking the public key certificates. At first, the file owner encrypts his files under his identity prior to outsourcing them to proxy servers. Then, he sends the ciphertexts to the proxy servers. Consequently, the proxy servers can transfer a ciphertext encrypted under the identity of the owner to a ciphertext encrypted under the identity of the receiver after they has obtained an access permission (re-encryption key) from the owner.

To provide confidentiality for the outsouced data, an efficient IBSDDS scheme should provide the following properties.

1) *Unidirectional.* After receiving an access permission, the proxy server can transfer a ciphertext for Alice to a ciphertext for Bob while he cannot transfer a ciphertext for Bob to a ciphertext for Alice.
2) *Non-interactive.* The access permission can be created by the file owner without any trusted third party and interaction with him.
3) *Key optimal.* The size of the secret key of the receiver is constant and independent of the delegations which he accepts.
4) *Collusion-safe.* The secret key of the file owner is secure even if the receiver can compromise the proxy server.
5) *Non-transitive.* Receiving the access permissions computed by Alice for Bob and Bob for Charlie, the proxy server cannot transfer a ciphertext for Alice to a ciphertext for Charlie.
6) *File-based access.* For one query, the receiver can only access one file. This can improve the security of the outsourced files and is desirable to maintain the access record.

Here, 1)-5) are from [5]. *Proxy invisibility* discussed in [5] is difficult to achieve as the length of the re-encrypted ciphertext is subject to be different from that of the original ciphertext. Furthermore, *original-access* mentioned in [5] cannot be guaranteed as the key escrow problem, namely the secret key is created by the PKG, instead of the user. Hence, the file owner in an IBSDSS scheme has less control on his secret key than that in other public key encryption schemes.

Although IBPRE holds partial properties of IBSDDS, it cannot been used in IBSDDS systems directly. For example, in the current IBPRE schemes, the receiver and the proxy servers can cooperate to access all the files outsourced by the owner as the access permission (re-encryption key) is only bound to the identity of the receiver and independent of the file. This is undesirable for the file owner to record the accessed number of his files. Furthermore, they are interactive [41], [42], [43] or *not* collusion safe [38], [39], [40].

Since the PKG can generate a secret key for each user, he can decrypt the ciphertexts and obtain the original files if he knows the identity used to encrypt the files. Therefore, in this paper, we assume that the PKG is honest and can be trusted by all users in the systems.

1.2 Our Contribution

In this paper, we propose two identity-based secure distributed data storage (IBSDDS) schemes in standard model where, for one query, the receiver can only access one of the owner's files, instead of all files. In other words, an access permission (re-encryption key) is bound not only to the identity of the receiver but also the file. The access permission can be decided by the owner, instead of the trusted party (PKG). Furthermore, our schemes are secure against the collusion attacks. Although the first scheme is CPA secure, the second scheme achieves CCA security. To the best of our knowledge, it is the *first* IBSDDS schemes where an access permission is made by the owner for an exact file and collusion attacks can be protected in the standard model.

To achieve a stronger security and implement file-based access control, the owner must be online to authenticate requesters and also to generate access permissions for them. Therefore, the owner in our schemes needs do more computations than that in PRE schemes. Although PRE schemes can provide the similar functionalities of our schemes when the owner only has one file, these are not flexible and practical.

1.3 Paper Organization

We review the preliminaries used throughout the paper in Section 2. In Section 3, we propose a CPA secure IBSDDS scheme and analyze its security. A CCA secure IBSDDS scheme is proposed and proven in Section 4. Section 5 concludes the paper.

2 Preliminaries

In the remainder of this paper, we denote $a \xleftarrow{R} A$ as a is chosen from A at random. Especially, we denote $a \xleftarrow{R} A$ as a is chosen uniformly from A if A is a finite set. For $n \in \mathbb{N}$, we denote $[n]$ as the integers $\{1, 2, \cdots, n\}$. By $A(x) \rightarrow y$, we denote that y is computed by running the algorithm A on input x. We say that a function $\epsilon : \mathbb{Z} \rightarrow \mathbb{R}$ is negligible if, for all $k \in \mathbb{Z}$, there exists a $z \in \mathbb{Z}$ such that $\epsilon(x) \leq \frac{1}{x^k}$ when $x > z$.

2.1 Identity-based Secure Distributed Data Storage (IBSDDS)

There are four entities in an identity-based secure distributed data storage (IBSDDS) scheme: the private key generator (PKG), the data owner, the proxy server and the receiver. The PKG validates the users' identities and issues secret keys to them. The data owner encrypts his data and outsources the ciphertexts to the proxy servers.

Proxy servers store the encrypted data and transfer the ciphertext for the owner to the ciphertext for the receiver when they obtains an access permission (re-encryption key) from the owner. The receiver authenticates himself to the owner and decrypts the re-encrypted ciphertext to obtain the data. An IBSDDS scheme consists of the following algorithms:

$\mathsf{Setup}(1^\ell) \to (params, MSK)$. The setup algorithm takes as input a security parameter 1^ℓ, and outputs the public parameters $params$ and a master secret MSK.

$\mathsf{KeyGen}(params, ID, MSK) \to SK_{ID}$. The key generation algorithm takes as input the public parameters $params$, an identity ID and the master secret key MSK, and outputs a secret key SK_{ID} for the identity ID.

$\mathsf{Encryption}(params, ID, M_i) \to CT_i$. Suppose that there are k messages $\{M_1, M_2, \cdots, M_k\}$. To encrypt the message M_i, the encryption algorithm takes as input the public parameters $params$, the identity ID and the message M_i, and outputs the ciphertext $CT_i = (C_{i,1}, C_{i,2})$, for $i = 1, 2, \cdots, k$. It sends the ciphertexts CT_i to the proxy servers.

$\mathsf{Query}(ID', SK_{ID'}, CT_i) \to AI$. The query algorithm takes as input the receiver's identity ID', the receiver's secrete key $SK_{ID'}$ and the ciphertext CT_i, and outputs an authentication information AI. It sends (ID', AI, CT_i) to the proxy server. The proxy server redirects $(ID', AI, C_{i,2})$ to the owner with identity ID.

$\mathsf{Permission}(params, ID', C_{i,2}, SK_{ID}) \to RK_{ID \to ID'}$ The permission algorithm checks the authentication information AI. If the receiver is legal, this algorithm takes as inputs the public parameters $params$, the receiver's identity ID' and the owner's secret key SK_{ID}, and outputs an access permission (re-encryption key) $RK_{ID \to ID'}$. It sends $RK_{ID \to ID'}$ to the proxy server.

$\mathsf{Re\text{-}encryption}(params, ID', RK_{ID \to ID'}, CT_i) \to CT'_i$. The re-encryption algorithm takes as input the public parameters $params$, the receiver's identity ID', the access permission $RK_{ID \to ID'}$ and the ciphertext CT_i, and outputs a ciphertext $CT'_i = \mathsf{Encryption}(params, ID', M_i)$ for the receiver with identity ID'.

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

1) $\mathsf{Decryption}_1(params, SK_{ID}, CT_i) \to M_i$. The owner decryption algorithm takes as input the public parameters $params$, the owner's secret key SK_{ID} and the ciphertext CT_i, and outputs the message M_i.
2) $\mathsf{Decryption}_2(params, SK_{ID'}, CT'_i) \to M_i$. The receiver decryption algorithm takes as input the public parameters $params$, the receiver's secret key $SK_{ID'}$ and the re-encrypted ciphertext CT'_i, and outputs the message M_i.

Definition 1. *We say an identity-based secure distributed data storage (IBSDDS) scheme is correct if*

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

and

$$\Pr\left[\begin{array}{l|l} \mathsf{Decryption}_2(params, SK_{ID'}, CT'_i) \to M_i & \begin{array}{l}\mathsf{Setup}(1^\ell) \to (params, MSK);\\ \mathsf{KeyGen}(params, ID, MSK) \to SK_{ID};\\ \mathsf{KeyGen}(params, ID', MSK) \to SK_{ID'};\\ \mathsf{Permission}(params, ID', C_{i,2}, SK_{ID}) \to RK_{ID \to ID'};\\ \mathsf{Re\text{-}encryption}(params, ID', RK_{ID \to ID'}, CT_i) \to CT'_i\end{array}\end{array}\right] = 1$$

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

2.2 Security Model

We formalize the security model of the IBSDDS scheme by the following game. This game is run between a challenger and an adversary as follows:

Setup. The challenger runs $\mathsf{Setup}(1^\ell)$ to generate the public parameters $params$ and a master secret key MSK, and sends $params$ to the adversary $\mathcal{A}$.

Phase 1. The adversary $\mathcal{A}$ can adaptively make the following queries:

1) Secret Key Query. The adversary $\mathcal{A}$ can query secret key for an identity ID. The challenger runs $\mathsf{KeyGen}\ (params, ID, MSK)$ to generate a secret key SK_{ID}. The challenger responds $\mathcal{A}$ with SK_{ID}.
2) Permission Query. The adversary $\mathcal{A}$ can query a permission on $(ID, ID', C_{i,2})$. The challenger runs $\mathsf{KeyGen}(params, ID, MSK)$ to extract the secret key SK_{ID} and $\mathsf{Permission}(\ params, ID', C_{i,2}, SK_{ID})$ to obtain $RK_{ID \to ID'}$. The challenger responds $\mathcal{A}$ with $RK_{ID \to ID'}$.
3) Re-encryption Query. The adversary $\mathcal{A}$ can query re-encryption on (ID, ID', CT_i). The challenger runs $\mathsf{KeyGen}(params, ID, MSK)$ to generate a secret key SK_{ID}, and runs $\mathsf{Permission}(params, ID', C_{i,2}, SK_{ID})$ to obtain $RK_{ID \to ID'}$. The challenger responds $\mathcal{A}$ with $\mathsf{Re\text{-}encryption}\ (params, ID', RK_{ID \to ID'}, CT_i)$.
4) Owner Decryption Query. The adversary $\mathcal{A}$ can query owner decryption on (ID, CT_i). The challenger runs $\mathsf{KeyGen}(params, ID, MSK)$ to extract

the secret key SK_{ID}. The challenger responds $\mathcal{A}$ with $\mathsf{Decryption}_1(params, SK_{ID}, CT_i)$.

5) Receiver Decryption Query. The adversary $\mathcal{A}$ can query receiver decryption on (ID, ID', CT_i). The challenger runs $\mathsf{KeyGen}(params, ID, MSK)$ and $\mathsf{KeyGen}(params, ID', MSK)$ to extract the secret keys SK_{ID} and SK'_{ID}, $\mathsf{Permission}(params, ID', C_{i,2}, SK_{ID})$ to obtain $RK_{ID\to ID'}$ and $\mathsf{Re\text{-}encryption}(params, ID', RK_{ID\to ID'}, CT_i)$ to get CT'_i. The challenger responds $\mathcal{A}$ with $\mathsf{Decryption}_2(params, SK'_{ID}, CT'_i)$.

Challenge. The adversary $\mathcal{A}$ submits an identity ID^* and two messages M_0 and M_1 with equal length. The challenger flips an unbiased coin with $\{0,1\}$ and obtains $b \in \{0,1\}$. The challenger computes $C^* = \mathsf{Encryption}(parmas, ID^*, M_b)$ and sends C^* to $\mathcal{A}$.

Phase 2. The adversary can adaptively make queries as in Phase 1 with the following restricts:

1) Secret key Query. The adversary $\mathcal{A}$ cannot query $KeyGen(params, ID^*, MSK)$.
2) Permission Query. The adversary $\mathcal{A}$ cannot query $\mathsf{Permission}(ID^*, ID, C_2^*)$ and $\mathsf{KeyGen}(params, ID, MSK)$.
3) Re-encryption Query. The adversary $\mathcal{A}$ cannot query re-encryption on (ID^*, ID, CT^*), $\mathsf{Permission}(ID^*, ID, C_2^*)$ and $\mathsf{KeyGen}(params, ID, MSK)$.
4) Owner Decryption Query. The adversary $\mathcal{A}$ cannot query owner decryption on (ID^*, CT^*).
5) Receiver Decryption Query. The adversary $\mathcal{A}$ cannot query re-encryption on (ID^*, ID, CT^*) and receiver decryption on $(ID, \tilde{CT}^*)$, where $\tilde{CT}^*$ is the re-encrypted ciphertext of CT^*.

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

Definition 2. *An identity-based secure distributed data storage (IBSDDS) scheme is $(T, q_1, q_2, q_3, q_4, q_5, \epsilon(\ell))$-secure against chosen ciphertext attacks (CCA) if no probabilistic polynomial-time adversary $\mathcal{A}$ making at most q_1 secret key queries, q_2 permission queries, q_3 re-encryption queries, q_4 owner decryption queries and q_5 receiver decryption queries can win the game with the advantage*

$$Adv_{\mathcal{A}}^{CCA} = |\Pr[b' = b] - \frac{1}{2}| \geq \epsilon(\ell)$$

in the above model.

Definition 3. *An identity-based distributed data storage (IBDDS) scheme is $(T, q_1, q_2, q_3, \epsilon(\ell))$-secure against chosen plaintext attacks (CPA) if no probabilistic polynomial-time adversary $\mathcal{A}$ making at most q_1 secret key queries, q_2 permission queries and q_3 re-encryption queries can win the game with the advantage*

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

in the above model.

Theorem 1. *An identity-based secure distributed data storage (IBSDDS) scheme is unidirectional, nontransitive and collusion safe if it is secure against the chosen plaintext attacks (CPA) in the above model.*

Proof: Our proof is similar to that in [42]. In the CPA security model, the adversary $\mathcal{A}$ can query secret key oracle, permission oracle and re-encryption oracle.

Collusion-safe. If the scheme is not collusion safe, there exists an algorithm $\mathcal{B}$ that can use $\mathcal{A}$ to break the CPA security in the above security model. $\mathcal{A}$ can query secret key $SK_{ID'}$ for an identity ID' and permission $RK_{ID\to ID'}$ from an identity ID to ID'. After receiving the challenged ciphertext CT^* for the identity ID, if $\mathcal{A}$ can compute the secret key SK_{ID} from $SK_{ID'}$ and $RK_{ID\to ID'}$, $\mathcal{B}$ can use SK_{ID} to decrypt CT^*. Hence, $\mathcal{B}$ can use $\mathcal{A}$ to break the CPA security in the above model.

Nontransitive. If the scheme is transitive, there exists an algorithm $\mathcal{B}$ that can use $\mathcal{A}$ to break the CPA security in the above security model. $\mathcal{A}$ can query secret keys $SK_{ID'}$ and $SK_{ID''}$ for identities ID' and ID''. Furthermore, $\mathcal{A}$ can query permissions $RK_{ID\to ID'}$ and $RK_{ID'\to ID''}$. After receiving the challenged ciphertext CT^* for the identity ID, if $\mathcal{A}$ can compute the permission $RK_{ID\to ID''}$ from $RK_{ID\to ID'}$ and $RK_{ID'\to ID''}$, $\mathcal{B}$ can use $RK_{ID\to ID''}$ to transfer the ciphertext CT^* to a ciphertext $\hat{CT}$ for the identity ID''. Then, $\mathcal{B}$ can use $SK_{ID''}$ to decrypt $\hat{CT}$. So, $\mathcal{B}$ can use $\mathcal{A}$ to break the CPA security in the above security model.

Unidirectional. If the scheme is not unidirectional in the above model, there exists an algorithm $\mathcal{B}$ that can use $\mathcal{A}$ to break the CPA security in the above security model. The adversary $\mathcal{A}$ can query secret key $SK_{ID'}$ for an identity ID' and permission $RK_{ID'\to ID}$ from an identity ID' to ID. After receiving the challenged ciphertext CT^* for the identity ID, if $\mathcal{A}$ can use $RK_{ID'\to ID}$ to transfer CT^* to a ciphertext $\hat{CT}$ for identity ID'. Then, $\mathcal{B}$ can use the secret key $SK_{ID'}$ to decrypt $\hat{CT}$. Therefore, $\mathcal{B}$ can use $\mathcal{A}$ to break the CPA security in the above model. □

2.3 Complexity Assumption

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

1) Bilinearity. For all $u, v \in \mathbb{G}$ and $x, y \in \mathbb{Z}_p$, $e(u^x, v^y) = e(u, v)^{xy}$.
2) No-degeneracy. $e(g, g) \neq 1$ where 1 is the identity of the group $\mathbb{G}_\tau$.
3) Computability. There exists an efficient algorithm which can compute $e(u, v)$ for all $u, v \in \mathbb{G}$.

We denote $\mathcal{GG}(1^\ell)$ as a bilinear group generator 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 4. Decisional Bilinear Diffie-Hellman (DBDH) Assumption [37]. *Let* $a, b, c, z \overset{R}{\leftarrow} \mathbb{Z}_p$, $\mathcal{GG}(1^\ell) \rightarrow (e, p, \mathbb{G}, \mathbb{G}_\tau)$, *and* g *be a generator of* $\mathbb{G}$. *We say that the DBDH assumption holds in* $(e, p, \mathbb{G}, \mathbb{G}_\tau)$ *if no probabilistic polynomial-time adversary* $\mathcal{A}$ *can distinguish* $(A, B, C, Z) = (g^a, g^b, g^c, e(g,g)^{abc})$ *from* $(A, B, C, Z) = (g^a, g^b, g^c, e(g,g)^z)$ *with the advantage*

$$Adv_{\mathcal{A}}^{DBDH} = \left| \begin{array}{l} \Pr[\mathcal{A}(A, B, C, e(g,g)^{abc})] \\ -\Pr[\mathcal{A}(A, B, C, e(g,g)^z)] \end{array} \right| \geq \epsilon(\ell)$$

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

2.4 Waters's Identity-based Encryption

This identity-based encryption (IBE) [44] works as follows:

Setup. This algorithm takes as input the security parameters 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 and η be generators of the group $\mathbb{G}$, $u_0 \overset{R}{\leftarrow} \mathbb{G}$ and $U = (u_1, u_2, \cdots, u_n)$ where $u_i \overset{R}{\leftarrow} \mathbb{G}$ for $i = 1, 2, \cdots, n$. It sets $g_1 = g^\alpha$ where $\alpha \overset{R}{\leftarrow} \mathbb{Z}_p$. The public parameters are $(e, p, \mathbb{G}, \mathbb{G}_\tau, g, \eta, u_0, g_1, U)$ and the master secret key is η^α.

KeyGen. Let ID represent an identity which is an n bit string, ID_i be the ith bit of ID, and $\mathcal{I}$ be the set which consists of all i with $ID_i = 1$. This algorithm chooses $r \overset{R}{\leftarrow} \mathbb{Z}_p$, and computes

$$K_{ID,1} = \eta^\alpha (u_0 \prod_{i \in \mathcal{I}} u_i)^r \quad \text{and} \quad K_{ID,2} = g^r.$$

The secret key for the identity ID is $SK_{ID} = (K_{ID,1}, K_{ID,2})$.

Encryption. To encrypt a message $M \in \mathbb{G}_\tau$, this algorithm chooses $s \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes

$$C_1 = M \cdot e(g_1, \eta)^s, \quad C_2 = g^s \quad \text{and} \quad C_3 = (u_0 \prod_{i \in \mathcal{I}} u_i)^s.$$

The ciphertext for the message M is $CT = (C_1, C_2, C_3)$.

Decryption. To decrypt the ciphertext $CT = (C_1, C_2, C_3)$, this algorithm takes as input the secret key $SK_{ID} = (K_{ID,1}, K_{ID,2})$ and computes

$$M = C_1 \cdot \frac{e(K_{ID,2}, C_3)}{e(K_{ID,1}, C_2)}$$

Theorem 2. *This identity-based encryption scheme is* $(T, q, \epsilon(\ell))$*-secure agasint chosen plaintex attacks (CPA) if the* $(T + \mathcal{O}(\epsilon(\ell)^{-2} ln(\epsilon(\ell)^{-1}) \lambda^{-1} ln(\lambda^{-1})), \frac{\epsilon(\ell)}{32q(n+1)})$*-decisional bilinear Diffie-Hellman (DBDH) assumption holds on the bilinear group* $(e, p, \mathbb{G}, \mathbb{G}_\tau)$, *where* $\lambda = \frac{1}{8q(n+1)}$ [44].

3 Identity-based Secure Distributed Data Storage I

In this section, we propose an identity-based secure distributed data storage (IBSDDS I) scheme which is secure against chosen plaintext attacks (CPA). At first, the file owner encrypts his files and outsources the ciphertexts to the proxy servers. The proxy servers validate the outsourced ciphertexts and store them for the owner. For one query, the receiver computes an authentication information (AI) using his secret key and sends it to the proxy server. The proxy server sends the identity of the receiver, AI and the partial intended ciphertext to the owner. Suppose that the owner can know which file the receiver wants to access from the partial ciphertext. To check whether the requester is a legal user in the system, the owner validates the the received AI. If the AI is correct, the owner computes an access permission (re-encryption key) using his secret key, the partial ciphertext and the identity of the receiver, and sends it to the proxy server. Otherwise, the access is denied. The proxy sever transfers the intended ciphertext to a ciphertext for the receiver using the received access permission. Finally, the receiver can decrypt the re-encrypted ciphertext by his secret key and obtains the original file. Fig.1 explains the model of our IBSDDS schemes.

The specific protocol of our IBSDDS I scheme is demonstrated in Fig.2. Our scheme can be seen as an extension of Water's IBE [44].

Correctness.
We have

$$\begin{aligned} & C_{i,1} \cdot \frac{e(K_{ID,2}, C_{i,3})}{e(K_{ID,1}, C_{i,2})} \\ = \; & M_i \cdot e(g_1, \eta)^{s_i} \frac{e(g^{r_{ID}}, (u_0 \prod_{i \in \mathcal{I}} u_i)^{s_i})}{e(\eta^\alpha (u_0 \prod_{i \in \mathcal{I}} u_i)^{r_{ID}}, g^{s_i})} \\ = \; & M_i \cdot e(g_1, \eta)^{s_i} \frac{e(g^{r_{ID}}, (u_0 \prod_{i \in \mathcal{I}})^{s_i})}{e(g_1, \eta)^{s_i} \cdot e(g^{r_{ID}}, (u_0 \prod_{i \in \mathcal{I}} u_i)^{s_i})} \\ = \; & M_i \cdot e(g_1, \eta)^{s_i} \cdot \frac{1}{e(g_1, \eta)^{s_i}} \\ = \; & M_i, \end{aligned}$$

$$\begin{aligned} K_1 &= K'_{ID',1} \cdot C'^{t}_{i,5} \cdot C'_{i,4} \\ &= K'_{ID',1} \cdot \mathfrak{g}^{\rho t} \cdot \frac{K_{ID,1}}{K'_{ID',1} \cdot \Gamma^\rho} \cdot (u_0 \prod_{i \in \mathcal{I}'} u_i)^\beta \\ &= K'_{ID',1} \cdot \Gamma^\rho \frac{K_{ID,1}}{K'_{ID',1} \cdot \Gamma^\rho} \cdot (u_0 \prod_{i \in \mathcal{I}'} u_i)^\beta \\ &= K_{ID,1} \cdot (u_0 \prod_{i \in \mathcal{I}} u_i)^\beta \\ &= \eta^\alpha (u_0 \prod_{i \in \mathcal{I}} u_i)^{r_{ID}} (u_0 \prod_{i \in \mathcal{I}'} u_i)^\beta \end{aligned}$$

and

$$\begin{aligned} C'_{i,1} &= D_2 \cdot C_{i,1} \\ &= M_i \cdot e(g_1, \eta)^{s_i} \cdot e(g, (u_0 \prod_{i \in \mathcal{I}'} u_i))^{\beta s_i}. \end{aligned}$$

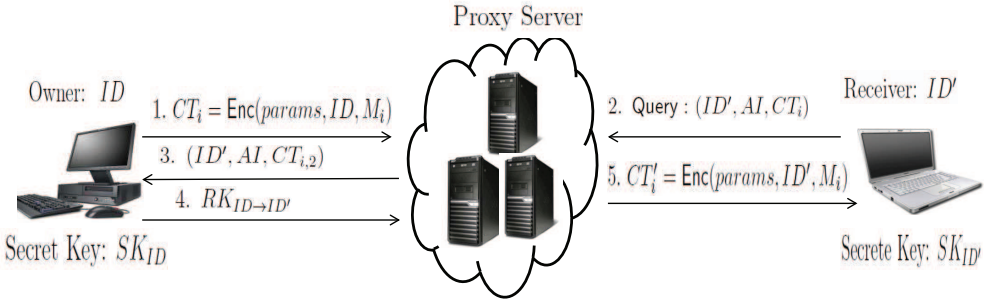


Fig. 1: The Model of Identity-Based Secure Distributed Data Storage Scheme

Therefore

$$\begin{aligned} & C'_{i,1} \cdot \frac{e(C'_{i,6}, C'_{i,3})}{e(K_1, C_{i,2})} \\ = \; & C'_{i,1} \cdot \frac{e(g^{r_{ID}}, (u_0 \prod_{i\in\mathcal{I}} u_i)^{s_i})}{e(\eta^{\alpha}(u_0 \prod_{i\in\mathcal{I}} u_i)^{r_{ID}}(u_0 \prod_{i\in\mathcal{I}'} u_i)^{\beta}, g^{s_i})} \\ = \; & C'_{i,1} \cdot \frac{1}{e(g_1,\eta)^{s_i} \cdot e(g, (u_0 \prod_{i\in\mathcal{I}'} u_i))^{\beta s_i}} \\ = \; & M_i \cdot \frac{e(g_1,\eta)^{s_i} \cdot e(g, (u_0 \prod_{i\in\mathcal{I}'} u_i))^{\beta s_i}}{e(g_1,\eta)^{s_i} \cdot e(g, (u_0 \prod_{i\in\mathcal{I}'} u_i))^{\beta s_i}} \\ = \; & M_i \end{aligned}$$

Theorem 3. *Our identity-based secure distributed data storage I (IBSDDS I) scheme is $(T, q_1, q_2, q_3, \epsilon(\ell))$-secure against chose plaintext attacks (CPA) if the $(T', \epsilon'(\ell))$-decisional bilinear Diffie-Hellman (DBDH) assumption holds in the bilinear group $(e, p, \mathbb{G}, \mathbb{G}_\tau)$ where*

$$T' = T + \mathcal{O}(T) \;\; \textit{and} \;\; \epsilon'(\ell) = \frac{\epsilon(\ell)}{32(q_1 + 2q_2 + 2q_3)(n+1)}.$$

Proof: Our proof is similar to that in Waters's IBE [44], except that we must answer the permission queries and re-encryption queries.

Suppose that there exists an adversary $\mathcal{A}$ that can $(T, q_1, q_2, q_3, \epsilon(\ell))$ break the CPA security of our IBS-DDS I scheme, we can construct an algorithm $\mathcal{B}$ that can use $\mathcal{A}$ to break the DBDH assumption as follows. The challenger generates the bilinear group $\mathcal{GG}(1^\ell) \to (e, p, \mathbb{G}, \mathbb{G}_\tau)$ and chooses a generator $g \xleftarrow{R} \mathbb{G}$. 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 $\mathcal{B}$. Otherwise, he sends $(A, B, C, Z) = (g^a, g^b, g^c, e(g,g)^z)$ to $\mathcal{B}$, where $z \xleftarrow{R} \mathbb{Z}_p$. The algorithm $\mathcal{B}$ will output his guess μ' on μ.

Setup. The algorithm $\mathcal{B}$ sets $\sigma = 4(q_1 + 2q_2 + 2q_3)$ and chooses an integer $\nu \xleftarrow{R} [n]$. It uniformly selects two integrity vectors $\Pi = \{\pi_1, \pi_2, \cdots, \pi_n\}$ and $\Phi = \{\phi_1, \phi_2, \cdots, \phi_n\}$, where $\pi_i \xleftarrow{R} [\sigma - 1]$ and $\phi_i \xleftarrow{R} \mathbb{Z}_p$ for $i = 1, 2, \cdots, n$. It choose $\pi_0 \xleftarrow{R} [\sigma - 1]$ and $\phi_0 \xleftarrow{R} \mathbb{Z}_p$. Then, the algorithm $\mathcal{B}$ defines three functions:

$$P(ID) = (p - \sigma\nu) + \pi_0 + \sum_{i\in\mathcal{I}} \pi_i,$$

$$Q(ID) = \phi_0 + \sum_{i\in\mathcal{I}} \phi_i$$

and

$$R(ID) = \begin{cases} 0, & \text{if } \pi_0 + \sum_{i\in\mathcal{I}} \pi_i \equiv 0 \pmod{\sigma} \\ 1, & \text{if } \pi_0 + \sum_{i\in\mathcal{I}} \pi_i \not\equiv 0 \pmod{\sigma} \end{cases}$$

$\mathcal{B}$ sets $g_1 = A$, $\eta = B$, $\mathfrak{g} = g^\theta$, $g_2 = A^\theta$, $u_0 = \eta^{p-\sigma\nu+\pi_0} g^{\phi_0}$ and $u_i = \eta^{\pi_i} g^{\phi_i}$, where $\theta \xleftarrow{R} \mathbb{Z}_p$. It chooses $\mathfrak{h} \xleftarrow{R} \mathbb{G}$. The public parameters are $(e, p, \mathbb{G}, \mathbb{G}_\tau, g, h, \eta, \mathfrak{g}, \mathfrak{h}, u_0, g_1, g_2, \mathbb{U})$. The master secret key is $\eta^a = g^{ab}$. The distribution of these parameters is identical to those in the real protocol.

Phase 1.

1) Secret Key Query. The adversary $\mathcal{A}$ can query secret key for an identity ID. $\mathcal{B}$ checks $R(ID) \stackrel{?}{=} 1$.
   a) If $R(ID) = 1$, $\mathcal{B}$ chooses $r \xleftarrow{R} \mathbb{Z}_p$ and computes

$$K_{ID,1} = A^{\frac{-Q(ID)}{P(ID)}} (\pi_0 \prod_{i\in\mathcal{I}} \pi_i)^r, \tag{1}$$

$$K_{ID,2} = A^{\frac{-1}{P(ID)}} g^r \tag{2}$$

   and

$$K_{ID,3} = K_{ID,2}^{\theta}. \tag{3}$$

   $\mathcal{B}$ responds with $SK_{ID} = (K_{ID,1}, K_{ID,2}, K_{ID,3})$.
   b) If $R(ID) = 0$, $\mathcal{B}$ aborts and outputs his guess μ' randomly.

We claim that the secret key is generated correctly.

$$\begin{aligned} K_{ID,1} &= A^{\frac{-Q(ID)}{P(ID)}} (u_0 \prod_{i\in\mathcal{I}} u_i)^r \\ &= g^{\frac{-aQ(ID)}{P(ID)}} (g^{bP(ID)+Q(ID)})^r \\ &= (g^{bP(ID)+Q(ID)})^{\frac{-a}{P(ID)}} g^{ab} (g^{bP(ID)+Q(ID)})^r \\ &= g^{ab} (g^{bP(ID)+Q(ID)})^{r - \frac{a}{P(ID)}} \\ &= \eta^a (u_0 \prod_{i\in\mathcal{I}} u_i)^{r - \frac{a}{P(ID)}} \end{aligned}$$

Let $\hat{r} = r - \frac{a}{P(ID)}$, we have

$$K_{ID,1} = \eta^a (u_0 \prod_{i\in\mathcal{I}} u_i)^{\hat{r}},$$

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 $g, h, \eta, \mathfrak{g}$ and $\mathfrak{h}$ be the generators of $\mathbb{G}$, $u_0 \overset{R}{\leftarrow} \mathbb{G}$ and $\mathbb{U} = (u_1, u_2, \cdots, u_n)$ where $u_i \overset{R}{\leftarrow} \mathbb{G}$ for $i = 1, 2, \cdots, n$. It chooses $\alpha \overset{R}{\leftarrow} \mathbb{Z}_p$ and sets $g_1 = g^\alpha$ and $g_2 = \mathfrak{g}^\alpha$. The public parameters are $(e, p, \mathbb{G}, \mathbb{G}_\tau, g, h, \eta, \mathfrak{g}, \mathfrak{h}, u_0, g_1, g_2, \mathbb{U})$ and the master secret key is η^α.

KeyGen. Let ID denote an identity which is an n bit string, ID_i be the ith bit of ID and $\mathcal{I}$ be the set which consists of all the index i with $ID_i = 1$. This algorithm takes as input the master secret key η^α and the user's identity ID, and computes

$$K_{ID,1} = \eta^\alpha (u_0 \prod_{i \in \mathcal{I}} u_i)^{r_{ID}}, \ K_{ID,2} = g^{r_{ID}} \text{ and } K_{ID,3} = \mathfrak{g}^{r_{ID}}.$$

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

$$e(K_{ID,1}, g) \overset{?}{=} e(\eta, g_1) \cdot e((u_0 \prod_{i \in \mathcal{I}} u_i), K_{ID,2}) \text{ and } e(K_{ID,2}, \mathfrak{g}) \overset{?}{=} e(g, K_{ID,3}).$$

Encryption. Suppose that there are k message $\{M_1, M_2, \cdots, M_k\}$. To encrypt M_i, the owner O choose $s_i \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes

$$C_{i,1} = M_i \cdot e(g_1, \eta)^{s_i}, \ C_{i,2} = g^{s_i} \text{ and } C_{i,3} = (u_0 \prod_{i \in \mathcal{I}} u_i)^{s_i}$$

for $i = 1, 2, \cdots, k$. The ciphertext for the message M_i is $CT_i = (C_{i,1}, C_{i,2}, C_{i,3})$. the owner sends $\{CT_1, CT_2, \cdots, CT_k\}$ to the proxy servers. The proxy servers (PSs) validate the ciphertexts by checking

$$e((u_0 \prod_{i \in \mathcal{I}} u_i), C_{i,2}) \overset{?}{=} e(C_{i,3}, g)$$

for $i = 1, 2, \cdots, k$. If the equation holds, the proxy servers store the ciphertext $CT_i = (C_{i,1}, C_{i,2}, C_{i,3})$ for the owner. Otherwise, the proxy servers reject the ciphtertexts.

Query. If a receiver R with identity ID' wants to access CT_i, he chooses $t \overset{R}{\leftarrow} \mathbb{Z}_p$, and computes $K'_{ID',1} = K_{ID',1} \mathfrak{h}^t$ and $\Gamma = \mathfrak{g}^t$. He sends $(ID', K'_{ID',1}, K_{ID',3}, \Gamma)$ to the proxy server. Then, the proxy server redirect $(ID', K'_{ID',1}, K_{ID',3}, \Gamma, C_{i,2})$ to the owner.

Permission. The owner checks whether the receiver has been verified by the PKG by

$$e(K'_{ID',1}, \mathfrak{g}) \overset{?}{=} e(\eta, g_2) \cdot e((u_0 \prod_{i \in \mathcal{I}'} u_i), K_{ID',3}) \cdot e(\mathfrak{h}, \Gamma).$$

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

$$D_1 = \frac{K_{ID,1}}{K'_{ID',1} \cdot \Gamma^\rho} \cdot (u_0 \prod_{i \in \mathcal{I}'} u_i)^\beta, \ D_2 = e(C_{i,2}, (u_0 \prod_{i \in \mathcal{I}'} u_i))^\beta \text{ and } D_3 = \mathfrak{g}^\rho.$$

The owner sends $(D_1, D_2, D_3, K_{ID,2})$ to the proxy server.

Re-encryption. Receiving $(D_1, D_2, D_3, K_{ID,2})$ from the owner, the proxy server computes the re-encrypted ciphterxt as

$$C'_{i,1} = D_2 \cdot C_{i,1}, \ C'_{i,2} = C_{i,2}, \ C'_{i,3} = C_{i,3}, \ C'_{i,4} = D_1, \ C'_{i,5} = D_3 \text{ and } C'_{i,6} = K_{ID,2}.$$

The proxy server responds the receiver with $CT'_i = (C'_{i,1}, C'_{i,2}, C'_{i,3}, C'_{i,4}, C'_{i,5}, C'_{i,6})$.

Decryption.

1) To decrypt the ciphertext $CT_i = (C_{i,1}, C_{i,2}, C_{i,3})$, the owner O computes $M_i = C_{i,1} \cdot \frac{e(K_{ID,2}, C_{i,3})}{e(K_{ID,1}, C_{i,2})}$.
2) To decrypt the ciphertext $CT'_i = (C'_{i,1}, C'_{i,2}, C'_{i,3}, C'_{i,4}, C'_{i,5}, C'_{i,6})$, the receiver R computes $K_1 = K'_{ID',1} \cdot C'^t_{i,5} \cdot C'_{i,4}$. Then, he can compute

$$M_i = C'_{i,1} \cdot \frac{e(C'_{i,6}, C'_{i,3})}{e(K_1, C'_{i,2})}.$$

Fig. 2: IBSDDS I: Identity-Based Secure Distributed Data Storage I

$$K_{ID,2} = A^{\frac{-1}{P(ID)}} g^r = g^{r-\frac{a}{P(ID)}} = g^{\hat{r}}$$

and

$$K_{ID,3} = K_{ID,2}^{\theta} = g^{\theta\hat{r}} = \mathfrak{g}^{\hat{r}}.$$

Therefore, the secret key is created correctly.

Permission Query. The adversary $\mathcal{A}$ can query permission on (ID, ID', C_2). $\mathcal{B}$ checks whether he has generated secret keys for identities ID and ID'. If he has not generated secret keys for ID and ID', $\mathcal{B}$ checks whether $R(ID) = 1$ and $R(ID)' = 1$.

a) If those hold, he computes $K_{ID} = (K_{ID,1}, K_{ID,2}, K_{ID,3})$ and $K_{ID'} = (K_{ID',1}, K_{ID',2}, K_{ID',3})$. Then, he can compute an access permission (the re-encryption key) as follows. $\mathcal{B}$ chooses $t, \beta, \rho \overset{R}{\leftarrow} \mathbb{Z}_p$, and computes

$$K'_{ID',1} = K_{ID',1}\mathfrak{h}^t, \tag{4}$$

$$\Gamma = \mathfrak{g}^t, \tag{5}$$

$$D_1 = \frac{K_{ID,1}}{K'_{ID',1}\Gamma^{\rho}} \cdot (u_0 \prod_{i \in \mathcal{I}'} u_i)^{\beta} \tag{6}$$

$$D_2 = e(C_2, (u_0 \prod_{i \in \mathcal{I}'} u_i))^{\beta}. \tag{7}$$

and

$$D_3 = \mathfrak{g}^{\rho}. \tag{8}$$

$\mathcal{B}$ sends $(D_1, D_2, D_3, K_{ID,2})$ to the adversary $\mathcal{A}$.

b) Otherwise, $\mathcal{B}$ aborts the simulation and outputs his guess μ' randomly.

Re-encryption Query. The adversary can query re-encryption on (ID, ID', C). $\mathcal{B}$ check whether he has generated an access permission $(D_1, , D_2, D_3, K_{ID,2}))$ from identities ID to identity ID'. If he has not generated an access permission from ID to ID', he generated $(D_1, , D_2, D_3, K_{ID,2}))$ as above. Otherwise, $\mathcal{B}$ can compute

$$C'_1 = D_2 \cdot C_1,\ C'_2 = C_2,\ C'_3 = C_3,$$

$$C'_4 = D_1,\ C'_5 = D_3 \text{ and } C'_6 = K_{ID,2}.$$

$\mathcal{B}$ responds with the re-encrypted ciphertext $C' = (C'_1, C'_2, C'_3, C'_4, C'_5, C'_6)$.

Challenge. The adversary $\mathcal{A}$ submits an identity ID^* and two messages M_0 and M_1 with the equal length. The algorithm $\mathcal{B}$ checks $R(ID^*) \overset{?}{=} 0$.

1) If $R(ID^*) = 1$, $\mathcal{B}$ aborts and outputs his guess μ' randomly.

2) If $R(ID^*) = 0$, $\mathcal{B}$ flips an unbiased coin with $\{0, 1\}$ and obtains $\omega \in \{0, 1\}$. The challenger computes $C_1^* = M_\omega \cdot Z$, $C_2^* = C = g^c$ and $C_3^* = C^{Q(ID^*)} = (u_0 \prod_{i \in \mathcal{I}^*} u_i)^c$. $\mathcal{B}$ sends the ciphertext $CT^* = (C_1^*, C_2^*, C_3^*)$ to $\mathcal{A}$.

Phase 2. Phase 1 is repeated with the following restrictions.

1) Secret key Query. The adversary $\mathcal{A}$ cannot query secret key for ID^*.

2) Permission Query. The adversary $\mathcal{A}$ cannot query permission on (ID^*, ID, C_2^*) and secret key for ID.

3) Re-encryption Query. The adversary $\mathcal{A}$ cannot query re-encryption on (ID^*, ID, C^*), permission on (ID^*, ID, C_2^*) and secret key for ID.

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

As shown above, the public parameters and the secret keys created in the simulation paradigm are identical to those created in the real protocol. The algorithm $\mathcal{B}$ does not abort the simulation if and only if the secret keys can be generated correctly and $R(ID^*) = 0$. In q_1 secret key queries, q_2 permission queries and q_3 re-encryption queries, $\mathcal{B}$ needs to create at most $q_1 + 2q_2 + 2q_3$ secret keys. Thereafter, from Theorem 1, the advantage with which $\mathcal{B}$ can break the DBDH assumption is at least $\frac{\epsilon(\ell)}{32(q_1+2q_2+2q_3)(n+1)}$. □

We demonstrate the computation cost and communication cost of our IBSDDS I scheme in Table 1 and Table 2, where by E and P, we denote the running time of executing one exponential and one paring, respectively. By $E_{\mathbb{G}}$ and $E_{\mathbb{G}_\tau}$, we denote the length of one element in the group $\mathbb{G}$ and $\mathbb{G}_\tau$, respectively. By PKG, U, PS, O and R, we denote the private key generator, the user, the proxy server, the data owner and the receiver, respectively. We compare the properties of the related schemes in Table 3.

4 Identity-based Secure Distributed Data Storage II

In some complex network environments, such as cloud computing and distributed systems, CPA security cannot satisfy the application requirement. Therefore, identity-based distributed data storage scheme with strong security (CCA) is desirable. In this section, we propose a CCA-2 secure identity-based secure distributed storage II (IBSDDS II) scheme by introducing an existentially unforgeable one-time signature scheme to the IBSDDS I scheme. This idea is from [45]. Our IBSDDS II scheme is demonstrated in Fig.3.

Correctness. This is the same as in the scheme IBSDDS I.

Theorem 4. *Our identity-based secure distributed data storage scheme II (IBSDDS II) scheme is* $(T, q_1, q_2, q_3, q_4, q_5, \epsilon(\ell))$*-secure against chose ciphertext attacks*

TABLE 1: The Computation cost of Our IBSDDS I Scheme

Scheme	Computation Cost							
	Setup	KeyGen	Encryption	Query	Permission	Re-encryption	O Decryption	R Decryption
IBSDDS I	3E	4E+5P	3E+2P	2E	3E+5P	0	2P	E+2P

TABLE 2: The Communication cost of Our IBSDDS I Scheme

Scheme	Communication Cost						
	KeyGen	Encryption	Query		Permission	O Decryption	R Decryption
	$PKG \to U$	$O \to PS$	$R \to PS$	$PS \to O$	$O \to PS$	$PS \to O$	$PS \to R$
IBSDDS I	$3E_{\mathbb{G}}$	$2E_{\mathbb{G}} + E_{\mathbb{G}_\tau}$	$3E_{\mathbb{G}}$	$4E_{\mathbb{G}}$	$3E_{\mathbb{G}} + E_{\mathbb{G}_\tau}$	$2E_{\mathbb{G}} + E_{\mathbb{G}_\tau}$	$5E_{\mathbb{G}} + E_{\mathbb{G}_\tau}$

TABLE 3: Property Comparison of related Schemes

Property	Matsuo [41]	WWMO [42]	WWMO [43]	GA [38]	CT [39]	THJ [40]	Our Scheme
Unidirectional	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Noninteractive	No	No	No	Yes	Yes	Yes	Yes
Key optimal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Collusion-safe	Yes	Yes	Yes	No	No	No	Yes
Nontransitive	Yes	Yes	Yes	Yes	Yes	Yes	Yes
File-based access	No	No	No	No	No	No	Yes

(CCA) if the one-time signature scheme is $(T', 1, \epsilon'(\ell))$-existentially unforgeable and the $(T'', \epsilon''(\ell))$ decisional bilinear Diffie-Hellman (DBDH) assumption holds in the bilinear group $(e, p, \mathbb{G}, \mathbb{G}_\tau)$ where

$$T' = T + T' + \mathcal{O}(T + T')$$

and

$$\epsilon(\ell) = \epsilon'(\ell) + 32(q_1 + 2q_2 + 2q_3 + q_4 + 2q_5)(n+1)\epsilon''(\ell)$$

Proof: Suppose that there exists an adversary $\mathcal{A}$ that can break the CCA security of our IBSDDS II scheme with advantage $Adv_{\mathcal{A}} > \epsilon'(\ell) + 32(q_1 + 2q_2 + 2q_3 + q_4 + 2q_5)(n+1)\epsilon''(\ell)$, we can construct an algorithm $\mathcal{B}$ that can use $\mathcal{A}$ to forge a signature or break the DBDH assumption as follows. The challenger generates the bilinear group $\mathcal{GG}(1^\ell) \to (e, p, \mathbb{G}, \mathbb{G}_\tau)$ and chooses a generator $g \xleftarrow{R} \mathbb{G}$. 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 $\mathcal{B}$. Otherwise, he sends $(A, B, C, Z) = (g^a, g^b, g^c, e(g, g)^z)$ to $\mathcal{B}$, where $z \xleftarrow{R} \mathbb{Z}_p$. The algorithm $\mathcal{B}$ will outputs his guess μ' on μ.

Setup. The algorithm $\mathcal{B}$ sets $\sigma = 4(q_1+2q_2+2q_3+q_4+2q_5)$ and chooses an integer $\nu \xleftarrow{R} [n]$. It uniformly selects two integrity vectors $\Pi = \{\pi_1, \pi_2, \cdots, \pi_n\}$ and $\Phi = \{\phi_1, \phi_2, \cdots, \phi_n\}$, where $\pi_i \xleftarrow{R} [\sigma - 1]$ and $\phi_i \xleftarrow{R} \mathbb{Z}_p$ for $i = 1, 2, \cdots, n$. It choose $\pi_0 \xleftarrow{R} [\sigma - 1]$ and $\phi_0 \xleftarrow{R} \mathbb{Z}_p$. Then, the algorithm $\mathcal{B}$ defines three functions:

$$P(ID) = (p - \sigma\nu) + \pi_0 + \sum_{i \in \mathcal{I}} \pi_i,$$

$$Q(ID) = \phi_0 + \sum_{i \in \mathcal{I}} \phi_i$$

and

$$R(ID) = \begin{cases} 0, & \text{if } \pi_0 + \sum_{i \in \mathcal{I}} \pi_i \equiv 0 \pmod{\sigma} \\ 1, & \text{if } \pi_0 + \sum_{i \in \mathcal{I}} \pi_i \not\equiv 0 \pmod{\sigma} \end{cases}$$

$\mathcal{B}$ sets $g_1 = A$, $\eta = B$, $\mathfrak{g} = g^\theta$, $g_2 = A^\theta$, $u_0 = \eta^{p - \sigma\nu + \pi_0} g^{\phi_0}$ and $u_i = \eta^{\pi_i} g^{\phi_i}$, where $\theta \xleftarrow{R} \mathbb{Z}_p$. It chooses $\mathfrak{h} \xleftarrow{R} \mathbb{G}$ and an one-time signature scheme $\mathcal{SG}(1^\ell) \to (\mathsf{SKeyGen}, \mathsf{Sign}, \mathsf{Verify})$. The public parameters are $(e, p, \mathbb{G}, \mathbb{G}_\tau, g, h, \eta, \mathfrak{g}, \mathfrak{h}, u_0, g_1, g_2, \mathbb{U}, \mathsf{Sign}, \mathsf{Verify})$ and the master secret key is $\eta^a = g^{ab}$. The distribution of these parameters is identical to those in the real protocol.

Phase 1.

1) Secret Key Query. $\mathcal{A}$ can query secret key for an identity ID. $\mathcal{B}$ checks $R(ID) \stackrel{?}{=} 1$. If $R(ID) = 0$, $\mathcal{B}$ aborts the simulation and outputs his guess μ' randomly. If $R(ID) = 1$, $\mathcal{B}$ generates a secret key for ID using (1), (2) and (3). $\mathcal{B}$ responds $\mathcal{A}$ with $SK_{ID} = (K_{ID,1}, K_{ID,2}, K_{ID,3})$.

2) Permission Query. $\mathcal{A}$ can query permission on (ID, ID', C_2). $\mathcal{B}$ checks whether $R(ID) = 1$ and $R(ID') = 1$.

   a) If those hold, $\mathcal{B}$ computes an access permission using (4), (5), (6), (7) and (8). $\mathcal{B}$ responds $\mathcal{A}$ with $(D_1, D_2, D_3, K_{ID,2})$.

   b) Otherwise, $\mathcal{B}$ aborts the simulation and outputs his guess μ' randomly.

3) Re-encryption Query. The adversary $\mathcal{A}$ can query on (ID, ID', CT), where $CT = (C_1, C_2, C_3, C_4, \sigma, vk)$. $\mathcal{B}$ check whether he has created an access permission for (ID, ID', C_2). If he has not created an access permission, he create an access permission as above to obtain $(D_1, D_2, D_3, K_{ID,2})$ and computes $C_1' = D_2 \cdot C_1, C_2' = C_2, C_3' = C_3, C_4' = C_4, C_5' = D_1, C_6' = D_3, C_7' = D_3, \sigma' = \sigma, vk' = vk$. $\mathcal{B}$ responds with $CT' = (C_1', C_2', C_3', C_4', C_5', C_6', C_7', \sigma', vk')$.

4) Owner Decryption Query. The adversary $\mathcal{A}$ can query owner decryption on (ID, CT), where $CT =$

Setup. This algorithm takes as input the 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 $g, h, \eta, \mathfrak{g}$ and $\mathfrak{h}$ be the generators of $\mathbb{G}$, $u_0 \overset{R}{\leftarrow} \mathbb{G}$ and $\mathbb{U} = (u_1, u_2, \cdots, u_n)$ where $u_i \overset{R}{\leftarrow} \mathbb{G}$ for $i = 1, 2, \cdots, n$. It chooses $\alpha \overset{R}{\leftarrow} \mathbb{Z}_p$ and sets $g_1 = g^\alpha$ and $g_2 = \mathfrak{g}^\alpha$. It generates an one-time signature scheme $\mathcal{SG}(1^\ell) \rightarrow (\mathsf{SKeyGen}, \mathsf{Sign}, \mathsf{Verify})$, where $\mathsf{SKeyGen}(1^\ell) \rightarrow (sk, vk)$. Let $H : vk \rightarrow \mathbb{Z}_p$ be a hash function. The public parameters are $(e, p, \mathbb{G}, \mathbb{G}_\tau, g, h, \eta, \mathfrak{g}, \mathfrak{h}, u_0, g_1, g_2, \mathbb{U}, H, \mathsf{Sign}, \mathsf{Verify})$ and the master secret keys is η^α.

KeyGen. Let ID denote an identity which is an n bit string, ID_i be the ith bit of ID and $\mathcal{I}$ be the set which consists of all the index i with $ID_i = 1$. This algorithm takes as input the master secret key η^α and the user's identity ID, and computes

$$K_{ID,1} = \eta^\alpha (u_0 \prod_{i \in \mathcal{I}} u_i)^{r_{ID}},\ K_{ID,2} = g^{r_{ID}} \text{ and } K_{ID,3} = \mathfrak{g}^{r_{ID}}.$$

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

$$e(K_{ID,1}, g) \overset{?}{=} e(\eta, g_1) \cdot e((u_0 \prod_{i \in \mathcal{I}} u_i), K_{ID,2}) \text{ and } e(K_{ID,2}, \mathfrak{g}) \overset{?}{=} e(g, K_{ID,3}).$$

Encryption. Suppose that there are k messages $M_i \in \{M_1, M_2, \cdots, M_k\}$. To encrypt M_i, the owner runs $\mathsf{SKeyGen}(1^\ell) \rightarrow (sk, vk)$, chooses $s_i \overset{R}{\leftarrow} \mathbb{Z}_p$ and computes

$$C_{i,1} = M_i \cdot e(g_1, \eta)^{s_i},\ C_{i,2} = g^{s_i},\ C_{i,3} = (u_0 \prod_{i \in \mathcal{I}} u_i)^{s_i},\ C_{i,4} = (g^{H(vk)} \mathfrak{g})^{s_i}$$

and

$$\sigma_i = \mathsf{Sign}(sk, C_{i,2}, C_{i,3}, C_{i,4})$$

for $i = 1, 2, \cdots, k$. The ciphertext for the message M_i is $CT_i = (C_{i,1}, C_{i,2}, C_{i,3}, C_{i,4}, \sigma_i, vk)$. the owner sends $\{CT_1, CT_2, \cdots, CT_k\}$ to the proxy servers. The proxy servers validate the ciphertexts by checking

$$\sigma_i \overset{?}{=} \mathsf{Verify}(vk, C_{i,2}, C_{i,3}, C_{i,4}),\ e((u_0 \prod_{i \in \mathcal{I}} u_i), C_{i,2}) \overset{?}{=} e(C_{i,3}, g) \text{ and } e(g, C_{i,4}) \overset{?}{=} e(C_{i,2}, (g^{H(vk)} \mathfrak{g}))$$

for $i = 1, 2, \cdots, k$. If the equations hold, the proxy servers store the ciphertext $CT_i = (C_{i,1}, C_{i,2}, C_{i,3}, C_{i,4}, \sigma_i, vk)$ for the owner. Otherwise, the proxy servers reject the ciphtertexts.

Query. If a receiver with identity ID' wants to access the message CT_i, he chooses $t \overset{R}{\leftarrow} \mathbb{Z}_p$, and computes $K'_{ID',1} = K_{ID',1} \mathfrak{h}^t$ and $\Gamma = \mathfrak{g}^t$. He sends $(ID', K'_{ID',1}, K_{ID',3}, \Gamma)$ to the proxy server. Then, the proxy server redirects $(ID', K'_{ID',1}, K_{ID',3}, \Gamma, C_{i,2})$ to the owner.

Permission. The owner checks whether the receiver has been verified by the PKG by

$$e(K'_{ID',1}, \mathfrak{g}) \overset{?}{=} e(\eta, g_2) \cdot e((u_0 \prod_{i \in \mathcal{I}'} u_i), K_{ID',3}) \cdot e(\mathfrak{h}, \Gamma).$$

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

$$D_1 = \frac{K_{ID,1}}{K'_{ID',1} \cdot \Gamma^\rho} \cdot (u_0 \prod_{i \in \mathcal{I}'} u_i)^\beta,\ D_2 = e(C_{i,2}, (u_0 \prod_{i \in \mathcal{I}'} u_i))^\beta \text{ and } D_3 = \mathfrak{g}^\rho.$$

The owner sends $(D_1, D_2, D_3, K_{ID,2})$ to the proxy server, where $(D_1, D_2, D_3, K_{ID,2})$ is a re-encryption key.

Re-encryption. Receiving $(D_1, D_2, D_3, K_{ID,2})$ from the owner, the proxy server computes the re-encrypted ciphterxt as

$$C'_{i,1} = D_2 \cdot C_{i,1},\ C'_{i,2} = C_{i,2},\ C'_{i,3} = C_{i,3},\ C'_{i,4} = C_{i,4},\ C'_{i,5} = D_1,\ C'_{i,6} = D_3,\ C'_{i,7} = K_{ID,2} \text{ and } \sigma'_i = \sigma_i.$$

The proxy server responds the receiver with $CT'_i = (C'_{i,1}, C'_{i,2}, C'_{i,3}, C'_{i,4}, C'_{i,5}, C'_{i,6}, C'_{i,7}, \sigma'_i, vk)$.

Decryption.

1) To decrypt the ciphertext $CT_i = (C_{i,1}, C_{i,2}, C_{i,3}, \sigma_i, vk)$, the owner O checks $\sigma_i \stackrel{?}{=} \mathsf{Verify}(vk, C_{i,2}, C_{i,3}, C_{i,4})$ and computes
$$M_i = C_{i,1} \cdot \frac{e(K_{ID,2}, C_{i,3})}{e(K_{ID,1}, C_{i,2})}.$$
2) To decrypt the ciphertext $CT'_i = (C'_{i,1}, C'_{i,2}, C'_{i,3}, C'_{i,5}, C'_{i,6}, C'_{i,7}, \sigma'_i, vk)$, the receiver R checks $\sigma'_i \stackrel{?}{=} \mathsf{Verify}(vk, C'_{i,2}, C'_{i,3}, C'_{i,4})$ and computes $K_1 = K'_{ID',1} \cdot C'^{t}_{i,6} \cdot C'_{i,5}$. Then, he can compute
$$M_i = C'_{i,1} \cdot \frac{e(C'_{i,7}, C'_{i,3})}{e(K_1, C'_{i,2})}.$$

Fig. 3: Identity-Based Distributed Data Storage II

$(C_1, C_2, C_3, C_4, \sigma, vk)$ is a ciphertext for the identity ID. $\mathcal{B}$ check $R(ID) \stackrel{?}{=} 0$.

a) If $R(ID) = 0$, $\mathcal{B}$ aborts and outputs his guess μ' randomly.

b) If $R(ID) \neq 0$, $\mathcal{B}$ check the signature $\sigma \stackrel{?}{=} \mathsf{Verify}(vk, C_2, C_3, C_4)$. If the equation holds, $\mathcal{B}$ generates a secret key K_{ID} for ID as above and responds with $C_1 \cdot \frac{e(K_{ID,2}, C_3)}{e(K_{ID,1}, C_2)}$.

5) Receiver Decryption Query. The adversary $\mathcal{A}$ can query receiver decryption on (ID, ID', CT). $\mathcal{B}$ checks whether he has created secret keys for ID and ID', an access permission for (ID, ID', C_2) and a re-encryption ciphertext CT'. If he has not done these, he creates secret keys, an access permission and a re-encryption as in the secret key query, permission query and re-encryption query to obtain $(SK_{ID}, SK_{ID'})$, $(D_1, D_2, D_3, K_{ID,2})$ and CT'. Then, $\mathcal{B}$ computes K_1 as above and responds with $C'_1 \cdot \frac{e(C'_7, C_3)}{e(K_1, C'_2)}$.

Challenge. The adversary $\mathcal{A}$ submits an identity ID^* and two messages M_0 and M_1 with the equal length. $\mathcal{B}$ checks $R(ID) \stackrel{?}{=} 0$.

1) If $R(ID) = 1$, $\mathcal{B}$ aborts and outputs his guess μ' randomly.
2) If $R(ID) = 0$, $\mathcal{B}$ flips an unbiased coin with $\{0, 1\}$ and obtains $\omega \in \{0, 1\}$. The challenger runs $\mathsf{SKeyGen}(1^\ell) \rightarrow (sk^*, vk^*)$ and computes $C_1^* = M_\omega \cdot Z$, $C_2^* = C = g^c$, $C_3^* = C^{Q(ID^*)} = (u_0 \prod_{i \in \mathcal{I}^*} u_i)^c$, $C_4^* = C^{H(vk^*)+\theta}$ and $\sigma^* = \mathsf{Sign}(sk^*, C_2^*, C_3^*, C_4^*)$. $\mathcal{B}$ sends the ciphertext $CT^* = (C_1^*, C_2^*, C_3^*, C_4^*, \sigma^*, vk^*)$ to $\mathcal{A}$.

Phase 2. Phase 1. is repeated with the following restricts.

1) Secret key Query. The adversary $\mathcal{A}$ cannot query secret key for ID^*.
2) Permission Query. The adversary $\mathcal{A}$ cannot query permission on (ID^*, ID, C_2^*) and secret key for ID.
3) Re-encryption Query. The adversary $\mathcal{A}$ cannot query re-encryption on (ID^*, ID, CT^*), permission on (ID^*, ID, C_2) and secret key for ID.
4) Owner Decryption Query. The adversary $\mathcal{A}$ cannot query the owner decryption algorithm on (ID^*, CT^*).
5) Receiver Decryption Query. The adversary $\mathcal{A}$ cannot query re-encryption on (ID^*, ID, CT^*) and the receiver decryption algorithm on $(ID, \tilde{CT}^*)$, where $\tilde{CT}^*$ is the re-encrypted ciphertext of CT^*.

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

As shown above, the public parameters and the secret keys created in the simulation paradigm are identical to those created in the real protocol. The algorithm $\mathcal{B}$ does not abort the simulation if and only if the secret keys can be generated correctly, $R(ID^*) = 0$ and the signatures in the ciphertext are valid. In q_1 secret key queries, q_2 permission queries, q_3 re-encryption queries, q_4 owner decryption queries and q_5 receiver decryption queries, $\mathcal{B}$ needs to create at most $q_1 + 2q_2 + 2q_3 + q_4 + 2q_5$ secret keys.

Now, we bound the probability with which $\mathcal{B}$ can break the DBDH assumption. This bound is computed using the method in [46]. If $\mu = 1$, $\mathcal{A}$ cannot obtain anything about ω. Hence, $\mathcal{A}$ can output $\omega' \neq \omega$ with no advantage, namely, $\Pr[\omega' \neq \omega | \mu = 1] = \frac{1}{2}$. Since $\mathcal{B}$ outputs $\mu' = 1$ when $\omega' \neq \omega$, we have $\Pr[\mu' = \mu | \mu = 1] = \frac{1}{2}$. If $\mu = 0$, $\mathcal{A}$ can output $\omega' = \omega$ with the advantage at least $\epsilon(\ell)$, namely $\Pr[\omega' = \omega | \mu = 0] \geq \frac{1}{2} + \epsilon(\ell)$. If $\mathcal{B}$ outputs $\mu' = 0$ when $\omega' = \omega$, we have

$$\begin{aligned} \Pr[\mu' = \mu | \mu = 0] - \frac{1}{2} &\geq Adv_{\mathcal{A}} - \Pr[abort] \\ &\geq 32(q_1 + 2q_2 + 2q_3 + q_4 + 2q_5) \\ &\quad (n+1)\epsilon''(\ell) + \epsilon'(\ell) - \Pr[abort] \end{aligned}$$

where $\Pr[abort]$ is the probability with which $\mathcal{B}$ aborts the simulation. The first inequality is from the case $Z = e(g, g)^{abc}$, so the simulation is performed correctly if $\mathcal{B}$ does not abort. Hence, $\mathcal{B}$ can solve the DBDH

assumption with the advantage at least

$$\frac{\epsilon(\ell)}{32(q_1 + 2q_2 + 2q_3 + q_4 + 2q_5)(n+1)} \geq \epsilon''(\ell).$$

It remains to bound the probability with which $\mathcal{B}$ aborts the simulation as a result of $\mathcal{A}$'s decryption queries. We claim that $\Pr[abort] < \epsilon'(\ell)$. Otherwise, a forged signature can be computed with advantage at least $\epsilon'(\ell)$. Briefly, receiving the challenged signature key sk^* in the simulation, $\mathcal{A}$ causes an abort by submitting a decryption query which includes a forged signature of one ciphertext under sk^*. Therefore, $\mathcal{B}$ can use the forged signature to break the existential unforgability of the one-time signature. Notably, $\mathcal{A}$ can only query one signature for the challenged ciphertext. Hence, we have $\Pr[abort] < \epsilon'(\ell)$.

So, $\mathcal{B}$ can break the decisional bilinear Diffie-Hellman assumption with advantage more than $\frac{\epsilon(\ell)}{32(q_1+2q_2+2q_3+q_4+2q_5)(n+1)}$. This finishes our proof. □

5 Conclusion

Distributed data storage schemes provide the users with convenience to outsource their files to untrusted proxy servers. Identity-based secure distributed data storage (IBSDDS) schemes are a special kind of distributed data storage schemes where users are identified by their identities and can communicate without the need of verifying the public key certificates. In this paper, we proposed two new IBSDDS schemes in standard model where, for one query, the receiver can only access one file, instead of all files. Furthermore, the access permission can be made by the owner, instead of the trusted party. Notably, our schemes are secure against the collusion attacks. The first scheme is CPA secure, while the second one is CCA secure.

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] H. Hacigümüs, B. R. Iyer, C. Li, and S. Mehrotra, "Executing SQL over encrypted data in the database-service-provider model," in *Proceedings: SIGMOD Conference - SIGMOD'02* (M. J. Franklin, B. Moon, and A. Ailamaki, eds.), vol. 2002, (Madison, Wisconsin, USA), pp. 216–227, ACM, Jun. 2002.

[2] L. Bouganim and P. Pucheral, "Chip-secured data access: Confidential data on untrusted servers," in *Proc. International Conference on Very Large Data Bases - VLDB'02*, (Hong Kong, China), pp. 131–142, Morgan Kaufmann, Aug. 2002.

[3] U. Maheshwari, R. Vingralek, and W. Shapiro, "How to build a trusted database system on untrusted storage," in *Proc. Symposium on Operating System Design and Implementation - OSDI'00*, (San Diego, California, USA), pp. 135–150, USENIX, Oct. 2000.

[4] A. Ivan and Y. Dodis, "Proxy cryptography revisited," in *Proc. Network and Distributed System Security Symposium - NDSS'03*, (San Diego, California, USA), pp. 1–20, The Internet Society, Feb. 2003.

[5] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," in *Proc. Network and Distributed System Security Symposium - NDSS'05*, (San Diego, California, USA), pp. 1–15, The Internet Society, Feb. 2005.

[6] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," *ACM Transactions on Information and System Security*, vol. 9, no. 1, pp. 1–30, 2006.

[7] S. D. C. di Vimercati, S. Foresti, S. Paraboschi, G. Pelosi, and P. Samarati, "Effficient and private access to outsourced data," in *Proc. International Conference on Distributed Computing Systems - ICDCS'11*, (Minneapolis, Minnesota, USA), pp. 710–719, IEEE, Jun. 2011.

[8] H.-Y. Lin and W.-G. Tzeng, "A secure erasure code-based cloud storage system with secure data forwarding," *IEEE Transactions on Parallel and Distributed Systems*, Digital Object Indentifier 10.1109/TPDS.2011.252 2012.

[9] H. Shacham and B. Waters, "Compact proofs of retrievability," in *Proc. Advances in Cryptology - ASIACRYPT'08* (J. Pieprzyk, ed.), vol. 5350 of *Lecture Notes in Computer Science*, (Melbourne, Australia), pp. 90–107, Springer, Dec. 2008.

[10] A. Juels and B. S. K. Jr., "PORs: Proofs of retrievability for large files," in *Proceedings: ACM Conference on Computer and Communications Security - CCS'07* (P. Ning, S. D. C. di Vimercati, and P. F. Syverson, eds.), (Alexandria, Virginia, USA), pp. 584–597, ACM, Oct. 2007.

[11] Y. Dodis1, S. Vadhan, and D. Wichs, "Proofs of retrievability via hardness amplification," in *Proc. Theory of Cryptography Conference - TCC'09* (O. Reingold, ed.), vol. 5444 of *Lecture Notes in Computer Science*, (San Francisco, CA, USA), pp. 109–127, Springer, Mar. 2009.

[12] K. D. Bowers, A. Juels, and A. Oprea, "Proofs of retrievability: Theory and implementation," in *Proc. ACM Cloud Computing Security Workshop - CCSW'09*, (Chicago, Illinois, USA), pp. 43–53, ACM, Nov. 13 2009.

[13] G. Ateniese, S. Kamara, and J. Katz, "Proofs of storage from homomorphic identification protocols," in *Proc. Advances in Cryptology - ASIACRYPT'09* (M. Matsui, ed.), vol. 5912 of *Lecture Notes in Computer Science*, (Tokyo, Japan), pp. 319–333, Springer, Dec. 2009.

[14] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in *Proc. ACM Conference on Computer and Communications Security - CCS'07* (P. Ning, S. D. C. di Vimercati, and P. F. Syverson, eds.), (Alexandria, Virginia, USA), pp. 598–610, ACM, Oct. 2007.

[15] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in *Proc. International conference on Security and privacy in communication netowrks - SecureComm'08*, (Istanbul, Turkey), Sep., ACM, 2008.

[16] C. C. Erway, A. Küpcü, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in *Proc. ACM Conference on Computer and Communications Security - CCS'09* (E. Al-Shaer, S. Jha, and A. D. Keromytis, eds.), (Chicago, Illinois, USA), pp. 213–222, ACM, Nov. 2009.

[17] B. Carbunar and R. Sion, "Toward private joins on outsourced data," *IEEE Transactions on Knowlege and Data Engineering*, vol. 9, no. 24, pp. 1699–1710, 2012.

[18] J. Hur, "Improving security and efficiency in attribute-based data sharing," *IEEE Transactions on Knowlege and Data Engineering*, p. Digital Object Indentifier 10.1109/TKDE.2011.78.

[19] J. Hur and D. K. Noh, "Attribute-based access control with efficient revocation in data outsourcing systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 7, pp. 1214–1221, 2011.

[20] M. L. Yiu, G. Ghinita, C. S. Jensen, and P. Kalnis, "Enabling search services on outsourced private spatial data," *The VLDB Journal*, vol. 19, no. 3, pp. 363–384, 2010.

[21] P. Samarati and S. D. C. di Vimercati, "Data protection in outsourcing scenarios: Issues and directions," in *Proc. ACM Symposium on Information, Computer and Communications Security - ASIACCS'10* (D. Feng, D. A. Basin, and P. Liu, eds.), (Beijing, China), pp. 1–14, ACM, Apr. 2010.

[22] V. Kher and Y. Kim, "Securing distributed storage: Challenges, techniques, and systems," in *Proc. ACM Workshop On Storage Security And Survivability - StorageSS'05* (V. Atluri, P. Samarati, W. Yurcik, L. Brumbaugh, and Y. Zhou, eds.), (Fairfax, VA, USA), pp. 9–25, ACM, Nov. 2005.

[23] S. Jiang, X. Zhang, S. Liang, and K. Davis, "Improving networked file system performance using a locality-aware cooperative cache protocol," *IEEE Transactions on Computers*, vol. 59, no. 11, pp. 1508–1519, 2010.

[24] R. Sandberg, D. Goldberg, S. Kleiman, D. Walsh, and B. Lyon, "Design and implementation of the sun network file system," in *Proc. USENIX Technical Conference - USENIX'85*, (Portland), pp. 119–130, USENIX, 1985.

[25] M. Satyanarayanan, "Scalable, secure, and highly available distributed file access," *IEEE Computer*, vol. 23, no. 5, pp. 9–21, 1990.

[26] S. Forrest, S. A. Hofmeyr, and A. Somayaji, "Sense of self for unix processes," in *Proc. IEEE Symposium on Security and Privacy-S&P'96*, (Oakland, CA, USA), pp. 120–128, IEEE, May 1996.

[27] A. G. Pennington, J. L. Griffin, J. S. Bucy, J. D. Strunk, and G. R. Ganger, "Storage-based intrusion detection," *ACM Transactions on Information and System Security*, vol. 13, no. 4, pp. 30:1–30:27, 2010.

[28] M. Banikazemi, D. Poff, and B. Abali, "Storage-based intrusion detection for storage area networks (SANs)," in *Proc. Conference on Mass Storage Systems and Technologies - MSST'05*, (Monterey, CA, USA), pp. 118–127, IEEE, Apr. 2005.

[29] A. G. Pennington, J. D. Strunk, J. L. Griffin, C. A. Soules, G. R. Goodson, and G. R. Ganger, "Storage-based intrusion detection: Watching storage activity for suspicious behavior," in *Proc. the USENIX Security Symposium - USENIX'03*, (Washington, D.C., USA), pp. 137–152, USENIX, Aug. 2003.

[30] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, "Plutus: Scalable secure file sharing on untrusted storage," in *Proc. Conference on File and Storage Technologies -FAST'03*, (San Francisco, CA, USA), pp. 29–42, USENIX, Mar. 2003.

[31] B. Blanchet and A. Chaudhuri, "Automatic formal analysis of a protocol for secure file sharing on untrusted storage," in *Proc. Symposium on Security and Privacy - S & P'08)*, (Oakland, California, USA), pp. 417–431, IEEE, May 2008.

[32] M. Blaze, "A cryptographic file system for unix," in *Proc. ACM Conference on Computer and Communications Security - CCS'93*, (Fairfax, Virginia, USA), pp. 9–16, ACM, Nov. 3-5 1993.

[33] E.-J. Goh, H. Shacham, N. Modadugu, and D. Boneh, "SiRiUS: Securing remote untrusted storage," in *Proc. Network and Distributed System Security Symposium - NDSS'03*, (San Diego, California, USA), pp. 1–15, The Internet Society, Feb. 2003.

[34] M. Mambo and E. Okamoto, "Proxy cryptosystems: Delegation of the power to decyrpt ciphertexts," *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, vol. E80A, no. 1, pp. 54–63, 1997.

[35] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *Proc. Advances in Cryptology - EUROCRYPT'98* (K. Nyberg, ed.), vol. 1403 of *Lecture Notes in Computer Science*, (Espoo, Finland), pp. 127–144, Springer, May 1998.

[36] A. Shamir, "Identity-based cryptosystems and signature scheme," in *Proc. Advances in Cryptology - CRYPTO'84* (G. R. Blakley and D. Chaum, eds.), vol. 196 of *Lecture Notes in Computer Science*, (Santa Barbara, California, USA), pp. 47–53, Springer, Aug. 1984.

[37] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Proc. Advances in Cryptology - CRYPTO'01* (J. Kilian, ed.), vol. 2139 of *Lecture Notes in Computer Science*, (Santa Barbara, California, USA), pp. 213–229, Springer, Aug. 2001.

[38] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in *Proc. Applied Cryptography and Network Security - ACNS'07* (J. Katz and M. Yung, eds.), vol. 4521 of *Lecture Notes in Computer Science*, (Zhuhai, China), pp. 288–306, Springer, Jun. 2007.

[39] C.-K. Chu and W.-G. Tzeng, "Identity-based proxy re-encryption without random oracles," in *Proc. Information Security Conference - ISC'07* (J. A. Garay, A. K. Lenstra, M. Mambo, and R. Peralta, eds.), vol. 4779 of *Lecture Notes in Computer Science*, (Valparaso, Chile), pp. 189–202, Springer, Oct. 2007.

[40] Q. Tang, P. Hartel, , and W. Jonker, "Inter-domain identity-based proxy re-encryption," in *Proc. Information Security and Cryptology - Inscrypt'08* (M. Yung, P. Liu, and D. Lin, eds.), vol. 5487 of *Lecture Notes in Computer Science*, (Beijing, China), pp. 332–347, Springer, Dec. 2008.

[41] T. Matsuo, "Proxy re-encryption systems for identity-based encryption," in *Proc. Pairing-Based Cryptography - Pairing'07* (T. Takagi, T. Okamoto, E. Okamoto, and T. Okamoto, eds.), vol. 4575 of *Lecture Notes in Computer Science*, (Tokyo, Japan), pp. 247–267, Springer, Jul. 2007.

[42] L. Wang, L. Wang, M. Mambo, and E. Okamoto, "New identity-based proxy re-encryption schemes to prevent collusion attacks," in *Proc. Pairing-Based Cryptography - Pairing'10* (M. Joye, A. Miyaji, and A. Otsuka, eds.), vol. 6487 of *Lecture Notes in Computer Science*, (Yamanaka Hot Spring, Japan), pp. 327–346, Springer, Dec. 2010.

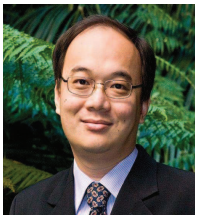
[43] L. Wang, L. Wang, M. Mambo, and E. Okamoto, "Identity-based proxy cryptosystems with revocability and hierarchical confidentialities," in *Proc. International Conference on Information and Communications Security - ICICS'10* (M. Soriano, S. Qing, and J. López, eds.), vol. 6476 of *Lecture Notes in Computer Science*, (Barcelona, Spain), pp. 383–440, Springer, Dec. 2010.

[44] B. Waters, "Efficient identity-based encryption without random oracles," in *Proc. Advances in Cryptology - EUROCRYPT'05* (R. Cramer, ed.), vol. 3494 of *Lecture Notes in Computer Science*, (Aarhus, Denmark), pp. 114–127, Springer, May 2005.

[45] R. Canetti, S. Halevi, and J. Katz, "Chosen-ciphertext security from identity-based encryption," in *Proc. Advances in Cryptology - EUROCRYPT'04* (C. Cachin and J. Camenisch, eds.), vol. 3027 of *Lecture Notes in Computer Science*, (Interlaken, Switzerland), pp. 207–222, Springer, May 2004.

[46] D. Boneh, C. Gentry, and B. Waters, "Collusion resistant broadcast encryption with short ciphertexts and private keys," in *Proc. Advances in Cryptology - Crypto'05* (V. Shoup, ed.), vol. 3621 of *Lecture Notes in Computer Science*, (Santa Barbara, California, USA), pp. 258–275, Springer, Aug. 2005.

Jinguang Han is currently a PhD candidate in School of Computer Science and Software Engineering, University of Wollongong, Australia. He is also a lecturer in College of Sciences, Hohai University, China. His research interests include applied cryptography, identity management, access control and data outsourcing. He is a student member of the IEEE since 2010.



Willy Susilo received a Ph.D. in Computer Science from University of Wollongong, Australia. He is a Professor at the School of Computer Science and Software Engineering and the co-director of Centre for Computer and Information Security Research (CCISR) at the University of Wollongong. He is currently holding the prestigious ARC Future Fellow awarded by the Australian Research Council (ARC). His main research interests include cryptography and information security. His main contribution is in the area of digital signature schemes. He has served as a program committee member in dozens of international conferences. He has published numerous publications in the area of digital signature schemes and encryption schemes. He is a senior member of IEEE since 2001.

Yi Mu received his PhD from the Australian National University in 1994. He currently is a professor, Head of School of Computer Science and Software Engineering and the co-director of Centre for Computer and Information Security Research at University of Wollongong, Australia. His current research interests include network security, computer security, and cryptography. Yi Mu is the editor-in-chief of International Journal of Applied Cryptography and serves as associate editor for nine other international journals. He is a senior member of the IEEE and a member of the IACR. He is a senior member of IEEE since 2000.