A Key-insulated Proxy Re-encryption Scheme for Data Sharing in a Cloud Environment

Yilei Wang[1], Dongjie Yan[1], Fagen Li[1] and Hu Xiong[1]
(Corresponding author: Hu Xiong)

School of Information and Software Engineering, University of Electronic Science and Technology of China[1]
No. 4, North Jianshe Road, Chenghua District, chengdu, Sichuan 610054, China
(Email: xionghu.uestc@gmail.com)

(Received Apr. 16, 2016; revised and accepted June 28 & July 19, 2016)

Abstract

Proxy re-encryption (PRE) enables a semi-trusted proxy to delegate the decryption right by re-encrypting the ciphertext under the delegator's public key to an encryption under the public key of delegatee. Fueled by the translation ability, PRE is regarded as a promising candidate to secure data sharing in a cloud environment. However, the security of the PRE will be totally destroyed in case the secret key of the delegator or the delegatee has been exposed. Despite the key exposure seems inevitable, the PRE scheme with resistance against secret key leakage has never been presented before. To deal with this intractable problem, we propose a key-insulated proxy re-encryption (KIPRE) scheme by incorporating the mechanisms of PRE and key-insulated cryptosystem. In the proposed scheme, the lifetime of the secret key associated with the user, i.e., the delegator or the delegatee, has been divided into several periods. In each time period, the user can interact with his/her physically-secure but computation-limited helper to update his/her temporary secret key. On the contrary, the public keys of the users remained unchanged during the whole lifetime of the system. We then apply our KIPRE scheme to construct a practical solution to the problem of sharing sensitive information in public clouds with resilience to the key exposure. The performance evaluation and the security analysis demonstrate that our scheme is efficient and practical.

Keywords: Cloud Environment; Key Insulation; KIPRE; PRE; Secret Key Exposure.

1 Introduction

In 1998, Blaze et al. [2] introduced the conception called *atomic proxy re-encryption*, in which a semi-trusted proxy is deployed to transform the ciphertext encapsulated under Alice's public key into the one that can be decrypted with Bob's secret key.

In such an environment, Alice can delegate her decryption right to Bob without making him access her secret key (That is to say, Alice and Bob respectively perform as the the delegator and the delegatee during the communication), meanwhile, the proxy is unable to see the underlying plaintext.

In the present decade, proxy re-encryption (PRE) has attracted much attention of many researchers [3, 4, 7, 18, 22, 27] and has been introduced to be applied in a number of interesting environments such as secure file systems [1, 24], multicast [20], email forwarding [13] and law enforcement [12].

PRE is also deemed as a promising technique to ensure data confidentiality and fine-access control in cloud computing environment [14, 15, 26, 27, 28] which is an emerging computing paradigm drawing extensive attention from both academia and industry. In such an environment, the cloud service provider (CSP) plays the role of the proxy, and the users respectively act as the delegator (Alice) and the delegatee (Bob). Despite that PRE has varies of inspiring applications, the security behind it relies on the basic precondition that the user's secret key in PRE settings is kept insulated from the attacker. That is to say, the security of user's sensitive data will no longer exist if the user's secret key is compromised.

Exposure of a secret key. We consider that it is reasonable for an attacker to compromise the legal user's secret key in a normal asymmetric cryptosystem (including PRE in cloud computing environment), in which a single secret key is bound to a public key or an identity. The secret key is the only necessity for a user to decrypt a ciphertext. In general, it is reposited in either a personal computer or a fully trusted server, and even protected by some naive approach such as a password. This protection method is valid and high-efficient if the computer or the server is absolutely isolated from an opening network. Unfortunately, this assumption is too strong and it is unlikely to be met in reality. (1) Threats from the Internet: the device holding the secret key may suffer from numerous attacks from network hackers with various in-

trusion capacities, meanwhile the key owner may be unaware about it. (2) Physical threats: the computer/server may be accessed by some other ill-disposed users when the original user (i.e. the key owner) leaves without locking it. In these cases, the malicious users or the network attackers can compromise the secret key to gain the access of its owner's personal data stored in the cloud system. Consequently, it is particularly imperative to tackle the compromise of user's secret key in public key settings.

To address the intractable key exposure problem mentioned above, broad research has been made to minimize the damage caused by key exposure rather than to prevent the attacker from getting access to the secret key (because the secret key is used frequently in decryption in public key settings). An effective and practical approach is to utilize key-insulated encryption (KIE) which was introduced by Dodis et al. [5] in 2002. In KIE environment, the core idea is that the user's secret key is made up of two parts, one of which is controlled by the user and the other is evolved by the helper (i.e. a physically-secure but computation-limited device). The system lifetime is divided into several distinct time periods, in which the the user's secret key is diverse from each other. However, the corresponding public key remains unchanged through the whole system lifetime. By utilizing the technic of key-insulation, the user interacts with the physically-secure helper at the beginning of each time period to yield a temporary secret key which is valid in decryption during that time period. In KIE, the key exposure in a certain time period only threatens the security during that time period rather than the others. In other words, KIE captures the forward security and the backward security simultaneously.

Our contribution. Inspired by [1] and [5], we present a new scheme called key-insulated proxy re-encryption (KIPRE) for data sharing in a cloud environment to enjoy the benefits of KIE in PRE settings, using a practical helper. The helper is actually a proper device which satisfies the following conditions: (1) its computation capacity may be limited, because it should be portable enough; and (2) it is physically secure and should not be eavesdropped, i.e. the secret information reserved in it cannot be accessed by any invalid user. In this paper, we introduce key-insulation to the PRE environment to cope with the exposure of user's private keys. More importantly, our scheme, for the first time, addresses the secret key exposure in PRE for data sharing in a cloud environment. Furthermore, we give our concrete construction of KIPRE as well as the security and efficiency analysis of our proposed scheme.

2 Related Work

In this section, we inspect some related works involving proxy re-encryption cryptosystems (which suffers from the exposure of user's secret key) and cryptosystems with key insulation.

2.1 Proxy Re-encryption Cryptosystems

The idea that the decryption right can be delegated from one legal user to another was introduced by Mambo and Okamoto [19] in 1997. Then Blaze et al. [2] introduced the conception of *atomic proxy re-encryption* as well as a concrete scheme which was based on Elgamal system. However, it was unfortunately bidirectional. (i.e. The corrupted proxy could re-encrypt the original ciphertexts from Alice to Bob and vise versa.) With some improvement, Ateniese et al. [1] showed their pairing-based unidirectional proxy re-encryption schemes in 2006. In their schemes, the proxy cannot collude with the delegatee to reveal the delegator's private key and their schemes are semantically secure based on the Decisional Bilinear Diffie-Hellman problem. In 2007, Canetti and Hohenberger [3] firstly defined the security against chosen ciphertext attacks (CCA) in PRE system and gave a concrete construction capturing the security they defined. Since then, varieties of PRE schemes [17, 21, 27] which satisfied different properties were proposed. In particular, Sur et al. [21] introduced the conception of certificateless proxy re-encryption (CL-PRE) as well as its security definitions. Moreover, they also constructed a CCA secure scheme, the security of which was proved in the random oracle model. In 2014, Liu et al. [17] presented a time-based proxy re-encryption scheme (TimePRE) for secure data sharing in a cloud environment, in which a user's access right to the re-encrypted data stored in the proxy could expire automatically by embedding a predetermined time period in user's private key, i.e. the delegatee could be revoked even if the delegator was not online. In the following year, Xu et al. [27] proposed an efficient Conditional Identity-based Broadcast proxy re-encryption scheme (CIBPRE) which was considered appropriate to be applied into secure cloud email system with more advantages than the existing secure email systems.

Despite that there are plenty of schemes which have been proposed with various properties, none of them focus on the exposure of user's secret key. We argue that the message (either the original ciphertext of the re-encrypted ciphertext) transferred in the PRE settings can be decrypted by the attackers if either the delegator's or the delegatee's private key is leaked.

2.2 Cryptosystems with Key Insulation

In 2002, Dodis et al. [5] firstly introduced the notion of key-insulated security as well as the first (t, N)-key-insulated encryption scheme based on any (standard) public key encryption scheme. Then, Hanaoka et al. [10] constructed an unconditionally secure key-insulated encryption scheme. Additionally, they also extended the model of key-insulated encryption (KIE) to dynamic and mutual key-insulated encryption (DMKIE) which could be constructed from broadcast encryption schemes or key pre-distribution schemes. In 2006, Hanaoka et al. [8] pre-

sented a new paradigm called parallel key-insulated encryption scheme (PKIE), in which more than one helper was employed to interact with the user to update the temporary secret key. They tried to address the increasing probability of the key leakage caused by frequent update of the temporary secret key. In [9], Hanaoka et al. firstly proposed a new primitive called one time forward-secure public key encryption (OTFS-PKE), which could be constructed from either ordinary identity-based encryption (IBE) or hierarchical identity-based encryption (HIBE). Then they also introduced how to extend a OTFS-PKE scheme to a parallel key-insulated encryption (PKIE) scheme. Recently, Hong and Sun [11] firstly presented a pairing-free key insulated attribute-based encryption scheme which is high efficient and provably secure. Their scheme combined the advantages of both key insulation and attribute-based encryption. Moreover, they also argued that their scheme was much more suitable to be applied in data sharing network systems, particularly those with limitation in computation such as mobile communication system and wireless sensor networks. The advantage of key insulated mechanism can also be combined into PRE to tackle key exposure problems in PRE settings and it is necessary to propose a KIPRE scheme with efficiency and security.

3 Preliminary

3.1 Mathematical Background

Bilinear Maps. $\mathbb{G}$, $\mathbb{G}_T$ are two groups with the same prime order q. g is the generator of $\mathbb{G}$, and e is a function, e: $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, such that:

1) **Bilinear**: $e(g^a, g^b) = e(g, g)^{ab}$, for $\forall a, b \in \mathbb{Z}_q^*$.
2) **Non-degenerate**: There exists some $g \in \mathbb{G}$ such that $e(g, g) \neq 1$.
3) **Computable**: There exists an efficient algorithm to compute $e(g^a, g^b)$, for $\forall a, b \in \mathbb{Z}_q^*$.

3.2 Assumption

q-weak Decisional Bilinear Diffie-Hellman Inversion (q-wDBDHI) Assumption. we assume the intractability of a variant of the q-Decisional Bilinear Diffie-Hellman Inversion (q-DBDHI) problem [6]. For an algorithm $\mathcal{A}$, define its advantage as $\mathrm{Adv}_{\mathcal{A}}^{q-\mathrm{wDBDHI}}(k) = \left|\Pr[\mathcal{A}(g, g^a, \ldots, g^{(a^q)}, g^b, e(g, g)^{b/a}) = 1] - \Pr[\mathcal{A}(g, g^a, \ldots, g^{(a^q)}, g^b, \Gamma) = 1]\right|$, where $g \leftarrow \mathbb{G}$, $a, b \in \mathbb{Z}_q^*$ and $\Gamma \in \mathbb{G}_T$. We say the q-wDBDHI assumption holds, if for any probabilistic polynomial-time (PPT) algorithm $\mathcal{A}$, $\mathrm{Adv}_{\mathcal{A}}^{q-\mathrm{wDBDHI}}(k)$ is negligible in security parameter k. We argue that the q-wDBDHI assumption is slightly weaker than the q-DBDHI assumption defined in [6], which is to recognize $e(g, g)^{1/a}$ given $(g, g^a, \ldots, g^{(a^q)}) \in \mathbb{G}^{q+1}$ and Dodis and Yampolskiy showed that it was indeed hard in generic groups.

4 System Model

In our system, we assume that the cloud server is a semi-trusted party which may correctly execute the corresponding delegation algorithms with curiosity on the underlying data. It may try to decrypt the ciphertext reserved in the cloud server. Furthermore, we assume that the helper is some proper device which is physically-secure (e.g. it is isolated from the opening network and well protected by its owner.) and may be computation-limited (in order to capture some other properties such as portability). To guarantee the consistency of the time periods among all the entities, we also assume that there is a global time flowing through the whole system lifetime. Actually, a global time is not quite easy to achieve in a cloud computing environment. However, we can utilize the techniques introduced in [16] to achieve this goal. In addition, we also suggest that our scheme is much more appropriate for the cloud environment where a coarse-grained time accuracy for the division of time periods is satisfactory.

Before giving the concrete construction of our proposed scheme, we show an intuition on it and further illustrate our mechanism's framework in Figure 1. In our system, there are four entities which are described as follows:

- Cloud Service Provider (CSP): The CSP maintains the cloud infrastructures including the bandwidth, storage devices and many cloud servers with powerful computation capability. We assume that the storage and the computation ability of the CSP are flexibly extensible. Therefore, it owns high reliability and efficiency far beyond the personal computers. In our system, the CSP mainly provides two kinds of services: data storage and re-encryption. After receiving the encrypted data from the delegator (Alice), the CSP stores the data on the cloud storage devices. After obtaining the re-encryption key sent from Alice, the CSP will correctly execute the re-encryption algorithm to transfer the original ciphertext to the re-encrypted ciphertext and sent it to the delegatee (Bob).
- Delegator (Alice): She is the delegator (or the data owner) and she is responsible for sending the ciphertext encrypted with her own public key to the CSP and generating re-encryption key which will be delivered to the CSP in an appropriate time period.
- Delegatee (Bob): He is the delegatee (the receiver) and he can issue a request for the data Alice outsourced in the cloud. Then the CSP send the re-encrypted data to him. He can decrypt the re-encrypted data utilizing his own secret key in the corresponding time period.
- Physically-Secure Device (Helper): It is a physically-secure but computation-limited device which is deployed to help the system user (i.e. the delegator and the delegatee) to update their secret keys (Each

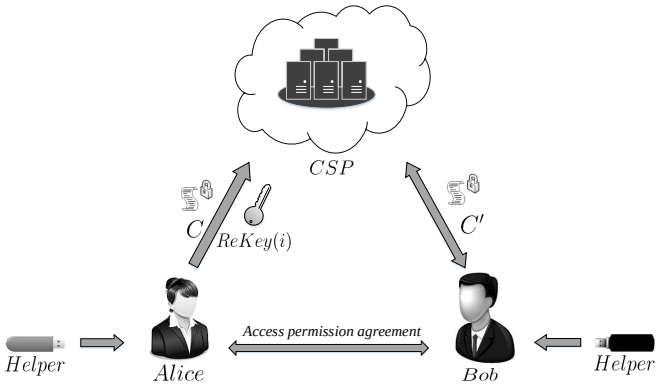

Figure 1: Key insulated proxy re-encryption framework

user owns a distinct temporary secret key in each time period).

On account of the local storage and computation limitation of personal computer, Alice prefers to outsource her data (which is encrypted before uploading) on the remote cloud server. To share her data stored in the CSP without reveal to the public, the system performs as follows:

1) **Upload encrypted data.** Before uploading the outsourced data to the CSP, Alice encrypts the data with her public key and current time period i. Since the decryption requires Alice's secret key which is properly protected by key-insulation mechanism, the data stored on the cloud server is well safeguarded.

2) **Access request.** To get the access permission from Alice, Bob needs to send an access request to Alice. Then Alice decides whether he is allowed to get access to the corresponding data. That is to say, Alice makes access control by herself.

3) **Access permission.** If Alice permits Bob's request, she will respond a permission (may be some material such as a signature representing Alice's confirmation) to Bob and executes the $ReKey(i)$ generation. Otherwise, she refuses Bob's request.

4) **ReKey(i) generation.** In this phase, Alice executes ReKeyGen algorithm to generate the re-encryption key $ReKey(i)$ for time period i and sends it to the CSP.

5) **Re-encrypted data response.** When the CSP receives the $ReKey(i)$ from Alice, it honestly re-encrypts the original ciphertext and sends the re-encrypted ciphertext to Bob. Though the CSP is always curious during the whole system runtime, it cannot get any underlying information except the source address and the destination.

6) **Decryption of the re-encrypted ciphertext.** After receiving the re-encrypted ciphertext from the CSP, Bob can run the Dec algorithm with his own secret key for time period i to obtain the underlying data.

5 Our Construction

In this section, we will give the concrete construction of our proposed KIPRE scheme which involves septuple algorithms: KeyGen, Update*, Update, ReKeyGen, Enc, ReEnc, Dec. Compared to the traditional PRE scheme, our scheme includes two additional algorithms Update* and Update, which make user's secret key evolve with time periods. Therefore, it can correctly capture key insulation in PRE settings and mitigate the damage caused by key exposure. In our system, we consider the two users Alice and Bob as the delegator and the delegatee respectively. The detail of the algorithms mentioned above are described as follows. (Note that the keys corresponding to Bob perform as the same as those of Alice in the algorithms Update* and Update. For simplicity, we omit them.)

- KeyGen: On input the security parameter 1^k, this algorithm randomly selects a prime q. Let $\mathbb{G}, \mathbb{G}_T$ be two groups of the same prime order q and g is a generator of $\mathbb{G}$. For Alice, this algorithm chooses $x^*_{A,0}, \ldots, x^*_{A,t} \leftarrow \mathbb{Z}^*_q$, where $t \in \mathbb{Z}^*_q$ is the total number of the time periods which the system lifetime is divided into. The system public parameters are g, $\mathbb{G}$, $\mathbb{G}_T$, $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, where e is a bilinear map. Alice's public key is $PK_A = \{g^{x^*_{A,0}}, \ldots, g^{x^*_{A,t}}\}$. The master key of Alice's helper is $SK^*_A = (x^*_{A,1}, \ldots, x^*_{A,t})$. The initial key of Alice is $SK_{A,0} = x^*_{A,0}$. Bob's initial keys including PK_B, SK^*_B, $SK_{B,0}$ can be generated in a similar way and we omit it here.

- Update*: At the end of time period $i-1 \in \{0, 1, \ldots, t-1\}$, Alice's helper executes algorithm Update* to generate the helper key $SK'_{A,i} = x'_{A,i}$ for period $i \in \{1, 2, \ldots, t\}$, where $x'_{A,i} = \sum_{j=1}^{t} x^*_{A,j}(i^j - (i-1)^j)$.

- Update: Given the helper key $SK'_{A,i}$ for period i, Alice runs this algorithm to Compute $x_{A,i} = x_{A,i-1} + x'_{A,i} = \sum_{j=0}^{t}(x^*_{A,j}(i-1)^j) + \sum_{j=1}^{t} x^*_{A,j}(i^j - (i-1)^j) = \sum_{j=0}^{t}(x^*_{A,j} \cdot i^j)$. Then output $SK_{A,i} = x_{A,i}$ for period i.

- ReKeyGen: This algorithm takes Bob's public key $PK_B = \{g^{x^*_{B,0}}, \ldots, g^{x^*_{B,t}}\}$ and time period i as input. Compute $rk_{A \xrightarrow{i} B,1} = \prod_{j=0}^{t}(g^{x^*_{B,j}})^{i^j} = g^{x_{B,i}}$. Then Alice uses her secret key $SK_{A,i}$ to compute $rk_{A \xrightarrow{i} B,2} = rk^{1/x_{A,i}}_{A \xrightarrow{i} B,1} = g^{x_{B,i}/x_{A,i}}$ and sends $rk_{A \xrightarrow{i} B,2}$ to the CSP.

- Enc: Given Alice's public key $PK_{A,i}$, this algorithm randomly selects $r \leftarrow \mathbb{Z}^*_q$ and computes the

original ciphertext $C = (C_1, C_2)$, where $C_1 = \left(\prod_{j=0}^{t}(g^{x^*_{A,j}})^{i^j}\right)^r = (g^{\sum_{j=0}^{t}(x^*_{A,j}\cdot i^j)})^r = (g^{x_{A,i}})^r$ and $C_2 = e(g,g)^r \cdot M$.

- ReEnc: On input the re-encryption key $rk_{A\xrightarrow{i}B,2}$ and the original ciphertext (C_1, C_2) for period i, this algorithm computes $C'_1 = e(C_1, rk_{A\xrightarrow{i}B,2}) = e(g^{r x_{A,i}}, g^{x_{B,i}/x_{A,i}})$. Then output the re-encrypted ciphertext as $C' = (C'_1, C_2)$.
- Dec: For a original ciphertext $C = (C_1, C_2)$, Alice can compute $M = \frac{C_2}{e(g,C_1)^{1/x_{A,i}}}$. Receiving the re-encrypted ciphertext $C' = (C'_1, C_2)$, Bob executes the algorithm Dec with input $SK_{B,i}$ to recover the plaintext $M = \frac{C_2}{(C'_1)^{1/x_{B,i}}}$.

6 System Evaluation

6.1 Discussion

Privacy of user's secret key. In our proposed scheme, we import the key-insulated mechanism to protect user's secret key from being exposed. Our construction supports key-insulated security which is captured by key update in each time period with the assistance of the helper, i.e. the secret keys for both previous and following time periods are safe even if the secret key for the current time period is exposed.

Time period consistency. To guarantee the time period consistency among all the related entities in our system, we should pre-defined a global time which can be achieved by utilizing some other techniques such as [16]. We further argue that it is reasonable and available to be achieved, since the time period consistency is the basic condition for all of the previous key-insulated schemes [5, 11, 23, 25] and some other time-based schemes [16, 17].

Time-release delegation. In ReKeyGen of our proposed scheme, we consider the sharing data should be outsourced in the same time period that Bob's data request is issued. That is to say, Alice's data outsourcing and Bob's data requirement are occurred in the same time period. Actually, we argue that our scheme is also suitable for time-release delegation. (Alice uploads her sharing data to the CSP in time period i, and Bob sends Alice a data request in arbitrary time period j, where $i < j$.) Specifically, when Alice uploads the encrypted data to the CSP, she can make a record on the map of the data message (some characteristics, e.g. hash value of the data) and the current time period i as $< data, i >$, and add the record to a list stored locally. Since the computation and storage cost of this record is quite cheap, it is acceptable to a personal computer. When Alice receives a data request from Bob, she first generates $rk_{A\xrightarrow{i}B,1}$ with Bob's public key PK_B in the same way as the above scheme. Then Alice search corresponding record from her list and get the time period i^*. She continues to compute SK_{A,i^*} with her helper (since our scheme supports random access key update which is defined in Section 6.2.1), rather than $SK_{A,i}$ which is the secret key for the current time period i. Finally, Alice computes $rk_{A\xrightarrow{i}B,2} = rk^{1/x_{A,i^*}}_{A\xrightarrow{i}B,1} = g^{x_{B,i}/x_{A,i^*}}$ and sends $rk_{A\xrightarrow{i}B,2}$ to the CSP. Thus, the data outsourced in previous time periods can also be correctly delegated in the following time periods.

6.2 Security Analysis

6.2.1 Key-insulation

In our scheme, the system lifetime is divided into a number of time periods and the user's secret key is not wholly preserved by himself but updated with his/her helper's help in each time period. For each time period, the user's secret key is generated with the previous secret key and the assistance of his unique physically-secure helper. When a legal user is intruded by an attacker, he can only decrypt the ciphertext encrypted for the current time period, since the user's secret key for each time period is distinct. Particularly, in an original PRE scheme, the attacker can decrypt all the ciphertext through the whole system lifetime if he compromises the secret keys of the delegator and the delegatee. However, our scheme enjoys the advantage of key insulation. The attacker cannot get the whole secret keys for any other time periods, even if he captures the temporary secret keys for several time periods, since he cannot get access to the helper. Moreover, our scheme satisfies the following properties: secure key update and random-access key update.

Secure key update. Similar to [5], we define secure key update as follows: A scheme satisfies secure key update if a *key-update exposure* in period i is equivalent to key exposures in both period $i-1$ and period i. The *key-update exposure* in period i denotes that the key exposure happens while the key update from SK_{i-1} to SK_i is taking place. (i.e. the attacker can get SK_{i-1}, SK'_i and SK_i.) We argue that our scheme has secure key update, since the attacker who makes key exposures in period $i-1$ and i can obviously get SK_{i-1} and SK_i. Then he can further compute $SK'_i = SK_i - SK_{i-1}$. That is the same as *key-update exposure* in period i.

Random access key update. *Random access key update* denotes that the temporary secret key can be updated to any other one for arbitrary time period instead of the next one. Obviously, our scheme enjoys the advantage of *random access key update*. In particular, the helper can compute the helper key as $SK'_{A,i+\Delta} = x'_{A,i+\Delta} = \sum_{j=1}^{t} x^*_{A,j}((i+\Delta)^j - i^j)$, and the user can further compute $SK_{A,i+\Delta} = x_{A,i} + x'_{A,i+\Delta} = \sum_{j=0}^{t}(x^*_{A,j} i^j) + \sum_{j=1}^{t} x^*_{A,j}((i+\Delta)^j - i^j) = \sum_{j=0}^{t}(x^*_{A,j} \cdot (i+\Delta)^j)$. It correctly performs the *random access key update* from time period i to $i+\Delta$, where Δ is the time period distance.

Table 1: Property comparison

Schemes	PRE	Unidirectional	Temporary delegation	Time-release delegation	Key insulation	Key update	Trusted server free	Helper requirement
[1]-2	√	√	×	×	×	×	√	×
[1]-3	√	√	×	×	×	×	√	×
[1]-4	√	√	√	×	×	×	×	×
Our scheme	√	√	√	√	√	√	√	√

6.2.2 Semantic Security

Since key-insulated mechanism has well addressed the exposure of user's secret key, we then analyze the semantic security of our scheme. Actually, this scheme has many attractive properties such as efficient unidirectional, non-interactive, proxy invisible and nontransitive.

The security of our scheme is based on the q-wDBDHI assumption defined in Section 3.2, and we set $q = t + 1$. Think of g^b as g^{ak} for some $k \in \mathbb{Z}_q^*$, and we consider the original ciphertext $C = (g^b, M \cdot \Gamma)$ which is encrypted for public key g^a (actually may be $g^{\sum_{j=0}^{t}(x_{u,j}^* \cdot i^j)}$, where u denotes a valid user.) and message M. Check $\Gamma \stackrel{?}{=} e(g,g)^k$, where $e(g,g)^k = e(g,g)^{b/a} = e(g,g)^{ak/a}$. If it is true, C is a correct ciphertext of M, otherwise, it is a ciphertext of some other message $M' \neq M$. Therefore, the semantic security of our scheme can be easily broken by the adversary that can solve the q-wDBDHI problem (which is indeed proven hard in the generic group by Dodis and Yampolskiy [6]). Moreover, the security of our scheme is also based on the assumption that a cannot be figured out given a tuple (g, g^a), where $g \in \mathbb{G}$ and $a \in \mathbb{Z}_q^*$.

6.3 Comparison to Existing Works

In this section, we show some attractive properties of our scheme compared to Ateniese et al's schemes [1], which is very similar to our scheme in system time division and temporary delegation.

In fact, there are four schemes proposed in [1]. However, the first scheme (we omit it in Table 1) cannot be seen as a pure proxy re-encryption scheme, since it is actually a special encryption scheme which has two decryption approaches rather than has a ciphertext transformation between users. The second and third schemes are much more like normal proxy re-encryption schemes with unidirectionality, non-interactivity, collusion-resilience and non-transitivity, unfortunately, they are ordinary versions without temporary delegation and the protection of user's secret key. Ateniese et al's fourth scheme, which can be deemed as an improvement of the previous three schemes, is very attractive with the properties of temporary delegation. Its system lifetime is also divided into several time periods to ensure a temporary delegation in period i, which is similar to our proposed scheme. They deployed a trusted server which broadcasted a random value $h_i \in \mathbb{G}_1$ in each time period for all users to see. Obviously, it is an efficient approach to enable Alice to temporarily delegate her decryption right to Bob for some period i. However, we consider that there are some drawbacks in reality. (1) To guarantee the honesty of a server on opening network is not quite easy. (2) It does not support time-release delegation which is very useful and flexible. (3) Most importantly, it suffers the problem of key exposure, i.e. if an adversary comprises Alice's and Bob's secret keys (h_i can be seen for all users), he will be able to decrypt all the ciphertexts, which is disastrous.

We argue that our scheme is very exciting. We deploy a physically-secure device named "helper" rather than a trusted server to achieve temporary delegation. Note that the helper is much more practical in reality compared to the trusted server, since it is isolated from the opening network and may be brought with the user. (In some sense, it is controlled by its owner rather than someone else.) Instead of broadcasting a random value h_i for every user, in our scheme, the user interacts with his own unique helper to update his temporary secret key for each time period i. Meanwhile, our scheme also supports time-release delegation which we explained in Section 6.1. In addition, since user's temporary key for each time period is distinct and can only be derived by interacting with the corresponding helper, the adversary can only compromise the temporary key for period i rather than the whole system lifetime, even if he makes a key exposure in period i. Thus, our KIPRE scheme enjoys the advantage of key insulation and mitigates the damage cased by key exposure.

Furthermore, we also provide the theoretical and experimental comparison with Ateniese et al's schemes [1] as follows.

Theoretical comparison. We show the theoretical comparison with Ateniese et al's schemes [1] in Table 1, Table 2 and Table 3 for property, communication and computation complexity, respectively. Since [1]-4 can be seen as an improvement of the previous three schemes in [1] and it is much more similar to our proposed scheme, we only give the efficiency comparison between our scheme and [1]-4 in Table 2 and Table 3. We define the notations we used in tables as follows: $|\mathbb{G}|$, $|\mathbb{G}_T|$ and $|\mathbb{Z}_q^*|$ respectively denote the bit-length of an element in $\mathbb{G}$, $\mathbb{G}_T$ and $\mathbb{Z}_q^*$. C_p, C_{e_T} and C_e denote the computation cost of a bilinear pairing, an exponentiation in $\mathbb{G}_T$ and an exponentiation in $\mathbb{G}$, respectively. t is the total number of the time periods that the system lifetime is divided into. We assume that the corresponding schemes share the same security parameter. Note by $\perp$ we mean non-applicable.

Experimental comparison. In addition, we provide an experimental evaluation of our proposed scheme and show the performance comparison with [1]-4. Our experiment is simulated on the PC equipped with an Intel Core i5-4460 Processor running at 3.2 GHz with 8G memory. The programming language is C and the operations in bilin-

Table 2: Communication comparison

Schemes	[1]-4	Ours
Secret key size	$2\|\mathbb{Z}_q^*\|$	$\|\mathbb{Z}_q^*\|$
Helper key size	$\perp$	$t\|\mathbb{Z}_q^*\|$
Original ciphertext size	$2\|\mathbb{G}_T\|$	$\|\mathbb{G}\| + \|\mathbb{G}_T\|$
Re-encrypted ciphertext size	$2\|\mathbb{G}_T\|$	$\|\mathbb{G}\| + \|\mathbb{G}_T\|$

Table 3: Computation comparison

Schemes	[1]-4	Ours
Initial keys generation	$2C_e$	$(t+1)C_e$
ReKey generation	C_e	C_e
Secret key update	$\perp$	tC_e
Original ciphertext generation	$2C_p + 2C_{e_T}$	$C_p + C_{e_T} + C_e$
Re-encrypted ciphertext generation	$2C_p + C_{e_T} + C_e$	C_p
Decryption (original ciphertext)	C_{e_T}	$C_p + C_{e_T}$
Decryption (re-encrypted ciphertext)	$C_p + 2C_{e_T}$	C_{e_T}

ear groups are implemented by using the stanford PBC library 0.5.14 (available at `https://crypto.stanford.edu/pbc/`). In our experimentation, we use PBC Type A pairing which is constructed on the curve $y^2 \equiv x^3 + x \mod p$ for some prime $p \equiv 3 \mod 4$ and the embedding degree is 2. We choose the 80-bit security level. The sizes of p, q are 512 bits and 160 bits respectively. The size of an element in group $\mathbb{G}$ is 1024 bits. With the above settings, we learn that an exponentiation operation in $\mathbb{G}$ costs 8.31 ms, an exponentiation operation in $\mathbb{G}_T$ costs 1.98 ms and a pairing operation costs 16.67 ms. Furthermore, we choose the number of time periods $t = 5$ and pellucidly describe the output results for each algorithms as well as the time consumption comparison with [1]-4 in Figure 2.

7 Conclusions

In this paper, we proposed a novel KIPRE scheme to achieve both key insulation and decryption right delegation in a cloud environment. Our scheme, for the first time, addressed the key exposure problem in PRE settings. Meanwhile, it not only supported temporary delegation, but also satisfied time-release delegation (we defined in Section 6.1). The main advantage of our scheme is that it can mitigate the damage caused by user's secret key leakage for data sharing in a cloud environment. Moreover, we also showed the acceptability of our scheme on security and efficiency.

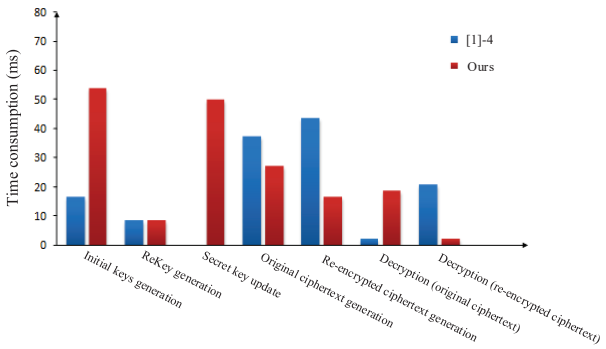

Figure 2: Time consumption comparison

Acknowledgments

This work was supported in part by the National Science Foundation of China (No. 61370026, No. 61602096 and No. 61370026), the National High Technology Research and Development Program of China (No. 2015AA016007), Science and Technology Project of Guangdong Province (No. 2016A010101002).

References

[1] 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.

[2] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *International Conference on the Theory and Application of Cryptography*, pp. 127–144, Springer, 1998.

[3] R. Canetti and S. Hohenberger, "Chosen-ciphertext secure proxy re-encryption," in *14th ACM Conference on Computer and Communications Security*, pp. 185–194, 2007.

[4] P. S. Chung, C. W. Liu, and M. S. Hwang, "A study of attribute-based proxy re-encryption scheme in cloud environments," *International Journal of Network Security*, vol. 16, no. 1, pp. 1–13, 2014.

[5] Y. Dodis, J. Katz, S. Xu, and M. Yung, "Key-insulated public key cryptosystems," in *International Conference on the Theory and Applications of Cryptographic Techniques Amsterdam*, pp. 65–82, 2002.

[6] Y. Dodis and A. Yampolskiy, "A verifiable random function with short proofs and keys," in *8th International Workshop on Theory and Practice in Public Key Cryptography*, pp. 416–431, 2005.

[7] X. Fu, "Unidirectional proxy re-encryption for access structure transformation in attribute-based encryption schemes," *International Journal of Network Security*, vol.17, no.2, PP. 142–149, Mar. 2015.

[8] G. Hanaoka, Y. Hanaoka, and H. Imai, "Parallel key-insulated public key encryption," in *9th International Conference on Theory and Practice in Public-Key Cryptography*, pp. 105–122, 2006.

[9] G. Hanaoka and J. Weng, "Generic constructions of parallel key-insulated encryption," in *7th International Conference on Security and Cryptography for Networks*, pp. 36–53, 2010.

[10] Y. Hanaoka, G. Hanaoka, J. Shikata, and H. Imai, "Unconditionally secure key insulated cryptosystems: models, bounds and constructions," in *4th International Conference on Information and Communications Security*, pp. 85–96, 2002.

[11] H. Hong and Z. Sun, "High efficient key-insulated attribute based encryption scheme without bilinear pairing operations," *SpringerPlus*, vol. 5, no. 1, pp. 1–12, 2016.

[12] A. Ivan and Y. Dodis, "Proxy cryptography revisited," in *Annual Network and Distributed System Security Symposium*, pp. 1–20, 2003.

[13] H. Khurana, A. Slagell, and R. Bonilla, "Sels: a secure e-mail list service," in *Proceedings of the 2005 ACM Symposium on Applied Computing*, pp. 306–313, 2005.

[14] C. Liu, W. Hsien, C. Yang, and M. Hwang, "A survey of attribute-based access control with user revocation in cloud data storage," *International Journal of Network Security*, vol. 18, no. 5, pp. 900–916, 2016.

[15] C. Liu, W. Hsien, C. Yang, and M. Hwang, "A survey of public auditing for shared data storage with user revocation in cloud computing," *International Journal of Network Security*, vol. 18, no. 4, pp. 650–666, 2016.

[16] Q. Liu, C. Tan, J. Wu, and G. Wang, "Reliable re-encryption in unreliable clouds," in *IEEE Global Telecommunications Conference (GLOBECOM'11)*, pp. 1–5, 2011.

[17] Q. Liu, G. Wang, and J. Wu, "Time-based proxy re-encryption scheme for secure data sharing in a cloud environment," *Information Sciences*, vol. 258, pp. 355–370, 2014.

[18] E. J. L. Lu, M. S. Hwang, C. J. Huang, "A new proxy signature scheme with revocation," *Applied mathematics and Computation*, vol. 161, no. 3, pp. 799–806, 2005.

[19] M. Mambo and E. Okamoto, "Proxy cryptosystems: Delegation of the power to decrypt ciphertexts," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 80, no. 1, pp. 54–63, 1997.

[20] R. Mukherjee and J. Atwood, "Proxy encryptions for secure multicast key management," in *28th Annual IEEE International Conference on Local Computer Networks*, pp. 377–384, 2003.

[21] C. Sur, C. Jung, Y. Park, and K. Rhee, "Chosen-ciphertext secure certificateless proxy re-encryption," in *IFIP International Conference on Communications and Multimedia Security*, pp. 214–232, 2010.

[22] X. Tian, X. Wang, and A. Zhou, "Dsp re-encryption based access control enforcement management mechanism in daas," *International Journal of Network Security*, vol. 15, no. 1, pp. 28–41, 2013.

[23] Z. Wan, J. Li, and X. Hong, "Parallel key-insulated signature scheme without random oracles," *Journal of Communications and Networks*, vol. 15, no. 3, pp. 252–257, 2013.

[24] Z. Wang, Y. Lu, G. Sun, "A policy-based deduplication mechanism for securing cloud storage," *International Journal of Electronics and Information Engineering*, vol. 2, no. 2, pp. 70–79, 2015.

[25] T. Wu, Y. Tseng, and C. Yu, "Id-based key-insulated signature scheme with batch verifications and its novel application," *International Journal of Innovative Computing, Information and Control*, vol. 8, no. 7, 2012.

[26] Q. Xie, S. Jiang, L. Wang, and C. Chang, "Composable secure roaming authentication protocol for cloud-assisted body sensor networks," *International Journal of Network Security*, vol. 18, no. 5, pp. 816–831, 2016.

[27] P. Xu, T. Jiao, Q. Wu, W. Wang, and H. Jin, "Conditional identity-based broadcast proxy re-encryption and its application to cloud email," *IEEE Transactions on Computers*, vol. 65, no. 1, pp. 66–79, 2016.

[28] M. Zareapoor, P. Shamsolmoali, and M. A. Alam, "Establishing safe cloud: Ensuring data security and performance evaluation," *International Journal of Electronics and Information Engineering*, vol. 1, no. 2, pp. 88–99, 2014.

Biography

Yilei Wang is pursuing his Ph.D. degree in the School of Information and Software Engineering, University of Electronic Science and Technology of China (UESTC). He received his B.S. degree from School of Communication and Information, UESTC, in 2008. He received his M.S. from Faculty of Engineering, Lund University of Sweden, in 2011. His research interests include security and application of mobile network data.

Dongjie Yan received his B.S. degree from University of Electronic Science and Technology of China (UESTC) in 2014. He is currently pursuing his M.S. degree in the School of Information and Software Engineering, UESTC. His research interests include cryptographic protocols and network security.

Fagen Li received his Ph.D. degree in Cryptography from Xidian University, Xi'an, P.R. China in 2007. He is now a associate professor in the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC). His recent research interests include cryptography and network security.

Hu Xiong is an associate Professor at University of Electronic Science and Technology of China (UESTC). He received his Ph.D. degree from UESTC in 2009. His research interests include cryptographic protocols and network security.