PAPER Special Section on the Architectures, Protocols, and Applications for the Future Internet

Achieving Scalable and Optimized Attribute Revocation in Cloud Computing

Somchart FUGKEAW[†a)], ***Student Member and*** **Hiroyuki SATO**[††], ***Nonmember***

SUMMARY Revocation is one of the major problems for access control systems. Especially, the revocation cost for the data outsourced in the third party environment such as cloud storage systems. The revocation in the cloud-based access control typically deals with the cryptographic operations that introduce costly overheads for key re-generation, file re-encryption, and key re-distribution. Also, the communication for retrieving files for re-encryption and loading them back to the cloud is another non-trivial cost for data owners. In this paper, we propose a Very Lightweight Proxy Re-Encryption (VL-PRE) scheme to efficiently support attribute-based revocation and policy update in the collaborative data sharing in cloud computing environment. To this end, we propose three-phase VL-PRE protocol including re-encryption key generation, re-encryption key update, and re-encryption key renewal for supporting the optimized attribute revocation and policy update. Finally, we conduct the experiments to evaluate the performance of our VL-PRE and show that it exhibits less computation cost with higher scalability in comparison with existing PRE schemes.

key words: *revocation, data access control, policy update, proxy re-encryption*

1. Introduction

Outsourcing data to the cloud is becoming a highly demanded service adopted by many enterprises due to the economy, scalability, ease of implementation, accessibility, and elasticity of computing resource management. Among the major concerns for cloud adoption, privacy and security of the sensitive data to be outsourced are considered as top requirements. Cryptographic-based access control having features of access control enforcement and data encryption is usually required for securing outsourced data. Even though, the traditional public key encryption or public key infrastructure (PKI) offers strong encryption, it would render the multiple copies of encrypted data as it uses each public key of all users to encrypt the data. While, the symmetric encryption encounters the maintenance cost and key distribution problem when there are a large number of users accessing the shared data.

Therefore, cryptographic-based access control using traditional PKI and symmetric key encryption is not suitable for supporting the access control in data outsourcing environment where there are multiple unknown users accessing the shared data.

In 2001, Boneh and Franklin [20] proposed fully functional Identity-based encryption (IBE) which is a public key cryptosystem based on the bilinear pairing. IBE offers a more flexibility for the encryption as it uses the identity attributes such as user id or email address to encrypt the data. The users who have the identity attributes used to encrypt the data can decrypt the data. However, IBE has limitations in offering the fine-grained policy enforcement when the data owner needs to specify a more comprehensive policy based on the collection of other attributes.

Ciphertext Policy-Attribute Based Encryption (CP-ABE) which is a kind of attribute-based encryption (ABE) was proposed [5] to overcome the limitation of IBE. CP-ABE provides more efficient and flexible access control solution for data outsourcing scenario. In CP-ABE, a data file is encrypted by an access policy constructed from a set of attributes and only users who have attributes satisfying the policy can decrypt the file. Even though CP-ABE is flexible and scalable in supporting secure data access control in data outsourcing, the major problem CP-ABE scheme is revocation management. In CP-ABE, when a user or an attribute is revoked, or the policy is updated, the data owner needs to download the affected ciphertexts, and re-encrypt them with a new re-encryption key before they are sent back to the cloud. Also, all non-revoked users are given with a new decryption key generated by the attribute authority (AA).

To alleviate such revocation effects, Proxy-based Re-encryption (PRE) is considered as an effective method to reduce the re-encryption cost in CP-ABE based revocation management. The concept of PRE is to delegate the re-encryption function to a semi-trusted proxy. Even though re-encryption is a major cost for the revocation and policy update, re-encryption key generation is another expensive overhead that data owners must account for. For today's advanced data sharing scenario, managing and accessing shared data can be done through the mobile device such as mobile phone or tablet.

For instance, a data owner can use a smart phone to compute and submit a small data to the proxy for supporting re-encryption key generation and file re-encryption in case of the attribute revocation or policy update. In a mobile cloud computing (MCC), users can also use mobile devices to execute the security functions for interacting with the applications or accessing the data outsourced. Hence, lightweight cryptographic protocols that require less power

Manuscript received July 22, 2016.
Manuscript revised December 9, 2016.
Manuscript publicized February 8, 2017.

[†]The author is with Graduate School of Engineering, The University of Tokyo, Tokyo, 113–8658 Japan.
[††]The author is with Department of Electrical Engineering and Information Systems, The University of Tokyo, Tokyo, 113–8658 Japan.
a) E-mail: somchart@satolab.itc.u-tokyo.ac.jp

DOI: 10.1587/transinf.2016NTP0006

consumption on mobile devices are essential, as it will enable the users to compute the security operation without the performance problem [18].

To date, most recent PRE schemes [2], [11], [13]–[15] have been designed to offload costly overheads of re-encryption key generation and ciphertext re-encryption to the proxy as much as possible. Outsourcing the computation of re-encryption key can be considered as optimized or lightweight PRE since the computation cost at client in dealing with PRE process is delegated to the proxy in the cloud. Thus, the naïve problem of the existing PRE schemes is the heavy loads at the proxy. In a cloud scenario, the cloud provider may charge the service usage based on CPU time in some cases. Thus, avoiding too much overload in terms of the frequency of re-encryption operation performed by a proxy is considered as a practical solution. This computation optimization aspect is mostly overlooked by the existing PRE schemes.

In this paper, we entail a more practical PRE scheme by proposing a very lightweight PRE (VL-PRE) to support a more flexible and scalable attribute revocation and policy update compared to existing PRE-based schemes.

Compared to those existing lightweight PRE schemes [2], [11], [13]–[15], VL-PRE is proposed to improve the computation and communication performance with a more lightweight PRE protocol. Existing lightweight PRE schemes focus on the optimization of the PRE cost at client side only, while VL-PRE aims at optimizing computation and communication cost at both data owner and cloud side. To this end, VL-PRE possesses two major optimization aspects including the reduction of re-encryption key size and the use of key update strategy in supporting attribute revocation or policy update. In VL-PRE, a root decryption key (RDK) is used to decrypt the existing ciphertext and it is a part of re-encryption key components sent to a proxy. The RDK used in VL-PRE contains only two attributes: owner id and digital signature. Also, VL-PRE employs re-encryption key update strategy to optimize the occurrences of re-encryption key generation.

With a very light packet for re-encryption key generation, it requires little memory and low computation for data pre-processing. Besides, the key update strategy improves the performance of PRE process and reduces the workload of the proxy.

We summarize the contributions of this paper as follows.

1. We introduce VL-PRE to enable the efficient and practical revocation and policy update in attribute-based access control in cloud computing. With the design of full outsourcing of re-encryption key generation, it eliminates communication cost at data owner side.
2. VL-PRE applies the re-encryption key update strategy to minimize the re-encryption key generation cost run by a proxy.
3. VL-PRE requires minimal computation and small size of data in dealing with PRE process. These lightweight properties enables high accessibility and flexibility in managing attribute revocation or policy update by using resource-constrained mobile devices.
4. We conduct the simulations to assess the performance of our proposed VL-PRE and compare with existing PRE schemes.

The remainder of this paper is organized as follows. Section 2 presents works related to our proposed approach. Section 3 reviews the theoretical backgrounds of the concepts used to construct our access control model. Section 4 describes our proposed VL-PRE scheme. Section 5 analyzes the security properties of the VL-PRE. Section 6 depicts the experimental evaluation. Finally, conclusion is given in Sect. 7.

2. Related Work

Most CP-ABE-based models focus on introducing the access control enforcement and proposing techniques to solve key management issue in a more complex environment such as multi-authority ABE (MA-ABE) [1], [4], [9], [10], [12] as well as to increase the cryptographic performance such as encryption and decryption, minimize the content delivery cost, and address the revocation problem. Especially, the latter problem introduces substantial costs in terms of computation overheads and renders the impact to existing users who share the outsourced data.

Proxy-based Re-encryption (PRE) was initially introduced by Mambo and Okamoto [6]. It is a technique that re-encryption task is delegated to an outsourcing server. Basically, re-encryption of the ciphertext sent by the originator and the delegated proxy learns neither the decryption keys nor original plaintext. For applying PRE to support data sharing in the cloud, data owner or content provider updates re-encryption keys for their group of users and enforces shared secret keys among authorized subscribers, with which the data or content retrieved from the cloud can be decrypted [3].

Generally, the PRE scheme can be classified into two major solutions: traditional PRE and outsourcing PRE. The traditional PRE schemes [6]–[8] have been adopted by both identity based encryption (IBE) [13] and attribute-based encryption (ABE) in supporting the revocation. However, the traditional PRE focuses only on delegating the re-encryption task to the proxy.The data owner or client needs to bear the computation cost for re-encryption key generation and new user key re-distribution.

Recent IBE [13] and ABE [2], [11], [14]–[16] approaches considered that the cost at client in dealing with the computation of re-encryption key should be optimized by offloading it to the proxy as well so that the client can save the communication and computation cost. This solution is classified as an outsourcing PRE approach. For example, K. Liang et al. [13] introduced re-encryption key update process to be done at a cloud. To do so, the public key generator publishes a constant-size public string at the beginning

of each time period. If there is any user revoked, the proxy re-encrypts the ciphertext to the next time period. Hence, revoked users cannot decrypt the re-encrypted ciphertext with their existing keys. However, this approach does not support attribute revocation, which is a finer revocation level.

For the PRE optimization in CP-ABE, Son et al. [2] proposed the outsourcing conditional PRE (OC-PRE) scheme with a focus on reducing overhead at a client side in dealing with the initial setup stage. In this scheme, the originator computes the conditional value and re-encryption keys and sends to the proxy in cloud to perform re-encryption process. If there is a change on membership, the originators or clients just re-compute the conditional value changing key (CCK) and the CCK will transform the ciphertext with a new conditional value. However, the paper only focuses on the change of user level while an attribute revocation level has not been discussed.

In [11], Tysowski et al. proposed a protocol based on ABE and PRE for supporting secure data sharing in mobile cloud computing. The PRE is used to reduce the computation workload on the data owner side. Key generation is divided between mobile data owner and trusted authorities. When a user is revoked, a trusted party called a manager needs to compute the re-encryption key based on the user group key value. Then, the re-encryption key is sent to the cloud and the cloud provider computes the new ciphertext. Nevertheless, their scheme requires another trusted party for computing the re-encryption key. Thus, the vulnerability and workload at the manager are the issues.

Nevertheless, the aforementioned outsourcing PRE approaches have not segregated the revocation level between user and attributes. Besides, they have not provided the practical implementation to show how much the PRE improves the revocation scenario where there are frequent revocation cases or policy updates. More importantly, all the presented PRE schemes have not focused on balancing the PRE cost at both client and the cloud side.

3. Background

In this paper, we extend our access control model called C-CP-ARBE [12] scheme to be capable to efficiently support the evolution of attributes and access control policy. To this end, VL-PRE scheme is proposed as an attribute revocation mechanism, which is embedded as a part of our C-CP-ARBE system. In essence, VL-PRE aims at delivering higher scalability with less computation and communication cost at both data and PRE proxy.

In this section, we first introduce the system definitions and discuss our access control model called Collaborative-Ciphertext Policy-Attribute Role-based Encryption (C-CP-ARBE) [12].

3.1 System Definitions

Definition 1: Bilinear map
Let G_1 and G_2 be two multiplicative cyclic groups of prime order p and e be a bilinear map, $e : G_1 \times G_1 \rightarrow G_2$. Let g be a generator of G_1. Let H: {0,1}* $\rightarrow G_1$ be a hash function that the security model is in a random oracle.

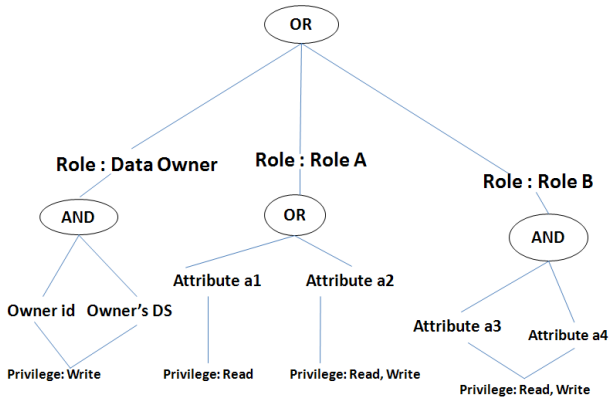

Fig. 1 Access control Policy Structure

The bilinear map e has the following properties:
1. Bilinearity: for all u, v $\in G_1$ and a, b $\in Z_p$, $e(u_a, v_b) = e(u, v)^{ab}$
2. Non-degeneracy: $e(g, g) \neq 1$.

Definition 2: (Access Structure [5])
Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection A $\subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ is monotone if

$$\forall B, C \; B \in A \text{ and } B \subseteq C \text{ then } C \in A.$$

An access structure is a collection A of non-empty subsets of $\{P_1, P_2, \ldots, P_n\}$, i.e. $A \subseteq 2^{\{P_1, P_2, \ldots, P_n\}} \setminus \{\emptyset\}$.
A set $X \in A$ is called the authorized set wrt. A.

In the context of CP-ABE, a set of parties is taken by the attributes. Thus, the access structure A consists of the authorized sets of attributes.

Definition 3: Access Control Policy (ACP)
ACP is a tree-based structure. Let ACP T be a tree that represents the access structure in C-CP-ARBE [12]. Each non-leaf node of the ACP tree represents the "Role" attribute and threshold gate. The threshold gate rule is the same as access tree of CP-ABE.
We denote the parent of a given node x by parent(x). Thus, the parent of a leaf node x is the pair of (Role, threshold gate).

In the ACP, data owner identity (Owner id) attribute and digital signature of her id value are embedded in the ACP by default. According to Fig. 1, it depicts that the privilege is modeled in the tree to specify the permission of user assigned to a specific role and set of attributes.

3.2 Collaborative-Ciphertext Policy – Attribute Role-based Encryption (C-CP-ARBE) Scheme

C-CP-ARBE is a multi-authority CP-ABE (MA-ABE) [12] which is a collection of twelve algorithms. The details of

Table 1 Notations used in our access control scheme

Notation	Description
$S_{uid.aid}$	A set of all attributes issued to user *uid* and managed by authority *aid*.
SK_{aid}	A secret key which belongs to authority *aid*.
PK_{aid}	A public key which belongs to authority *aid*.
GSK_{uid}	A global secret key of a user *uid*. GSK is a private key issued by the certification authority CA.
$Cert_{uid}$	A public key certificate containing user's public key issued by a certification authority CA.
$UDK_{uid.aid}$	A user decryption key issued by authority *aid*
$EDK_{uid.aid}$	An encrypted form of a $UDK_{uid.aid}$ encrypted by a user's public key.
RDK_{oid}	A root decryption key which belongs to data owner *oid*.
Owner id (O_{id})	An identity attribute of the data owner.
Owner's DS	A digital signature derived from a signing of data owner's private key over her id (owner id) value.
GRP (Group role parameter)	Seed numbers computed from all authorized user members of the role.
M	Message to be encrypted
SS (Secret Seal)	A symmetric key created from the AES algorithm together with the GRP.
ACP	An access control policy used to encrypt the data files.
CT	A ciphertext or a message encrypted by the CP-ABE method
SCT (Sealed ciphertext)	A ciphertext encrypted with the SS

these algorithms are described in [12]. C-CP-ARBE supports the enforcement of policy on attributes issued by multiple attribute authorities (AAs).

Key characteristics of C-CP-ARBE can be summarized as follows.

- C-CP-ARBE model is based on the combination of role-based access control model and CP-ABE scheme where the bilinear map is a major cryptographic key generation and the access structure defined in [5] is applied to formulate our ACP.
- Similar to CP-ABE, users' secret keys (we call user decryption keys (UDKs)) are associated with a set of attributes.
- C-CP-ARBE exploits two-layer encryption consisting of a CP-ABE encryption and a symmetric encryption to deliver the fine-grained attribute based encryption and optimized user revocation cost.
- C-CP-ARBE provides no user decryption key (UDK) distribution cost as all generated UDKs are encrypted with the individual user's public key and stored in the cloud.

The following table presents the notations and its description used in our proposed algorithms.

Here, we present four major algorithms used in this paper. The algorithms include user key generation (UserKeyGen), root decryption key generation (RDKGen), encryption (Enc), and decryption (Dec). In our scheme, UserKeyGen and RDKGen are run by the AA, while Enc and Dec are executed by data owners or users.

- **UserKeyGen**($S_{uid,aid}$, SK_{aid}, $Cert_{uid}$) $\rightarrow$ $EDK_{uid,aid}$, RDK_{aid}.
  The UserKeyGen algorithm consists of the UDKGen and EDKGen algorithms to produce the $UDK_{uid,aid}$ and $EDK_{uid,aid}$ respectively:

  (1) UDKGen takes input as set of attributes $S_{uid,aid}$, attribute authority's secret key SK_{aid}, and user's certificate $Cert_{uid}$, then it returns a set of user decryption keys $UDK_{uid,aid}$. The algorithm first chooses a random $r \in Z_p$, and then random $r_i \in Z_p$ for each attribute $i \in S_{uid,aid}$. Then, the key is computed as:

$$UDK_{uid,aid} = (D = g^{(\alpha_k + r)/\beta_k},$$
$$A_i \in S : D_i = g^r \cdot H(i)^{r_i}, D_i' = g^{r_i}).$$

  (2) EDKGen then encrypts the $UDK_{uid,aid}$ with the public key of the user $Cert_{uid}$ and outputs an encrypted decryption key $EDK_{uid,aid}$. The encryption function is expressed as:

$$UDK_{uid,aid} \rightarrow ENC_{RSA}(Cert_{uid}, UDK_{uid,aid})$$
$$\equiv EDK_{uid,aid}$$

  The derived $EDK_{uid,aid}$ is then stored in the cloud and it can be requested anytime by the legitimate user who has the matched GSK_{uid}.

- **RDKGen**(SK_{aid}, Oid, Owner's DS) $\rightarrow$ RDK_{oid}.
  RDKGen takes input attribute authority's secret key SK_{aid}, identity attribute of data owner O_{id}, and data owner's digital signature Owner's DS, then it produces the root decryption key RDK_{oid} for further use in re-encryption key generation process.
- **ENC(PK_{aid}, SS, M, ACP, $Cert_{uid}$) $\rightarrow$ SCT.** The encryption algorithm performs two consecutive steps as follows:

  (1) Encrypt Message M

$$M \rightarrow ENC_{C\text{-}CP\text{-}ARBE}(PK_{aid}, ACP, M) \equiv CT$$

  The algorithm takes as inputs authority public key PK_{aid}, access control policy ACP, and data M. Then it returns a ciphertext CT.

(2) Encrypt Ciphertext CT and SS

$$CT \rightarrow ENC_{AES}(SS, CT) \equiv SCT$$
$$SS \rightarrow ENC_{RSA}(Cert_{uid}, SS) \equiv ESS$$

Then, the algorithm takes group role parameter GRP as a key together with AES algorithm to generate the session key referred as a secret seal SS to encrypt the ciphertext CT. Then, the algorithm returns sealed ciphertext SCT.
Finally, the SS is encrypted with user's public key $Cert_{uid}$, the algorithm will return encrypted SS ESS and it is stored in a cloud server.

- **DEC(PK_{aid}, SCT, GSK_{uid}, $EDK_{uid,}$) → M.** The decryption algorithm performs two consecutive steps as follows:

(1) Decrypt SS and SCT

$$SS = DEC_{RSA}(GSK_{uid}, ENC_{RSA}(Cert_{uid}, SS))$$
$$CT = DEC_{AES}(SS, SCT)$$

The algorithm takes user's global secret key GSK_{uid} and it returns the session key SS. Then, the algorithm takes SS to decrypt SCT and gets the CT.

(2) Decrypt CT
The user (U_{id}) requests a cloud for $EDK_{uid,aid}$ for decryption. The decryption step is conducted as follows:

$$UDK_{uid,aid} = DEC(GSK_{uid}, EDK_{uid,aid})$$
$$M = DEC_{C\text{-}CP\text{-}ARBE}(UDK_{uid,aid}, CT)$$
$$\equiv DEC(UDK_{uid,aid}, ENC(PK_{aid}, ACP, M))$$

The algorithm takes user's global secret key GSK_{uid} to decrypt the encrypted user decryption key $EDK_{uid,aid}$. Then, the algorithm returns $UDK_{uid,aid}$. Finally, the $UDK_{uid,aid}$ is used to decrypt the ciphertext CT and it returns the message M.

4. Our Proposed VL-PRE

4.1 System Model

Figure 2 presents our proposed access control system embedded with proxy re-encryption function for data outsourcing scenario.

In the data outsourcing scenario, data owners initially upload encrypted data files to a cloud storage server. For the access control, we deploy C-CP-ARBE system accommodating access control functions to support authentication and authorization in multi-user and multi-owner cloud setting. In our approach, we introduce a PRE function with semi-trusted proxy server as the attribute revocation mechanism of the C-CP-ARBE system. A proxy is generally responsible for (1) computing or updating the re-encryption key initiated by data owners when there is attribute revocation and the revoked attribute appear in the access policy.

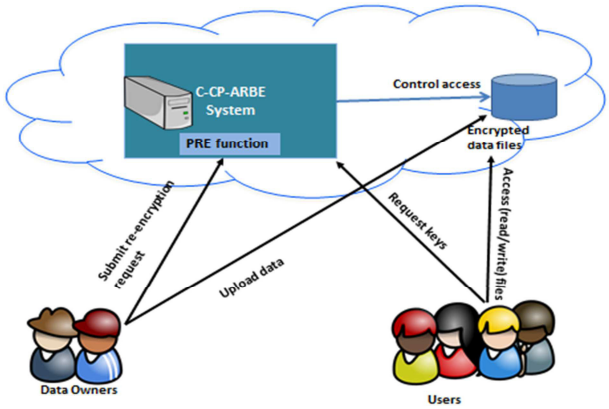

Fig. 2 A System model of Access Control Model with PRE for Outsourced Data

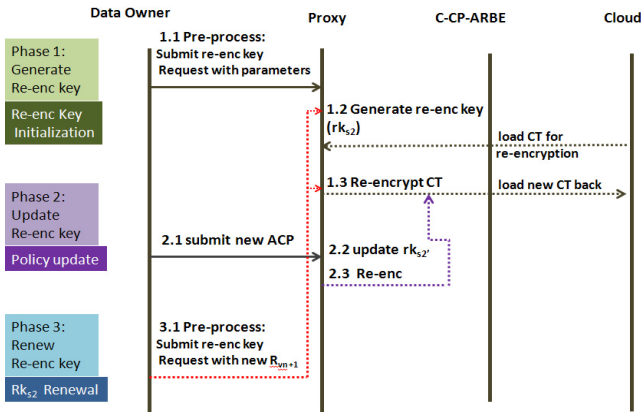

Fig. 3 VL-PRE process

(2) re-encrypting the ciphertext by using the generated re-encryption key. A proxy is issued a key pair and a public certificate which are used to verify the authenticity of the delegated proxy. The proxy has no privilege in accessing the content of file as it has no knowledge of keys necessary for decryption. In a PRE in cloud environment, data owners have full privilege in holding the root key for decrypting any files and managing the policies used for encryption. Users typically request for accessing resources via C-CP-ARBE system. Any related-PRE processes are transparent to users.

4.2 VL-PRE Process

We design the execution of VL-PRE into three phases: Generate Re-encryption key, Update Re-encryption key generation, and Renew re-encryption key. Basically, the proxy transforms ciphertext CT_{k1} to CT_{k2} with a re-encryption key $RK(rk_{s1 \rightarrow s2})$ generated by a proxy server. In the proposed scheme, Phase 2 and phase 3 are added to reduce the computation costs at data owner side in preparing the re-encryption key generation package to the proxy. In addition, we keep re-encryption key in the proxy and make it updated upon the key changeover policy. To secure the re-encryption key components such as RDK and ACP, we em-

ploy a cryptographically secure random number generator (CSPRNG) [17] (we refer as random number or randomness throughout the paper) for securing these key components. Figure 3 illustrates the overall process of VL-PRE.

Phase I: Generate Re-encryption key (Initialization):
In Phase I, the initial re-encryption key, system and all related-PRE parameters are initially generated. Some of them such as re-encryption key generated, random number can be used in a specified period of time. This phase consists of Pre-process, ReKeyGen, and ReENC algorithms which are described as follows.

- **Pre-process:** Data owner initially runs the following algorithms.
  (1) **GenPREConfig**($\mathrm{IP_{Proxy}}$, $\mathrm{Cert_{ownerid}}$) → $\mathrm{PREConF_{V1}}$
  The algorithm takes the input as the network address of proxy server, $\mathrm{IP_{Proxy}}$ and owner's public key certificate, $\mathrm{Cert_{ownerid}}$ which is used to authenticate with the proxy before executing the ReKeyGen algorithm. It then outputs the PRE configuration file, $\mathrm{PREConF_{V1}}$. In general, the PRE configuration file is generated once and can be used until there is a change of its input parameter. The file will be included with the *param* which is a part of Re-KeyGen algorithm.
  (2) **GenR(R{$rs_1, rs_2, \ldots rs_n$}) → $\mathbf{R_{vn}}$**
  The algorithms randomly chooses a set of random seeds *rs* as input and generates secure random number R.
  (3) **AppR($\mathbf{R_{vn}}$, $\mathbf{RDK_{aid}}$, $\mathbf{ACP_{vn}}$) → [R($\mathbf{RDK_{aid}}$), $\boldsymbol{ACP_{vn}^{R_{vn}}}$].**
  The algorithm applies random number R_{vn} (tagged with the current version number *vn*) to encrypt the root decryption key $\mathrm{RDK_{aid}}$ and the attributes in the leaf node of the updated version of access control policy ACP_{vn}. Then it outputs the encrypted RDK, $R(\mathrm{RDK_{aid}})$ and encrypted access policy $ACP_{vn}^{R_{vn}}$.

Then, data owner submits $\mathrm{PREConF_{V1}}$, encrypted $\mathrm{RDK_{aid}}$ and $ACP_{v_n}^{R_{vn}}$ as parts of re-encryption key to the proxy.

- **ReKeyGen**(param; SS, $R_{vn}(\mathrm{RDK_{aid}})$, $(ACP_{v_n}^{R_{vn}})$, ExpireTime) → $\mathrm{rk_{s2\rightarrow(M',ACP')}}$.
  The algorithm takes input param, secret seal SS, root decryption key encrypted by the Random R_{vn}, $R_{vn}(\mathrm{RDK_{aid}})$, a new access policy embedded with Random R_{vn}, $ACP_{v_n}^{R_{vn}}$, and Expire_time. First, the SS is used to decrypt the sealed ciphertext (SCT) and the original ciphertext (CT) is derived. The Expire_time is used to indicate the validity of re-encryption key $\mathrm{rk_{s2}}$. Hence, if the key expires, the owner needs to initiate re-key generation with a new random $\mathrm{R_{vn}}$.

Then, the algorithm outputs a re-encryption key $\mathrm{rk_{s2\rightarrow(M',ACP')}}$ that can be used to transform a ciphertext under (M, ACP) to another ciphertext under (M′, ACP′).

- **ReENC**(param; $\mathrm{rk_{s2\rightarrow(M',ACP')}}$, CMR function, CT(M, ACP)) → $\mathrm{CT_{k2}}$: The algorithm takes input param, a re-encryption key $\mathrm{rk_{s2\rightarrow(M',ACP')}}$, CMR function, and an original CT(M, ACP). It outputs a re-encrypted ciphertext CT′(M′, ACP′).

According to the element of $\mathrm{rk_{s2}}$, we embed the Combine-MatchRemove (CMR) function deployed in the execution file format to support the re-encryption process as follows:

(1) Combine pieces of R applied in leaf nodes of a new ACP′.
(2) Match R between ACP′ and R($\mathrm{RDK_{aid}}$).)).
(3) Remove R from $R_{vn}(\mathrm{RDK_{aid}})$.

Then, the $\mathrm{RDK_{aid}}$ is automatically used to decrypt the old ciphertext and the algorithm applies a new ACP′ to re-encrypt the data. The ciphertext re-encryption is done by the following ReENC algorithm.

$$\mathrm{CT} \rightarrow \mathrm{ReENC_{C\text{-}CP\text{-}ARBE}}(\mathrm{RDK_{aid}}, \mathrm{CT}, \mathrm{ACP'}) \equiv \mathrm{CT'}$$

Finally, the proxy takes SS to encrypt a new Ciphertext ($\mathrm{CT_{k2}}$). Since $\mathrm{rk_{s2}}$ is independent to the original message M, the re-encryption does not harm any privacy of the message.

Phase 2: Update Re-encryption key:
There are two algorithms for updating re-encryption key.

1. UpdateACP($\mathbf{R_{vn}}$, $\mathbf{ACP_{vn+1}}$) → $\boldsymbol{ACP_{v_{n+1}}^{R_{vn}}}$

Data owner applies current random number $\mathrm{R_{vn}}$ to encrypt the updated ACP, and the $ACP_{v_{n+1}}^{R_{vn}}$ is obtained and sent to the proxy.

2. UpdateReEncKey($\mathbf{rk_{s2,vn}}$, $\boldsymbol{ACP_{v_{n+1}}^{R_{vn}}}$) → $\mathbf{rk_{s2,vn+1}}$

The proxy run the algorithm by taking the updated ACP, $ACP_{v_{n+1}}^{R_{vn}}$ to update the current version of re-encryption key, $\mathrm{rk_{s2,vn}}$. The new $\mathrm{rk_{s2,vn+1}}$ is used to re-encrypt the ciphertext.

The algorithms help to reduce both computation and communication overhead at both data owner side and proxy. This is because RDK needs not to be encrypted every time and the information (only the updated ACP) sent out to the proxy is small. Besides, the proxy does not need to fully compute a new re-encryption key upon every revocations; it only updates the key instead.

Phase 3: Renew Re-encryption key
In this phase, if the current re-encryption key $\mathrm{rk_{s2,vn}}$ expires, the GenR, AppR, ReKeyGen and ReEnc algorithms in phase 1 will be run. Then, re-encryption key generation and ciphertext re-encryption are performed by the proxy. In the renewal phase, PREConF file and CMR function are not required to be re-generated. They can be re-called for generating (updating) the re-encryption key automatically.

Thus, the re-enc key renewal phase usually consumes less computation time than the initialization phase.

In addition, re-encryption key renewal is not required to perform instantly when the key expires, it will be executed when there is the next policy update or attribute revocation.

Table 2 Computation cost of VL-PRE

Operations	Computation Cost
Re-encryption Key Generation	$O(1) + O(RDK)\cdot C_r + (\#S_{aid}(\text{ACP}))\cdot C_r$
Re-encryption Key Update	$O(\#S_{aid}(\text{ACP}))$
Re-encryption Key Renewal	$O(RDK)\cdot C_{r'} + (\#S_{aid}(\text{ACP'}))\cdot C_{r'}$

4.3 Complexity Analysis of VL-PRE

In this section, we provide the computation complexity analysis of our proposed VL-PRE algorithms including Generate Re-enc Key, Update Re-encryption Key, and Renew RE-encryption key. The complexity is described in terms of computational processing time. Table 2 shows the computation complexity in terms of the processing time taken by the entire VL-PRE process. We use C_r to denote the computational cost of random encryption and S_{aid} is a set of attributes constituting the ACP.

Table 2 shows that VL-PRE enjoys constant complexity in each phase. The most expensive computation cost is taken by the re-encryption key generation which requires $O(1)$ in initializing the setup parameters which are parts of re-encryption key component. Then the major components of re-encryption key including RDK and ACP are encrypted with the randomness of which the computation cost is equal to the complexity of random encryption (C_r).

For the computation cost of re-encryption key update, we only need the processing time for updating the ACP which is proportional to the number of attributes S contained in the ACP. The computation cost for key update is small since only the updated policy is used to update the re-encryption key component. For the cost of re-encryption key renewal, it enjoys constant communication cost for renewing key for each expiry period. Also, the key renewal cost is less than the initial re-encryption key generation since there is no cost for the parameter setup. A new set of random r' is used to encrypt RDK and ACP.

5. Security Analysis

5.1 Security Property

Our VL-PRE technique is proposed as an optimized mechanism to support attribute revocation and policy update of our access control model C-CP-ARBE where its security construct is based on the original CP-ABE [5]. Based on the core construct of CP-ABE, we define the security properties possessed by C-CP-ARBE as follows.

Property 1: Since the underlying CP-ABE scheme is collusion resistant and secure under oracle security model, our C-CP-ARBE is also collusion resistant and secure in the standard oracle model.

Property 2: Suppose there is no polynomial time adversary who can attack the cryptographic algorithm of CP-ABE with nonnegligible advantage; then there is no polynomial time that adversary can attack our C-CP-ARBE with nonnegligible advantage.

5.2 Security Analysis of VL-PRE Process

Security of Re-encryption Key generation process
In VL-PRE, re-encryption key generation is fully outsourced to a delegated proxy, root decryption key and the most updated ACP which are the component of re-encryption key are encrypted with the random number. The outsourcing server or even the attackers who can collude the cloud and obtain the re-encryption key, they cannot gain access to the plaintext as the re-encryption key is encrypted by a cryptographically secure random number generator (CSRNG) proven to be secure for implementing in standardized cryptographic modules [21].

Security of Re-encryption Key update process
When there is an update of access control policy, the data owner will exploit an updated $\text{ACP}_{\text{vn}+1}$ where its leaf nodes are applied with Random R. Then the updated policy is sent to the proxy for updating the re-encryption key. The proxy runs re-encryption key update algorithm and the existing re-encryption key is updated to $\text{rk}_{2,\text{vn}+1}$. Here, the re-encryption key is only updated with a new ACP. The proxy does not have any advantage in gaining access to the plaintext since the RDK which is a part of re-encryption key remains encrypted.

Upon the re-encryption key-update, the security of the updated key is preserved in the same security level (random encryption) as the re-encryption key generation but it gains less computation cost compared to the re-computation of the a new re-encryption.

Security of Re-encryption key renewal process
The security of re-encryption key renewal shares the same security protection as the re-encryption key generation does. In our VL-PRE, once the key expires, a new randomness is re-generated and it will be used to encrypt the re-encryption key elements such as RDK and ACP. The proxy cannot use the expired keys to access the plain data as all keys are always available in the encrypted form. In our VL-PRE process, when a client (data owner) sends any requests (for key generation, update, and renewal) to the proxy, the client's public key certificate is required for the authentication.

6. Evaluation and Experiments

We evaluate the efficiency of our VL-PRE by focusing on the revocation case that causes the re-encryption of data outsourced in the cloud. We perform a comparative analysis of our proposed VL-PRE and the recent optimized PRE schemes proposed in [11] and [13]. We also conduct the experimental scenarios to measure the performance and throughput between these three schemes.

Table 3 Comparison of PRE characteristics

Characteristics	Tysowki and Hasan scheme [11]	Liang et al. scheme [13]	VL-PRE
Offloading Re-encryption key generation to cloud	Partially support	Partially support	Fully support
Re-encryption key Size	No. of attributes contained in two ciphertexts	No. of attributes contained in ciphertext	2 identity attributes
Cryptographic Operation (for constructing re-encryption key)	Attribute-based encryption	Identity-based encryption	Random encryption
Re-encryption Key generation computation cost	$O(1)$GSK + $O(1)$ $RK_{0\rightarrow x}$	$O(1)C_e$	$O(1)rk_{new}$
Frequency of Re-encryption key generation	Upon the revocation/Policy Update	Upon the revocation/Policy Update	Upon the Key Expiry

6.1 Comparative Analysis of Existing Schemes

We conduct comparative analysis of the attribute-based access control revocation schemes of VL-PRE and related works including Tysowki and Hasan scheme [11], and Liang et al. scheme [13]. [11] and [13] are chosen to compare with our scheme since they share a similar goal in optimizing the PRE cost but using different methods. From Table 3, we can see that both [11] and [13] still need a client to compute or update the re-encryption key in the PRE process. VL-PRE fully offloads re-encryption key generation to the proxy. Thus, it eliminates the computation cost at client side. Besides, the size of re-encryption key of both [11] and [13] is subject to the number of attributes contained in the ciphertext they use to compute or update the re-encryption key, while VL-PRE contains only fixed two attributes accommodated in the RDK. Therefore, VL-PRE delivers less decryption cost compared to those two schemes.

To complete re-encryption key generation used for ciphertext re-encryption, [11] suffers from high computation complexity that needs two computation steps from both client in computing secret group key (GSK) and proxy in computing re-encryption key $RK_{0\rightarrow x}$. Also, the GSK will be delivered to all active users which introduce additional communication cost.

As shown in Table 3, [13] and VL-PRE provide least complexity for re-encryption key computation since they require single computation with $O(1)$ to generate the re-encryption key. However, in [13] the computation cost is subject to the complexity of bilinear pairing and exponentiation C_e. Also, the cost is proportional to the number of attributes contained in the ciphertexts used in computing the key. In our scheme, the RDK is pre-computed and the computation cost for securing the re-encryption key is based on random encryption which is a kind of symmetric key encryption.

In essence, symmetric key encryption provides less computation cost than pairing-based cryptography requiring exponentiation in generating the re-encryption key [22].

Furthermore, both [11] and [13] expose the computational cost for re-generating the re-encryption key upon the attribute revocation or policy update. In contrast, VL-PRE uses key update strategy instead of key re-generation. With this strategy, the re-encryption key will be re-generated periodically with respect to the key changeover policy specified by the data owner. Therefore, VL-PRE provides less computation cost compared to those schemes using naive PRE key generation.

6.2 Performance Evaluation

Our evaluation focuses on minimizing the PRE computation cost and improving the scalability of VL-PRE compared to existing lightweight PRE schemes.

To this end, we conduct the simulation to measure the processing time of individual cost of three VL-PRE phases. To demonstrate the improvement of PRE cost optimization and scalability of VL-PRE over existing PRE schemes, we also set up the experiments to measure the PRE performance and server throughput of VL-PRE and two related works.

6.2.1 PRE Performance Evaluation

In our simulation environment, the proxy is run on Intel(R)Xeon(R)-CPU E5620, 2.40 GHz. The simulation environment on client end is run on MacBook Pro Intel Core i5 Dual-core, 2.7 GHz.

- **Performance of individual VL-PRE phase**

We first measure the processing time of three phases of VL-PRE including re-encryption key generation, re-encryption key update, and re-encryption key renewal. In the experiment, we use ACP containing 10 attributes and random number generator library [17] which are parts of a re-encryption key. Table 4 presents the processing time used to execute the individual algorithm of VL-PRE.

From Table 4, we can see that key update algorithm takes least time in updating the re-encryption key and it significantly reduces the computation and communication cost if there is an update of policy that requires the re-generation of re-encryption key. The re-encryption key renewal enjoys the less computation cost compared to the initial re-encryption key generation. This is because it avoids comput-

Table 4 VL-PRE Algorithm performance

VL-PRE Operations	Processing Time (ms.)
Re-encryption key generation	88
Re-encryption key update	31
Re-encryption key renewal	45

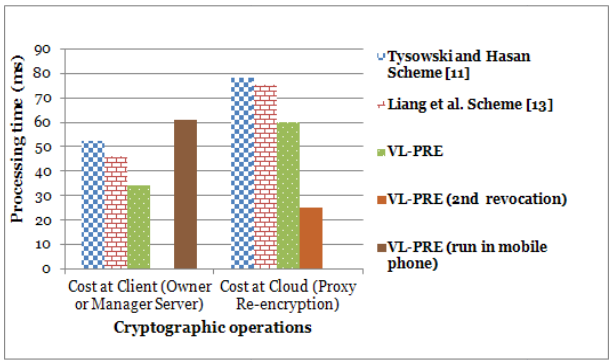

Fig. 4 Comparison of PRE Cost

ing basic system parameters as the re-encryption key generation does.

- **Performance of entire PRE cost occurring at a client side and a proxy side.**

In this section, we compare the PRE performance of VL-PRE with [11] and [13]. For the experiment, we developed a custom program and implement the algorithms of [11] and [13] by using Java pairing-based cryptography (jPBC) [19] to measure the performance of re-encryption task.

For the evaluation, the processing time for the client setup and re-encryption process of all schemes is measured. In our simulation, we also measure the PRE setup cost by using mobile phone. We use Android 6.0 SDK and employ secureRandom Java class in [17] to build a custom simulation program of cryptographic secure random number generator in Android OS. Samsnug Galaxy Note II smartphone with a quad-core 1.6 GHz ARM with 2 GB RAM is used to test the performance of PRE setup cost initiated by the client (data owner).

In our simulation, the ciphertext size is 50 KB and the policy size contains 20 attributes. The group secret key (GSK) used in [11] and a user secret key (sk_{id}) used in [13] contain 10 attributes. From Fig. 4, all schemes require client (data owner or manager) to deal with re-encryption key generation for the setup cost. In VL-PRE, the time used at both the client and the proxy is less than [11] and [13]. As for the setup cost, VL-PRE requires the client to prepare the key construction parameters and then fully offload re-encryption key generation task to the proxy.

In [11] and [13], the re-encryption key is partially computed at client before it is sent to the proxy for executing the process of ciphertext re-encryption. The scheme proposed in [11] takes more processing time than [13] as it requires two computation tasks in computing group secret key and re-encryption key.

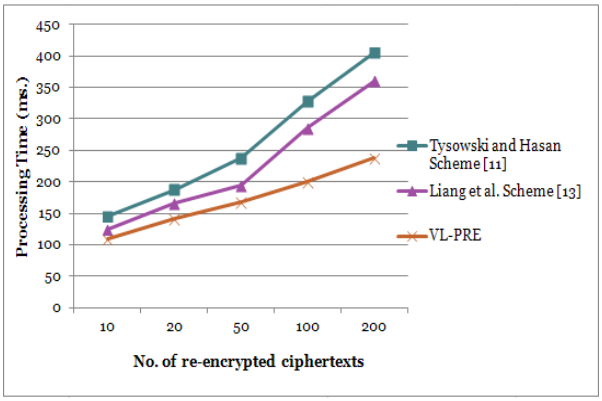

Fig. 5 PRE Performance in supporting multiple ciphertexts

To demonstrate the efficiency of VL-PRE in supporting the revocation through client mobile device, a mobile phone is used to run the setup algorithm of PRE. As around 60 ms for the set up time run by the mobile phone, it demonstrates that our VL-PRE is proven to be feasible and efficient in supporting mobile revocation management through less-constrained device.

Regarding the ciphertext re-encryption cost performed by the proxy, VL-PRE uses a smaller key size. Hence, it provides better performance for ciphertext re-encryption than [11] and [13]. With a single ciphertext to be re-encryption, [13] takes a bit less PRE processing time than [11] since the decryption based on the identity-based encryption using fewer identity attributes is faster than attribute-based encryption.

For the 2nd round of attribute revocation, it is assumed that the same PRE cost will occur for both [11] and [13], while VL-PRE offers no re-encryption key generation cost. Here, the valid re-encryption key can be used to re-encrypt the ciphertext.

- **Performance of PRE Server in supporting multiple ciphertexts**

We also compare the PRE performance of three schemes when they support multiple ciphertexts to be re-encrypted. For the simulation, the average size of all ciphertexts used in the test is 20 KB. Figure 5 shows the results of PRE processing time (ms) when there is an increased number of ciphertexts to be re-encrypted.

As seen from Fig. 5, both [11] and [13] suffer from high computation cost of re-encryption key when there is an increased number of ciphertexts to be re-encrypted. This is because the computation of re-encryption keys is subject to the number of ciphertexts to be re-encrypted. In [11], the computation of re-encryption key is equal to the number of chiphertexts to be re-encrypted, while [13] requires the computation of all re-encryption keys for each revocation. Hence, [11] delivers more rounds of computation than [13]. In contrast, our VL-PRE applies only one re-encryption key for all affected ciphertexts since the RDK contains two ma-

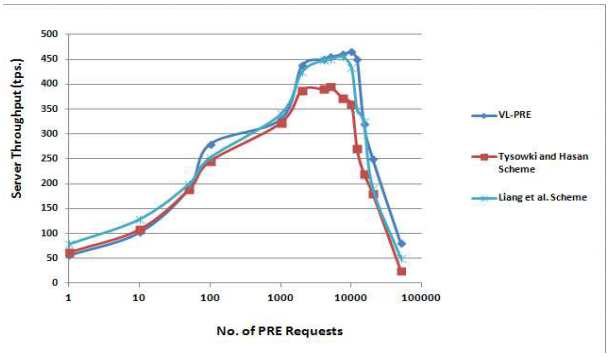

Fig. 6 Comparison of PRE Server Throughput

jor attributes (owner id and owner's digital signature) that are sufficient to decrypt all ciphertext stored in the cloud. Therefore, VL-PRE outperforms the compared works as it takes least processing time when it processes a high number of re-encrypted ciphertexts.

6.2.2 Throughput Measurement

To demonstrate the scalability of the proposed scheme, we conduct the experiment to test the throughput of the proxy server in serving multiple re-encryption requests. The policy used for the test contains 20 attributes and it is used for encrypting 1 MB file. Figure 6 presents the comparison of the PRE throughput of three schemes.

The graph shows that VL-PRE yields higher throughput in transaction per second (tps) of PRE operation than [11] and [13]. According to the result, the maximum throughput of VL-PRE was about 465 tps with 10,000 request threads, while [11] and [13] can best support approximately 5,000 and 7,500 request threads respectively. VL-PRE can tolerate around upto 12,000 requests before it declines. The result confirms that our new scheme provides a more scalability in processing concurrent workloads at the cloud side compared to recent PRE schemes.

7. Conclusion

The attribute-based proxy re-encryption scheme called VL-PRE (Very Lightweight Proxy Re-encryption) has been proposed as an optimized PRE scheme focusing on reducing computation cost at both client and cloud side. The proposed VL-PRE scheme introduces three-phase PRE where the key update and key renewal optimize the overall PRE cost at both client and proxy side. In addition, the new scheme is designed to minimally carry the re-encryption key generation data package of which the decryption key size is reduced. Finally, we conduct the experiments by doing the simulations to evaluate the performance of our proposed VL-PRE. The results confirm that VL-PRE delivers more performance and scalability for supporting attribute revocation than the existing PRE schemes. For future works, we will perform extensive experiments to evaluate the performance of VL-PRE in real cloud systems such as OpenStack or Cloudstack.

References

[1] K. Yang, X. Jia, K. Ren, B. Zhang, and R. Xie, "DAC-MACS: Effective data access control for multiauthority cloud storage systems," IEEE Trans. Inf. Forensics Security, vol.8, no.11, pp.1790–1801, 2013.

[2] J. Son, D. Kim, R. Hussain, and H. Oh, "Conditional proxy re-encryption for secure big data group sharing in cloud Environment," Proc. IEEE INFOCOM Workshop on Security and Privacy in Big Data, Hong Kong, pp.541–546, April 2014.

[3] H. Xiong, X. Zhang, D. Yao, X. Wu, and Y. Wen, "Towards End-to-End Secure Content Storage and Delivery with Public Cloud," Proc. of ACM CODASPY 2012, Texas, USA, Feb. 2012.

[4] Z. Wan, J. Liu, and R.H. Deng, "HASBE: A Hierarchical attribute-based solution for flexible and scalable access control in cloud computing," IEEE Trans. Inf. Forensics Security, vol.7, no.2, pp.743–754, 2012.

[5] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," Proc. IEEE Symposium of Security and Privacy (S&P 2007), Oakland, CA, USA, pp.321–334, May 20-23, 2007.

[6] M. Mambo and E. Okamoto, "Proxy cryptosystems: delegation of the power to decrypt ciphertexts," IEICE Trans. Fundamentals, vol.E80-A, no.1, pp.54–63, Jan. 1997.

[7] X. Liang, Z. Cao, H. Lin, and J. Shao, "Attribute based proxy re-encryption with delegating capabilities," ASIACCS, pp.276–286, ACM, 2009.

[8] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," ACM Trans. Inf. Syst. Secur., vol.9, no.1, pp.1–30, 2006.

[9] M. Chase and S.S.M. Chow, "Improving privacy and security in multi-authority attribute-based encryption," Proc. 16th ACM Conference on Computer and Communications Security (CCS'09), ACM, 2009.

[10] A.B. Lewko and B. Waters, "Decentralizing attribute-based encryption," Proc. 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology, EUROCRYPT'11, vol.6632, pp.568–588, Springer 2011.

[11] P.K. Tysowski and M.A. Hasan, "Hybrid attribute- and re-encryption-based key management for secure and scalable mobile applications in clouds," IEEE Trans. Cloud Computing, vol.1, no.2, pp.172–186, 2013.

[12] S. Fugkeaw and H. Sato, "An extended CP-ABE based Access control model for data outsourced in the cloud," Proc. IEEE 39th Annual Computer Software and Applications (COMPSAC 2015), Taiwan, pp.73–78, 2015.

[13] K. Liang, J K. Liu, D.S. Wong, and W. Susilo, "An efficient cloud-based revocable identity-based proxy re-encryption scheme for public clouds data sharing," Proc. European Symposium Research in Computer Security, Wroclaw, Poland, pp.257–272, Springer, Sept. 2014.

[14] Y. Kawai, "Outsourcing the re-encryption key generation: flexible ciphertext-policy attribute-based proxy re-encryption," Proc. International Conference on Information Security Practice and Experience (ISPEC 2015), Beijing, China, pp.301–315, Springer, May 2015.

[15] J. Lai, R.H. Deng, Y. Yangj, and J. Weng, "Adaptable ciphertext-policy attribute-based encryption," Proc. of Pairing 2013. LNCS, vol.8365, pp.199–214. Springer, Heidelberg, 2014.

[16] S. Fugkeaw and H. Sato, "Embedding lightweight proxy re-encryption for efficient attribute revocation in cloud computing," International Journal on High Performance Computing and Networking, vol.9. no.4, pp.299–309, 2016.

[17] Secure random number generator Java library, https://docs.oracle.com/javase/7/docs/api/java/security/SecureRandom.html

[18] P. Angin, B. Bhargava, and Z. Jin, "A self-cloning agents based

model for high performance mobile cloud computing," Proc. IEEE International Conference on Cloud Computing (CLOUD'15), 2015 New York, USA, pp.301–308, June 2015.
[19] Java Paring -Based Cryptography Library (JPBC), http://gas.dia.unisa.it/projects/jpbc/download.html
[20] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol.2139, pp.213–229, Springer, Heidelberg, 2001.
[21] National Institute of Standards and Technology (NIST), Security Requirements for Cryptographic Modules, FIPS-PUB 140-2, 2001, Available from http://csrc.nist.gov/publications
[22] W.C. Garrison III, A. Shull, S. Myers, and A.J. Lee, "On the practicality of cryptographically enforcing dynamic access control policies in the cloud," Proc. IEEE Symposium on Security and Privacy (S&P 2016), San Jose, CA, USA, pp.819–838, May 23-25, 2016.

Somchart Fugkeaw is currently a Ph.D. candidate with the Department of Electrical Engineering and Information Systems, University of Tokyo. His research interests include security and privacy in cloud computing, access control models, and PKI.

Hiroyuki Sato is currently an Associate Professor with the Department of Electrical Engineering and Information Systems, University of Tokyo. His research interests include programming language, trust, information security, and optimization.