Journal of Biomedical Informatics 50 (2014) 226–233

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

FRR: Fair remote retrieval of outsourced private medical records in electronic health networks

Huaqun Wang[a,*], Qianhong Wu[b], Bo Qin[c], Josep Domingo-Ferrer[d]

[a] School of Information Engineering, Dalian Ocean University, China
[b] School of Electronic and Information Engineering, Beihang University, China
[c] School of Information, Renmin University of China, China
[d] Universitat Rovira i Virgili, Department of Computer Engineering and Mathematics, UNESCO Chair in Data Privacy, E-43007 Tarragona, Catalonia, Spain

ARTICLE INFO

Article history:
Received 13 August 2013
Accepted 8 February 2014
Available online 18 February 2014

Keywords:
Cloud computing
Medical records
Fair remote data retrieval

ABSTRACT

Cloud computing is emerging as the next-generation IT architecture. However, cloud computing also raises security and privacy concerns since the users have no physical control over the outsourced data. This paper focuses on fairly retrieving encrypted private medical records outsourced to remote untrusted cloud servers in the case of medical accidents and disputes. Our goal is to enable an independent committee to fairly recover the original private medical records so that medical investigation can be carried out in a convincing way. We achieve this goal with a fair remote retrieval (FRR) model in which either t investigation committee members cooperatively retrieve the original medical data or none of them can get any information on the medical records. We realize the first FRR scheme by exploiting fair multi-member key exchange and homomorphic privately verifiable tags. Based on the standard computational Diffie–Hellman (CDH) assumption, our scheme is provably secure in the random oracle model (ROM). A detailed performance analysis and experimental results show that our scheme is efficient in terms of communication and computation.

1. Introduction

Cloud servers (CS) can provide shared resources, software and information to remote clients over a network such as Internet. The clients can access the CS through a web browser or a smartphone application, while the business software and data are stored on the CS at a remote location. An important application of cloud computing is to store medical documents from electronic health networks (EHNs) in order to facilitate the exchange of health information between healthcare organizations, clinical research organizations, healthcare payers and public health agencies. Each day, an EHN generates and records numerous medical documents, including medical images, medical check reports, clinical reports, medical prescriptions, patient monitoring reports, and so on [1,2]. These documents are sensitive and need to be protected, so that only authorized parties can access them. It is preferable to employ cloud servers to store and maintain such a large number of medical documents, because the cloud reduces cost and offers universal accessibility.

Cloud computing poses some security problems: remote data integrity checking, remote data retrieval and data privacy. For example, CS may neglect keeping or may deliberately delete rarely accessed data of some clients. Thus, it is important to enable the client to easily check the integrity of the outsourced data without downloading the entire database. Furthermore, the remotely stored data may need to be used in the future; therefore, the client must have the ability to retrieve his/her data anywhere anytime. To achieve data privacy, a general approach is to let the clients encrypt their data before uploading them to CS. The encryption can be elegantly designed to allow searching encrypted data or retrieving data with specific content. To improve the robustness of data retrieval, error-correcting codes can be used before the data are stored at the CS.

1.1. Related work

The provable data possession (PDP) model, introduced by Ateniese et al. [3], performs a probabilistic check and is designed to catch a misbehaving CS deleting or altering a file outsourced by a client. In [3] Ateniese et al. also instantiated two non-dynamic PDP schemes that are provably secure in the random oracle model. Subsequently, they proposed a dynamic PDP security model and

* Corresponding author. Fax: +86 411 84763996.
E-mail address: wanghuaqun@aliyun.com (H. Wang).

http://dx.doi.org/10.1016/j.jbi.2014.02.008
1532-0464/© 2014 Elsevier Inc. All rights reserved.

realized a concrete PDP scheme [4] supporting data updates except the insert operation. Fully dynamic PDP was realized by Erway et al. based on an authenticated flip table [5]. Independently, Sebé et al. designed an efficient PDP scheme by using the problem of factoring large integers [6]. Since then, several variants of the original PDP models and new instantiations have been proposed. Wang et al. studied proxy PDP [7], cooperative PDP [8] and identity-based PDP [9,10]. Zhu et al. proposed the concept of cooperative PDP for hybrid clouds [11,12]. Curtmola et al. proposed the multiple-replica PDP [13]. Yu et al. studied the security and privacy-preserving problems in remote data integrity checking [14,15].

Proofs of Retrievability (PoRs) is an enhanced model to guarantee that the clients can retrieve their remotely stored data. The above PDP model ensures that the clients will catch with high probability a misbehaving CS who modified the clients' files. However, secure remote data storage is only part of the client's requirements. The clients must be able to retrieve the remote data no matter whether the misbehaving CS was caught or not. This gap is filled by the PoRs model [16,17] that ensures that the clients can definitely retrieve their data. In the PoRs model, CS proves to a client that it is actually storing the entire data of the client and the client can retrieve the data anytime anywhere. In a multi-client setting, Zheng et al. more recently proposed a fair and dynamic PoRs model and a corresponding scheme [18]. Since then, several variants of the original PoRs models and new instantiations have been proposed. Bower et al. concluded the theory and implementation of PoRs [19]. Dodis et al. proposed novel PoRs via hardness amplification [20]. Zhu et al. proposed the zero-knowledge proofs of retrievability [21].

Efforts have been also devoted to privacy and management of outsourced data. Wan et al. proposed hierarchical attribute-set-based encryption by extending ciphertext-policy attribute-set-based encryption with a hierarchical structure of users [22]. The encryption approach can be added to PoRs to simultaneously achieve data privacy and retrievability [23].

1.2. Motivation and contribution

In this paper, we study the security and privacy concerns in medical investigation in EHNs. In such a scenario, numerous medical records are encrypted and stored in a CS. Unlike in other cloud storage applications, here the CS, the hospital and the patients may be interested in dishonestly modifying the outsourced medical data. Hence, an independent third party is employed to guarantee the integrity and the retrievability of the outsourced data. Further, when medical investigation is performed in the case of medical accidents, the third party should be able to recover the original medical records in a fair way. We argue that an independent third party is necessary since both the hospital and the patients might tamper with the original medical records or refuse to decrypt the encrypted records if the original records could be used as arguments against them in a dispute.

Motivated by the above application, we present a fair remote retrieval (FRR) model in EHNs. We capture the data privacy, retrievability and fairness requirements in such applications. To achieve data privacy, private medical records need to be encrypted before they are outsourced to cloud servers so that curious CS cannot understand the content of the stored data. In FRR, t of n cooperating medical committee members should be able to check whether the data are kept intact or not. Further, the t cooperating members should be also able to recover the original data if necessary; however, if less than t members collude, they must not be able to obtain any information about the outsourced data, which guarantees fairness in medical investigation in the case of medical accidents or disputes.

1.3. Organization

The rest of the paper is organized as follows. Section 2 states the problem to be investigated and defines the system model. Section 3 describes our FRR scheme and proves its security. Section 4 evaluates our scheme in terms of computation and communication cost. Section 5 concludes the paper.

2. System model and security requirements

In this section, we present a functional FRR model in EHNs (Section 2.1) and then formally define its security requirements (Section 2.2). Our FRR system considers the following scenario: a hospital uploads the patients' private medical records to a CS maintained by a cloud service provider; the provider is not assumed to be trustable; specifically, the provider may be curious and occasionally misbehave, *e.g.*, by deleting or modifying some parts of the medical records; a medical committee of n members is employed to validate whether the records have been kept intact; the committee can also recover the records if required.

2.1. FRR system model

An FRR system, as shown in Fig. 1, comprises five types of participants: Patient, Hospital, CS, Committee and Semi-Trusted Neutral Party (STNP). They are described as follows.

1. Patient: When a patient goes to the hospital for health treatment, private medical records are generated and encrypted by the patient.
2. Hospital: The hospital helps the patient to create some private medical records. After receiving the patient's encrypted medical records, the hospital uploads them to the CS.
3. CS: CS has significant storage space and computational resources to maintain the data from the hospitals. However, CS may also occasionally misbehave and modify some parts of the stored data.
4. Committee: This is the third party trusted by the patient and the hospital to solve a medical tangle; this is a situation in which the patients (or their family members) query the hospital's course of treatment and are not satisfied with the hospital's reply. In such a situation, the patient and the hospital may appeal to the committee of medical tangle, which consists of n independent members who do not necessarily trust each other. At least t members must cooperate in a fair exchange in order to recover the original patient data.
5. STNP: The Semi-Trusted Neutral Party (STNP) is selected on a case-by-case basis and asked to aid in the execution of a fair multi-member exchange among the independent committee members. STNP is trusted to ensure the fairness of the multi-member exchange, that is, all members trust it to honestly

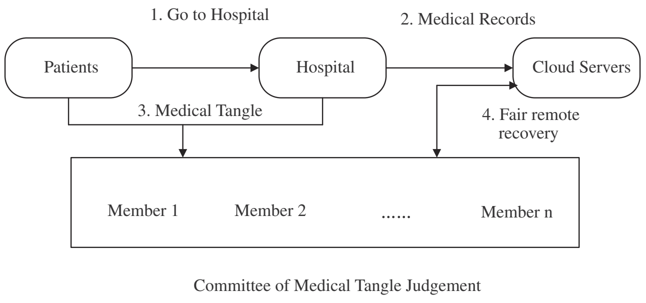

Fig. 1. A medical investigation system.

process the received messages and respond to the corresponding members. However, it is not assumed to be fully trusted by the members. In particular, STNP is not allowed to see the final exchanged secret shares of the members; hence, a malicious STNP cannot get the private medical records as long as the number of dishonest members is less than $t-1$. The members can request STNP's intervention in case of dispute. If there is no dispute, STNP stays offline. If there exists a dispute, STNP stays online.

Definition 1 (*FRR scheme*). An FRR scheme among untrusted members, CS, hospital and patient, and utilizing STNP, is a collection of eight probabilistic polynomial-time algorithms (*KeyGen*, *TagGen*, *CheckTag*, *GroupBarter*, *GenProof*, *GenRetrieval*, *Verify*, *Extract*) such that:

1. $KeyGen(1^k) \rightarrow (pk, sk)$ is a probabilistic polynomial-time key generation algorithm, where k is the security parameter. CS runs the algorithm *KeyGen* to obtain its own pair (y, Y) of private/public keys. The hospital runs the algorithm *KeyGen* to obtain its own pair (z, Z) of private/public keys. On the other hand, the patient runs the algorithm *KeyGen* to get a pair (s, v) of matching private/public keys. Then the patient distributes the shares $s_1, s_2, \ldots, s_n$ of the private key s to the corresponding committee members. This phase produces the system parameters which are denoted by *param*.
2. $TagGen(s, m) \rightarrow T_m$ is a polynomial-time algorithm run by the patient to generate the verification metadata. On input the private key s and a file block m, it returns the verification tag T_m. Then, with the help of the hospital, the patient uploads the block-tag pairs (m, T_m) to CS. Some corresponding metadata will be kept by the patient and the hospital.
3. $CheckTag(m_i, T_i, param, Y, Z, v) \rightarrow \{\text{"success"}, \text{"failure"}\}$ is a polynomial-time algorithm run by CS to check whether a block-tag pair is valid. It takes as input a block-tag pair (m_i, T_i) and the public keys of CS, the hospital and the patient; it returns "success" or "failure". "Success" denotes that the pair is valid and "failure" denotes that it is invalid.
4. $GroupBarter(s_{1'}, \ldots, s_{t'}) \rightarrow s$ is a polynomial-time algorithm fairly run by t committee members with the help of STNP. It takes as input t secret shares $s_{1'}, \ldots, s_{t'}$ which are distributed in the phase *KeyGen*, and returns the final private key s. Fair running ensures that either the t participating members obtain the private key s or no member obtains it. s can be used to decrypt the patient's private records.
5. $GenProof(F, chal, \Sigma) \rightarrow V$ is run by CS in order to realize the remote data integrity checking and the retrieval protocol. It takes as inputs an ordered collection F of blocks, a challenge *chal* and an ordered collection Σ of verification metadata which corresponds to the blocks in F. It returns V for the blocks in F that are determined by the challenge *chal*.
6. $GenRetrieval(F, Rchal, \Sigma) \rightarrow V$ is run by CS. It takes as inputs an ordered collection F of blocks, a challenge *Rchal* and an ordered collection Σ of verification metadata which corresponds to the blocks in F. It returns V that corresponds to the challenge *Rchal*.
7. $Verify((z, Z), chal, V) \rightarrow b \in \{0, 1\}$ is run by the hospital to check whether the response V is valid. "1" denotes valid; "0" denotes invalid.
8. $Verify\text{-}Retrieve(s_{1'}, s_{2'}, \ldots, s_{t'}, Rchal, V) \rightarrow M$ is run by the members of the medical tangle expert committee to retrieve the remote private medical records M. It takes as inputs the challenge *Rchal*, the shares $(s_{1'}, s_{2'}, \ldots, s_{t'})$ of the patient's private key s, and the corresponding response V. The members verify the response V and get the remote private medical records M if successful.

Note. The step Verify-Retrieve can also be performed by the patient since it owns the private key s.

An FRR scheme should be correct. *Correctness* requires that, for the keypairs $(y, Y), (z, Z)$ and (s, v) output by *KeyGen*, for all file block-tag pairs (m_i, T_i) output by *TagGen*, the verification algorithm accepts the response V when interacting with the honest CS.

2.2. Security requirements in FRR

We now formally define the security properties: unforgeability against untrusted CS, retrievability, and threshold fairness among the untrusted clients.

Definition 2 (*Unforgeability against untrusted CS*). An FRR scheme is unforgeable against any probabilistic polynomial-time untrusted CS if the probability that any such CS wins the FRR game below is negligible. We denominate the untrusted CS as the adversary $\mathcal{A}$. The FRR game is described below:

1. Setup: The challenger is denoted by $\mathcal{C}$. $\mathcal{C}$ obtains CS's public–private key pair $(pk_s, sk_s) \leftarrow KeyGen(1^k)$. It obtains the *Client*'s public–private key pair $(pk_c, sk_c) \leftarrow$ KeyGen (1^k). Then, $\mathcal{C}$ sends $(params, pk_s, sk_s, pk_c)$ to $\mathcal{A}$ (*i.e.*, untrusted CS) and keeps the private key sk_c confidential.
2. First-Phase Queries: $\mathcal{A}$ adaptively makes different queries to $\mathcal{C}$ as listed below:
   - Hash queries. $\mathcal{A}$ makes hash queries adaptively and $\mathcal{C}$ responds to them.
   - Tag queries. $\mathcal{A}$ makes block-tag pair queries adaptively. For a query m_i issued by $\mathcal{A}$, $\mathcal{C}$ computes the tag $T_i \leftarrow \mathsf{TagGen}(sk_c, m_i)$ and sends it back to $\mathcal{A}$. Without loss of generality, let (m_i, T_i) be the queried block-tag pairs for index $i \in \mathbb{I}_1$.
3. Challenge: $\mathcal{C}$ generates a challenge *chal* which defines an ordered queried index collection $\{i_1, i_2, \ldots, i_c\}$, where $\{i_1, i_2, \ldots, i_c\} \not\subseteq \mathbb{I}_1$ and c is a positive integer. In the index set $\{i_1, i_2, \ldots, i_c\}$, at least there exists one corresponding $m_{i_j} (1 \leqslant j \leqslant c)$ that has never been queried by $\mathcal{A}$.
4. Second-Phase Queries: Similar to the First-Phase Queries. Suppose that (m_i, T_i) is the queried and answered block-tag pair for index $i \in \mathbb{I}_2$, where $\mathbb{I}_2$ is the queried and answered block-tag pair index set. The restriction is that $\{i_1, i_2, \ldots, i_c\} \not\subseteq \mathbb{I}_1 \cup \mathbb{I}_2$.
5. Forge: $\mathcal{A}$ computes an integrity proof V for the blocks indicated by *chal* and returns V to $\mathcal{C}$.

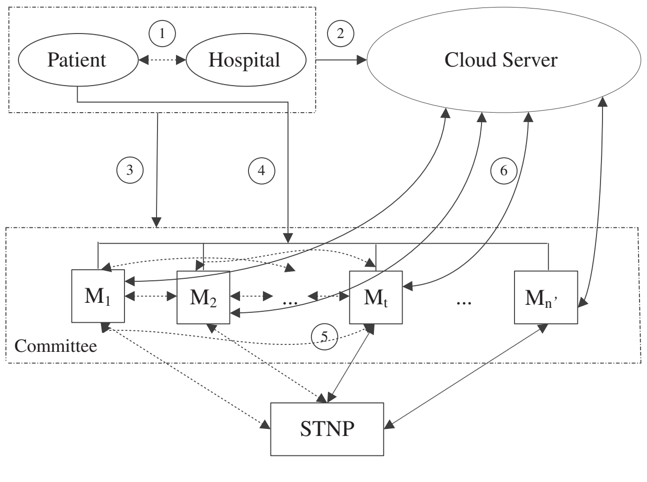

Fig. 2. FRR scheme architecture.

We say that FRR satisfies unforgeability against the untrusted CS if the adversary $\mathcal{A}$ wins the FRR game with negligible probability.

Now, we define the property of retrievability where the adversary $\mathcal{A}$ is still an untrusted CS. For $\epsilon^{\mathcal{A}} \leqslant \epsilon$, an ϵ-adversary $\mathcal{A}$ replies correctly to a fraction of at least $1 - \epsilon^{\mathcal{A}}$ challenges over the lifetime of the file.

Definition 3 ((ϵ, γ)-*Retrievability*). An FRR scheme against the untrusted cloud is an (ϵ, γ)-valid proof of retrievability if for every polynomial-time ϵ-adversary $\mathcal{A}$ (*i.e.*, untrusted cloud), the probability that Verify-Retrieve outputs the correct stored file M is γ.

Definition 4 (*Threshold fairness of FRR with STNP among untrusted members*). Suppose the set of committee members is S and the number of members is $|S| = n'$. The threshold value is t. The FRR scheme with STNP against untrusted members satisfies the property of (t, n')-threshold fairness if it satisfies the following conditions.

1. Any t or more honest members can cooperate to check and retrieve the remote data. If there exist dishonest members among the t members, the t members cannot check or retrieve the remote data.
2. Any $t - 1$ or less members cannot check or retrieve the remote data.

3. Proposed FRR scheme for untrusted clients and CS

In this section, we recall some mathematical background (Section 3.1) and then we present our pairing-based FRR scheme to fairly protect the hospital and the patients in the case of medical accident investigation (Section 3.2). Then, in Section 3.3, we analyze the correctness and security of our scheme, which is shown to satisfy provable security in the ROM.

3.1. Basic mathematical background

Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two cyclic multiplicative groups with the same prime order q, *i.e.*, $|\mathcal{G}_1| = |\mathcal{G}_2| = q$. Let $e: \mathcal{G}_1 \times \mathcal{G}_1 \rightarrow \mathcal{G}_2$ be a bilinear map which satisfies the following properties: (1) Bilinearity: $\forall Q, R, S \in \mathcal{G}_1$ and $a, b \in \mathcal{Z}_q$, $e(R^a, Q^b) = e(R, Q)^{ab}$. (2) Non-degeneracy: $\exists Q, R \in \mathcal{G}_1$ such that $e(Q, R) \neq 1_{\mathcal{G}_2}$. (3) Computability: $\forall Q, R \in \mathcal{G}_1$, there is an efficient algorithm to calculate $e(Q, R)$.

We can construct such a bilinear map e based on the modified Weil pairings [24] or Tate pairings [25] on elliptic curves over finite fields. A group with such a map e is called a bilinear group. Our scheme is proven to be secure under some well-understood computational difficulty assumptions in the random oracle model.

Let P be the generator of $\mathcal{G}_1$. Now, given unknown $a, b, c \in \mathcal{Z}_q$ and $P, P^a, P^b, P^c \in \mathcal{G}_1$, it holds that there exists an efficient algorithm to determine whether $ab = c \bmod q$ by verifying $e(P^a, P^b) = e(P, P)^c$ in polynomial time (Decisional Diffie-Hellman or DDH problem), while there exists no efficient algorithm to compute $P^{ab} \in \mathcal{G}_1$ with non-negligible probability within polynomial time (Computational Diffie–Hellman or CDH problem); see [26] for details. A group where DDH is easy and CDH is hard is called a co-GDH group; hence, $\mathcal{G}_1$ is a co-GDH group.

3.2. Scheme construction

In order to give the intuition of our FRR scheme construction, its architecture is given in Fig. 2. There are five entities: patient, hospital, cloud server, committee and STNP. Parameters t and n' are determined by the patient, the committee and STNP. STNP is used for the multi-member fair exchange. The architecture can be outlined as follows:

① The patient goes to the hospital for health treatment. With the help of the hospital, he gets his own private medical records. The patient encrypts these medical records.
② The hospital uploads these encrypted private medical records to the cloud server.
③ A medical tangle happens between the patient and the hospital. The patient and the hospital appeal to the committee.
④ The patient distributes the secret shares $\{s_1, s_2, \ldots, s_{n'}\}$ to all committee members $\{M_1, M_2, \ldots, M_{n'}\}$.
⑤ Without loss of generality, we let the threshold be t and the active member subset be $\{M_1, M_2, \ldots, M_t\}$. The entities $\{STNP, M_1, M_2, \ldots, M_t\}$ perform the threshold fair multi-member key exchange.
⑥ The members $M_1, M_2, \ldots, M_t$ interact with the cloud server and retrieve the remote medical records.

We next describe our scheme in detail and start by introducing some additional notation used in the construction. Suppose the maximum number of block-tag pairs is n. Let f and w be two pseudo-random functions. Let H, h, H_1 and h_1 be cryptographic hash functions. The functions mentioned in this paragraph can be defined as follows:

$$f: \mathcal{Z}_q^* \times \{1, 2, \ldots, n\} \rightarrow \mathcal{Z}_q^*, \quad \omega: \{1, 2, \ldots, n\} \rightarrow \mathcal{Z}_q^*, \quad H: \mathcal{G}_2 \rightarrow \mathcal{Z}_q^*, \quad H_1: \{0, 1\}^* \rightarrow \mathcal{Z}_q^*, \quad h: \mathcal{Z}_q^* \rightarrow \mathcal{G}_1^*, \quad h_1: \{0, 1\}^* \rightarrow \mathcal{Z}_q^*.$$

Let $f_z(i)$ denote the function f with input i and trapdoor z. Let g be a generator of $\mathcal{G}_1$ and $\bar{g}$ be a generator of $\mathcal{Z}_q^*$, where q is the order of $\mathcal{G}_1$ and $\mathcal{G}_2$. The phases of our FRR scheme are described below.

KeyGen: *CS* picks a random number $y \in \mathcal{Z}_q^*$ as its private key and computes $Y = g^y$ as its public key. The hospital picks a random number $z \in \mathcal{Z}_q^*$ as its private key and computes $Z = g^z$ as its public key. The patient picks a random number $s \in \mathcal{Z}_q^*$ and computes $v = g^s$. Then, it picks a random polynomial $\hat{f}(x)$ of degree $t - 1: \hat{f}(x) = s + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1}$, where $a_i \in \mathcal{Z}_q^*$ and $1 \leqslant i \leqslant t - 1$. For n' different members $M_1, M_2, \ldots, M_{n'}$, the patient performs the following steps:

1. For $1 \leqslant i \leqslant n'$, compute $d_i = H_1(M_i)$ and $s_i = \hat{f}(d_i)$. Then, send s_i to the member M_i via a secure channel.
2. For $1 \leqslant i \leqslant n'$, compute $P_i = g^{s_i}$ and make them public.

Then, the patient picks a random element $u \in \mathcal{G}_1^*$ and its public key is $\{u, v\}$. The patient's private key is s and the private key shares are $\{s_i, 1 \leqslant i \leqslant n'\}$.

TagGen (s, F, i). Given the file F, the patient encrypts F by using a symmetric encryption algorithm E and the private key s of the patient. Suppose the ciphertext is $\widehat{F} = E_s(F)$. Then, the patient applies an erasure code on $\widehat{F}$ to obtain F'. Then, the patient splits F' into l blocks $F' = (\bar{m}_1, \bar{m}_2, \ldots, \bar{m}_l)$ for some positive integer l. To generate the tag T_i that corresponds to the block $\bar{m}_i$, the patient performs the following procedures:

1. Compute $m_i = h_1(\bar{m}_i)$, $\delta = H(e(Y, Z)^s)$ and generate $W_i = \omega_\delta(i)$.
2. Compute $T_i = (h(W_i) u^{m_i})^s$ and output (T_i, W_i).

As soon as the patient generates the l block tags, the hospital uploads all the block-tag pairs $\{(\bar{m}_i, T_i), 1 \leqslant i \leqslant l\}$ to CS.

CheckTag$(\{(m_i, T_i), 1 \leqslant i \leqslant l\})$. Given $\{(\bar{m}_i, T_i), 1 \leqslant i \leqslant l\}$, CS performs the following procedures for every $i, 1 \leqslant i \leqslant l$:

1. Compute $m_i = h_1(\bar{m}_i)$, $\hat{\delta} = H(e(v, Z)^y)$ and $\widehat{W}_i = \omega_{\hat{\delta}}(i)$.
2. Verify whether $e(T_i, g) \overset{?}{=} e(h(\widehat{W}_i)u^{m_i}, v)$ holds.
3. If it holds, it is valid and it is stored in CS. Otherwise, it is rejected.

Notes: In *CheckTag*, CS can perform the batch checking as follows. After receiving the l block-tag pairs $\{(\bar{m}_i, T_i),\ 1 \leqslant i \leqslant l\}$, CS picks l random $a_i \in \mathcal{Z}_q^*$ and computes $m_i = h_1(\bar{m}_i)$, $T = \prod_{j=1}^{l} T_j^{a_j}$, $m' = \sum_{j=1}^{l} a_j m_j$. If $e(T, g) \overset{?}{=} e\left(\prod_{j=1}^{l} h(\omega_\delta(j))^{a_j} u^{m'}, v\right)$ holds, these l block-tag pairs are accepted; otherwise, CS can get the invalid block-tag pairs by dichotomizing the search and then reject them.

GroupBarter. t members cooperate to perform fair exchange in order to check and retrieve remote private medical records. Let the t members' identity set be $\{M_{c_1}, M_{c_2}, \ldots, M_{c_t}\}$. They do as follows:

1. For $i \in \{c_1, c_2, \ldots, c_t\}$, M_i picks a random $r_i \in \mathcal{Z}_q^*$, computes $\hat{s}_i = s_i - r_i \bmod (q-1)$ and $R_i = \bar{g}^{r_i} \bmod q$, and makes R_i public.
2. For $i \in \{c_1, c_2, \ldots, c_t\}$, M_i sends $\hat{s}_i$ to STNP through a secure channel.
3. For $i \in \{c_1, c_2, \ldots, c_t\}$, M_i sends r_i to M_j, $j \neq i$ and $j \in \{c_1, c_2, \ldots, c_t\}$.
4. For $j \in \{c_1, c_2, \ldots, c_t\}$, when M_j receives $\hat{r}_i$, M_j checks whether $\bar{g}^{\hat{r}_i} \overset{?}{=} R_i \bmod q$ holds. If it holds, M_j accepts r_i and sends a sign $y_{ji} = 1$ to STNP through a secure channel; otherwise, M_j rejects r_i.
5. For $i \in \{c_1, c_2, \ldots, c_t\}$, STNP checks: $P_i \overset{?}{=} \bar{g}^{\hat{s}_i} R_i \bmod q$. If it holds, STNP accepts $\hat{s}_i$; otherwise, STNP rejects it.
6. When STNP receives the t responses $\{(\hat{s}_{c_1}, R_{c_1}), (\hat{s}_{c_2}, R_{c_2}), \ldots, (\hat{s}_{c_t}, R_{c_t})\}$ and $t(t-1)$ signs $\{y_{ji}, i, j \in \{c_1, c_2, \ldots, c_t\}, i \neq j\}$, it broadcasts $S = \{\hat{s}_{c_1}, \hat{s}_{c_2}, \ldots, \hat{s}_{c_t}\}$ to all the clients $\{M_{c_1}, \ldots, M_{c_t}\}$.
7. For $i \in \{c_1, c_2, \ldots, c_t\}, M_i$ computes $s_j = \hat{s}_j + r_j \bmod (q-1)$, where $j \neq i$. Then, by using the Lagrange interpolation method, M_i obtains

$$s = \hat{f}(0) = \sum_{j=1}^{t} s_{c_j} \left(\prod_{r=1, r\neq j}^{t} \frac{-H_1(M_{c_r})}{H_1(M_{c_j}) - H_1(M_{c_r})} \right) \bmod q$$

Thus, after *GroupBarter*, every member can get the private key s.

GenProof$(pk, \mathcal{M} = (\bar{m}_1, \bar{m}_2, \ldots, \bar{m}_l), chal, \Sigma = (T_1, \ldots, T_l))$. Let the hospital's integrity challenge be $chal = (l', k, i_1, i_2, \ldots, i_{l'})$. After receiving the challenge *chal*, CS performs the following steps:

1. For $1 \leqslant j \leqslant l'$, compute $a_j = f_k(j)$, and $m_{i_j} = h_1(\bar{m}_{i_j})$. Then, compute $T = \prod_{j=1}^{l'} T_{i_j}^{a_j}$, $m' = \sum_{j=1}^{l'} a_j m_{i_j}$.
2. Output $V = (T, m')$ to the hospital as the response to *chal*.

GenRetrieval$(pk, \mathcal{M} = (\bar{m}_1, \bar{m}_2, \ldots, \bar{m}_l), Rchal, \Sigma = (T_1, \ldots, T_l))$. Let the committee members' *Verify-Retrieve* challenge be $Rchal = \{i_1, i_2, \ldots, i_{l'}\}$. Then, CS sends $V' = \{(\bar{m}_{i_j}, T_{i_j}), 1 \leqslant j \leqslant l'\}$ to the members.

Verify$((pk, sk), chal, V)$. For the hospital's challenge-response pair $(chal, V)$, the hospital verifies the response V as follows:

1. Compute $\delta = H(e(v, Y)^z)$ and check: $e(T, g) \overset{?}{=} e\left(\prod_{j=1}^{l'} h(\omega_\delta(j))^{f_k(j)} u^{m'}, v\right)$.
2. If it holds, accept this response. Otherwise, reject it.

Verify-Retrieve$(pk, sk, Rchal, V')$. After receiving the medical tangle, the members of the committee query the medical records $Rchal = \{1, 2, \ldots, l\}$. Every member $M_i, i \in \{c_1, c_2, \ldots, c_t\}$ performs the following steps:

1. For $1 \leqslant j \leqslant l$, pick the random coefficients of the blocks $a_{i,j} \in \mathcal{Z}_q^*$ and compute $m_j = h_1(\bar{m}_j)$, $T_i = \prod_{j=1}^{l} T_j^{a_{i,j}}$, $m'_i = \sum_{j=1}^{l} a_{i,j} m_j$
2. Compute $\delta = H(e(Y, Z)^s)$ and check: $e(T_i, g) \overset{?}{=} e\left(\prod_{j=1}^{l} h(\omega_\delta(j))^{a_{i,j}} u^{m'_i}, v\right)$. If it holds, accept the records. Otherwise, reject them and exit the protocol.
3. Obtain $F' = (\bar{m}_1, \bar{m}_2, \ldots, \bar{m}_l)$. Then, get $\widehat{F}$ by eraser decoding on F'. After that, retrieve $F = D_s(\widehat{F})$ by using s to decrypt $\widehat{F}$, where D is the symmetric decryption algorithm that corresponds to E.

In this way, all the M_i, $i \in \{c_1, c_2, \ldots, c_t\}$ can fairly check and retrieve the remote private medical records M.

Notes: Since s is also known to the patient, the patient can also verify and retrieve his own remote medical records. On the other hand, s is unknown to the hospital, so the hospital cannot retrieve the patient's remote medical records although it can verify their integrity.

3.3. Correctness and security analysis

The security analysis of our FRR scheme is given by the following lemmas and theorems. Due to the page limits, the detailed process can be seen in the Appendix. Theorem 1 shows that our protocol can ensure the property of threshold fairness.

Theorem 1. *In the GroupBarter phase, our FRR scheme realizes multi-member threshold fair key exchange with STNP. The advantage of an adversary $\mathcal{A}$ (i.e., a dishonest member), is $\frac{1}{q}$ where q is the order of the field $\mathcal{Z}_q$, which is negligible.*

Proof. Given in Appendix 1. □

Lemma 1. *If the patient and CS are honest, i.e., they follow the proposed phases, then any block-tag pair can pass CS's tag checking, i.e., the CheckTag phase satisfies correctness.*

Proof. Given in Appendix 2. □

Lemma 2. *If the hospital and CS are honest, i.e., they follow the proposed phases, response V can pass the hospital's integrity checking, i.e., the Verify phase satisfies correctness.*

Proof. Given in Appendix 3. □

Lemma 3. *If the members and CS are honest, i.e., they follow the proposed phases, all members can retrieve the remote medical records, i.e., the Verify-Retrieve phase satisfies correctness.*

Proof. Given in Appendix 4. □

Lemmas 1–3 have shown the correctness results. We now turn to security results. Lemma 4 shows that a single *Tag* is existentially unforgeable.

Lemma 4. *Let $(\mathcal{G}_1, \mathcal{G}_2)$ be a (t', ϵ')-co-GDH group pair of order q. Then the tag generation scheme on $(\mathcal{G}_1, \mathcal{G}_2)$ is $(\hat{t}, q_S, q_h, \epsilon)$-secure against existential forgery under an adaptive chosen-message attack for all t and ϵ satisfying $\epsilon' \geqslant \frac{\epsilon}{\hat{e}(q_S+1)}$ and $t' \leqslant \hat{t} + c_{\mathcal{G}_1}(q_h + 2q_S) + c_e(q_H - 1)$, where $c_{\mathcal{G}_1}$ is a constant that depends on $\mathcal{G}_1$, $\hat{e}$ is the base of the natural logarithm, and q_S, q_h are the number of Tag queries and h-queries, respectively; $c_{\mathcal{G}_1}$ denotes the time cost of exponentiation on $\mathcal{G}_1$; c_e denotes the time cost of bilinear pairings.*

Proof. Given in Appendix 5. □

Lemma 5. *Given that a single tag is existentially unforgeable (proven in Lemma 4), a forged grouped tag-block pair $(T, \hat{m})$ can pass the verification with probability less than $\frac{1}{q}$, which is negligible.*

Proof. Given in Appendix 6. □

Lemma 6. *If some block-tag pairs $(\bar{m}_l, T_l),\ l \in \mathcal{M}_1 \subseteq \{i_1, i_2, \ldots, i_c\}$ are modified, CS can substitute the other valid block-tag pairs $(\bar{m}_{\hat{l}},\ T_{\hat{l}},\ \hat{l} \in \mathcal{M}_2$ for $(\bar{m}_l,\ T_l), l \in \mathcal{M}_1$ only with negligible probability.*

Proof. Given in Appendix 7. □

Theorem 2. *Let $(\mathcal{G}_1, \mathcal{G}_2)$ be a (t', ϵ')-co-GDH group pair of order q. Then a grouped message-tag is $(\hat{t}, q_S, q_h, \epsilon)$-secure against existential forgery under an adaptive chosen-message attack (in the ROM) for all $\hat{t}$ and ϵ satisfying $\min\left\{\hat{e}(q_S + 1)\epsilon', \frac{1}{q}\right\} \geqslant \epsilon$ and $t' \leqslant \hat{t} + c_{\mathcal{G}_1}(q_h + 2q_S) + c_e(q_H - 1)$, where $c_{\mathcal{G}_1}$ is a constant that depends on $\mathcal{G}_1, \hat{e}$ is the base of the natural logarithm, and $q_S,\ q_h$ are the number of Tag queries and h-queries, respectively; $c_{\mathcal{G}_1}$ denotes the time cost of exponentiation on $\mathcal{G}_1;\ c_e$ denotes the time cost of bilinear pairings.*

Proof. Given in Appendix 8. □

According to Lemmas 1–3, our FRR scheme satisfies the property of correctness. Putting these lemmas together with Theorems 1 and 2, we can assert that our FRR scheme is provably secure in the ROM.

4. Performance analysis

In this section, we first analyze the performance of our proposed FRR protocol in terms of computation and communication costs (Section 4.1). Then, in Section 4.2, we report on empirical figures obtained from an implementation of the solution.

4.1. Theoretical analysis

Computation. Suppose there exist l message blocks. In the *TagGen* phase, the patient needs 1 pairing, $2l$ exponentiations and l multiplications over $\mathcal{G}_1$. In the *CheckTag* phase, CS needs 1 pairing, $2l$ exponentiations and l multiplications. In the *GroupBarter* phase, every member needs $2(t-1)$ exponentiations and STNP needs t multiplications. In the *GenProof* phase, CS needs l exponentiations and $l-1$ multiplications. In the *Verify* phase, the hospital needs 3 pairings, $l+1$ exponentiations and l multiplications. In the *Verify-Retrieve* phase, the members need 3 pairings, $2l+1$ exponentiations and $2l-1$ multiplications. Other operations, such as hashing, exponentiation, and multiplication in the group $\mathcal{Z}_q^*$, are omitted since they just contribute a negligible computation cost compared with the cost of the multiplication and exponentiation operations in $\mathcal{G}_1$. This previously listed computational costs are totally tolerable with current computing devices.

Communication. The communication overhead mostly comes from the queries and responses. In the hospital query phase, if the number of queried blocks is l', the hospital needs to send $l'+2$ elements in $\mathcal{Z}_q^*$ to CS. In the response, CS needs to send 1 element in $\mathcal{G}_1$ and 1 element in $\mathcal{Z}_q^*$ to the hospital. In the member query phase, if the number of queried blocks is l, every member needs to send l elements in $\mathcal{Z}_q^*$ to CS. In the response, CS needs to send l elements in $\mathcal{Z}_q^*$ to the members. This communication overhead is totally tolerable with current communication technology.

4.2. Experimental analysis

In order to study the performance of our scheme, we have implemented a prototype. The tests were run on an ASUS S56C Laptop with the following settings:

- CPU: Intel Core i7-3517U @ 1.90 GHz.
- Physical Memory: 4 GB DDR3 1600 MHz.
- OS: Ubuntu 13.04 Linux 3.8.0-19-generic SMP i686.

Our code was written in Java7, using the java-7-openjdk-i386 environment. Java is slower than, for example, C when it comes to high-demand computations, but, considering that parts of this protocol might be run in mobile devices and that some of the most important cloud service providers run their systems in Java, the choice of this environment is reasonable.

We used the Java Pairing Based Cryptography Library (jPBC, [27]), which is a port of the PBC library [28]. Our choice of elliptic curve was Type A (see Chapter 8 of [29]), with $|r| = 160$ bits and $|q| = 512$ bits, making elements in $\mathcal{Z}_q^{\star}$ 160 bits long and elements in $\mathcal{G}_1$ 1024 bits long. Considering a scenario with l message blocks

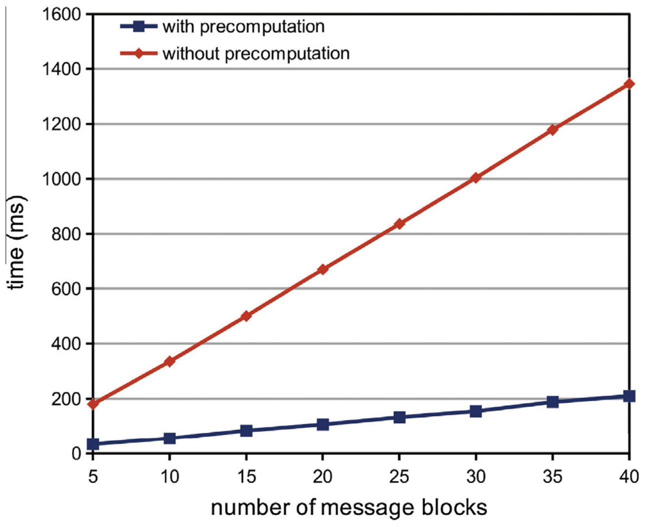

Fig. 3. Computation time for the patient.

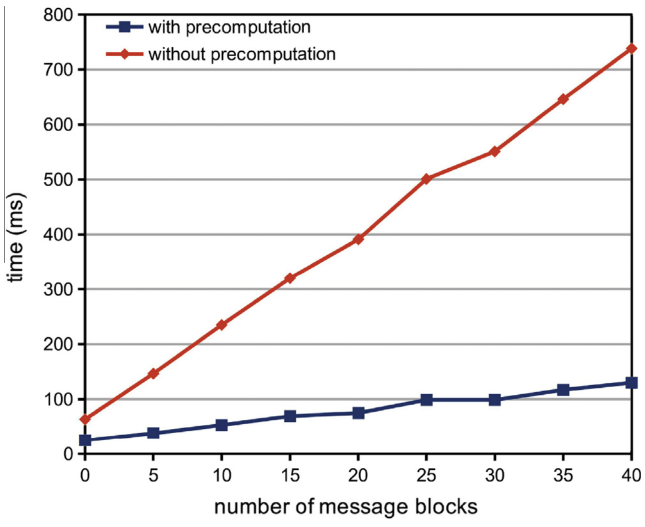

Fig. 4. Computation time for the hospital.

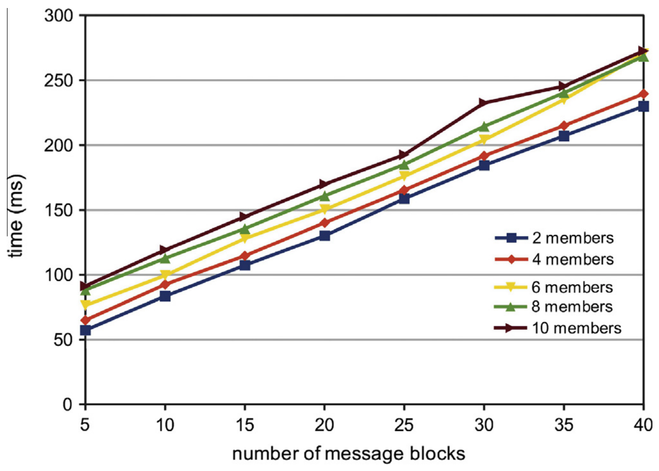

Fig. 5. Computation time for the members with precomputation.

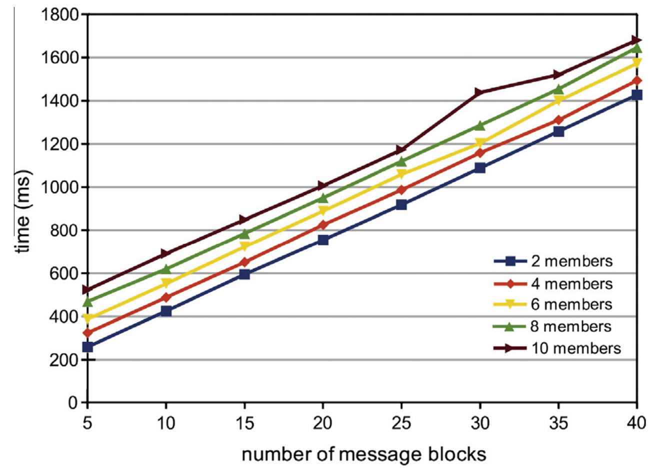

Fig. 6. Computation time for the members without precomputation.

and t members in the committee, we present the performance results for each of the actors of the protocol, namely the patients, the CS, the hospital and the STNP, for different values of l and t. Note

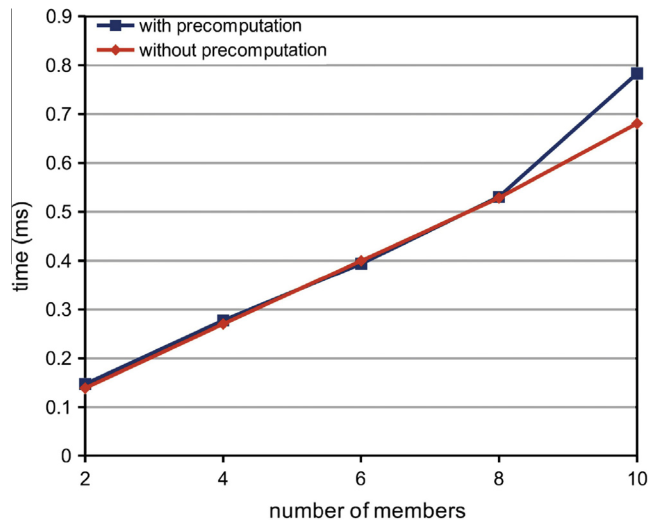

Fig. 7. Computation time for the STNP.

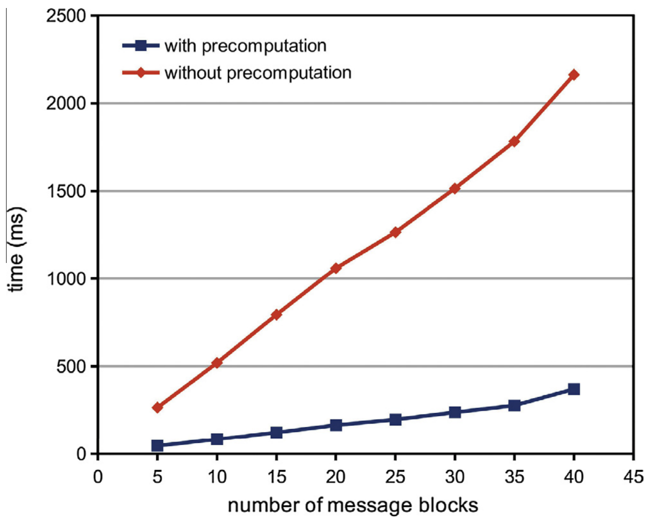

Fig. 8. Computation time for the cloud service.

that the jPBC library allows precomputation of operations, thereby trading storage capacity for speed in the computations. Based on two cases, with or without precomputation, Fig. 3 depicts the computation time for the patient and Fig. 4 depicts the computation time for the hospital. X-label denotes the file block number and Y-label denotes the computation time (ms) for the patient and the hospital. The computation times for the members with and without precomputation are depicted in Figs. 5 and 6, respectively. X-label denotes the file block number and Y-label denotes the computation time (ms) for committee members. After that, the computation times for the STNP and for the cloud service are depicted in Figs. 7 and 8, respectively. In Fig. 7, X-label denotes the number of committee members and Y-label denotes the computation time (ms) for STNP. In Fig. 8, X-label denotes the file block number and Y-label denotes the computation time (ms) for CS.

In order to show the feasibility of our proposed scheme in concrete terms, we consider a more realistic case for a hospital. We assume the one-day transactions of a hospital take 1 GB. These data will be stored in CS. Let the hash function h_1 be SHA-1, which maps 1 MB to 160 bits. Thus, the 1 GB data can be split into 1024 blocks of 1 MB each. According to the implementation, for 1024 blocks, every entity's time cost is given below: (1) The patient's time cost is 5025 ms = 5.025 s with precomputation and 34,163 ms = 34.163 s without precomputation. (2) The hospital's time cost is 2565 ms = 2.565 s with precomputation and 17034 ms = 17.034 s without precomputation. (3) When there are 8 members, every member's time cost is 5106 ms = 5.106 s with precomputation and 34176 ms = 34.176 s without precomputation. (4) CS's time cost is 7265 ms = 7.265 s with precomputation and 51,159 ms = 51.159 s without precomputation. The implementation results show that our proposed protocol is practical. Compared to the other remote data integrity checking schemes [30,31], our scheme has better efficiency. When the data are 1 GB, they are split into 1024 blocks in our scheme while they will be split into $8 \times 1024 \times 1024 \times 1024/160 = 51.2^2$ blocks in Refs. [30,31]. A larger block number results in lower efficiency. On the other hand, besides remote data integrity checking, our scheme has also another important security property, fair remote remote data retrieval, not offered by [30,31].

5. Conclusion

In this paper, we have proposed an efficient pairing-based FRR scheme which can check and retrieve remote data between untrusted members. We formally proved that the scheme is secure under the standard CDH assumption. Our performance analysis and implementation show that our scheme is efficient.

Acknowledgment and disclaimer

Thanks go to Alberto Blanco for his help with the simulations. This work was partly supported by the NSF of China through Projects (Nos. 61272522, 61173154, 61370190, 61003214), the Open Project of Shanghai Key Laboratory of Trustworthy Computing (No. 07dz22304201306), and by the Spanish Government through projects TIN2011-27076-C03-01 "CO-PRIVACY" and CONSOLIDER INGENIO 2010 CSD2007-00004 "ARES", by the Government of Catalonia under Grant 2009 SGR 1135, by the European Commission under FP7 projects "DwB" and "Inter-Trust". The last author is partially supported as an ICREA-Acadèmia researcher by the Catalan Government; he leads the UNESCO Chair in Data Privacy, but this paper does not necessarily reflect the position of UNESCO nor does it commit that organization.

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.jbi.2014.02.008.

References

[1] Eysenbach G, Powell J, Englesakis M, Rizo C, Stern A. Health related virtual communities and electronic support groups: systematic review of the effects of online peer to peer interactions. Br Med J 2004;328(7449):1166.
[2] Sun J, Fang Y, Zhu X. Privacy and emergency response in e-healthcare leveraging wireless body sensor networks. IEEE Wireless Commun 2010;17(1):51–8.
[3] Ateniese G, Burns R, Curtmola R, Herring J, Kissner L, Peterson Z, et al. Provable data possession at untrusted stores. In: CCS'07; 2007. p. 598–609.
[4] Ateniese G, DiPietro R, Mancini LV, Tsudik G. Scalable and efficient provable data possession. In: SecureComm 2008, art. no. 9; 2008.
[5] Erway CC, Kupcu A, Papamanthou C, Tamassia R. Dynamic provable data possession. In: CCS'09; 2009. p. 213–22.
[6] Sebé F, Domingo-Ferrer J, Martínez-Ballesté A, Deswarte Y, Quisquater J. Efficient remote data integrity checking in critical information infrastructures. IEEE Trans Knowledge Data Eng 2008;20(8):1–6.
[7] Wang H. Proxy provable data possession in public clouds. IEEE Trans Service Comput 2013;6(4):551–9.
[8] Wang H, Zhang Y. On the knowledge soundness of a cooperative provable data possession scheme in multicloud storage. IEEE Trans Parallel Distrib Syst 2013;25(1):264–7.
[9] Wang H. Identity-based distributed provable data possession in multi-cloud storage. IEEE Trans Service Comput 2014. http://dx.doi.org/10.1109/TSC.2014.1.
[10] Wang H, Wu Q, Qin B, Domingo-Ferrer J. Identity-based remote data possession checking in public clouds. IET Inf Secur 2013. http://dx.doi.org/10.1049/iet-ifs.2012.0271.
[11] Zhu Y, Wang H, Hu Z, Ahn GJ, Hu H, Yau SS. Efficient provable data possession for hybrid clouds. In: CCS'10; 2010. p. 756–8.
[12] Zhu Y, Hu H, Ahn GJ, Yu M. Cooperative provable data possession for integrity verification in multi-cloud storage. IEEE Trans Parallel Distrib Syst 2012;23(12):2231–44.
[13] Curtmola R, Khan O, Burns R, Ateniese G, MR-RDIC: multiple-replica provable data possession. In: ICDCS'08; 2008. p. 411–20.
[14] Yu Y, Niu L, Yang G, Mu Y, Susilo W. On the security of auditing mechanisms for secure cloud storage. Future Gen Comput Syst 2014;30:127–32.
[15] Fan X, Yang G, Mu Y, Yu Y. On indistinguishability in remote data integrity checking. Comput J 2013. [in press]. http://dx.doi.org/10.1093/comjnl/bxt137.
[16] Juels A, Kaliski Jr BS. PoRs: proofs of retrievability for large files. In: CCS'07; 2007. p. 584–97.
[17] Shacham H, Waters B. Compact proofs of retrievability. In: Asiacrypt; 2008. p. 90–107.
[18] Zheng Q, Xu S. Fair and dynamic proofs of retrievability. In: CODASPY'11; 2011. p. 237–48.
[19] Bowers KD, Juels A, Oprea A. Proofs of retrievability: theory and implementation. In: CCSW'09; 2009. p. 43–54.
[20] Dodis Y, Vadhan S, Wichs D. Proofs of retrievability via hardness amplification. In: TCC; 2009. p. 109–27.
[21] Zhu Y, Wang H, Hu Z, Ahn GJ, Hu H. Zero-knowledge proofs of retrievability. Sci China Inf Sci 2011;54(5):1608–17.
[22] Wan Z, Liu J, Deng R. HASBE: a hierarchical attribute-based solution for flexible and scalable access control in cloud computing. IEEE Trans Inf Forensics Secur 2012;7(2):743–54.
[23] Qin B, Wang H, Wu Q, Liu J, Domingo-Ferrer J. Simultaneous authentication and secrecy in identity-based data upload to cloud. Cluster Comput 2013;16(4):845–59.
[24] Boneh D, Franklin M. Identity-based encryption from the Weil pairing. In: Crypto; 2001. p. 213–29.
[25] Miyaji A, Nakabayashi M, Takano S. New explicit conditions of elliptic curve traces for FR-reduction. IEICE Trans Fundam 2001;5:1234–43.
[26] Boneh D, Lynn B, Shacham H. Short signatures from the Weil pairing. In: Asiacrypt; 2001. p. 514–32.
[27] De Caro A. jPBC – Java pairing-based cryptography library. Università degli Studi di Salerno. <http://gas.dia.unisa.it/projects/jpbc/index.html> [accessed 07.02.14].
[28] Lynn B. PBC library – pairing-based cryptography. Stanford University. <http://crypto.stanford.edu/pbc> [accessed 07.02.14].
[29] Lynn B. PBC library manual 0.5.11. Stanford University; 2006. <http://crypto.stanford.edu/pbc/manual.pdf>.
[30] Shen ST, Tzeng WG. Delegable provable data possession for remote data in the clouds. In: ICICS; 2011. p. 93–111.
[31] Purushothama BR, Amberker BB. Publicly auditable provable data possession scheme for outsourced data in the public cloud using polynomial interpolation. In: SNDS; 2012. p. 11–22.