SDSM: A Secure Data Service Mechanism in Mobile Cloud Computing

Weiwei Jia [†‡], Haojin Zhu[†], Zhenfu Cao[†§], Lifei Wei[†], Xiaodong Lin [♮]
[†]Shanghai Jiao Tong University, Shanghai 200240, P. R. China
jiaweiwei@hhu.edu.cn, zhu-hj@cs.sjtu.edu.cn, zfcao@cs.sjtu.edu.cn, weilifei@sjtu.edu.cn
[‡] Hohai University, Nanjing 210098, P. R. China
[♮] University of Ontario Institute of Technology, xiaodong.lin@uoit.ca
[§]corresponding author

***Abstract*—To enhance the security of mobile cloud users, a few proposals have been presented recently. However we argue that most of them are not suitable for mobile cloud where mobile users might join or leave the mobile networks arbitrarily. In this paper, we design a secure mobile user-based data service mechanism (SDSM) to provide confidentiality and fine-grained access control for data stored in the cloud. This mechanism enables the mobile users to enjoy a secure outsourced data services at a minimized security management overhead. The core idea of SDSM is that SDSM outsources not only the data but also the security management to the mobile cloud in a trust way. Our analysis shows that the proposed mechanism has many advantages over the existing traditional methods such as lower overhead and convenient update, which could better cater the requirements in mobile cloud computing scenarios.**

I. INTRODUCTION

The recent development of cloud computing has shown its potential to reshape the current way IT hardware designed and purchased. Among numerous benefits, cloud computing offers customers a more flexible way to obtain computation and storage resources on demand. Rather than owning (and maintaining) a large and expensive IT infrastructure, customers can now rent the necessary resources as soon as, and as long as, they need them. The benefits brought by Cloud Computing have been also demonstrated by the emergence of *Mobile Cloud Computing*, which is regarded as one of most disruptive technology for future mobile applications. Different from the general cloud computing concept, mobile cloud computing refers to an emerging infrastructure where both of the data storage and the data processing happen outside of the mobile device from which an application is launched. According to a latest study from Juniper Research, the market for cloud-based mobile applications will grow 88% from 400 million in 2009 to 9.5 billion in 2014 [1]. A similar forecast has also been made by ABI, which predicts that the number of mobile cloud computing subscribers worldwide is expected to grow rapidly over the next five years, rising from 42.8 million subscribers in 2008, (approximately 1.1% of all mobile subscribers) to just over 998 million in 2014 (nearly 19%). ABI further forecasts that mobile cloud will soon become a disruptive force in the mobile world, eventually becoming the dominant way in which mobile applications operate [2].

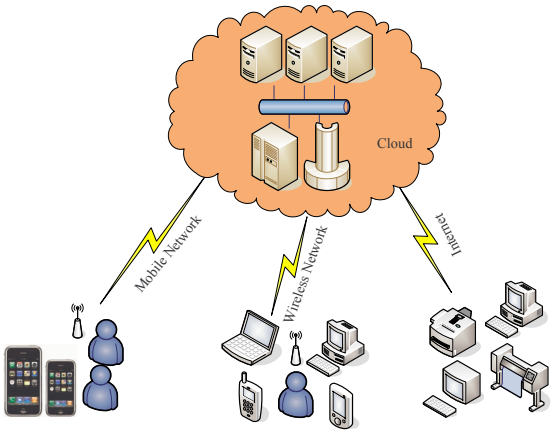

Fig. 1. Architecture of mobile cloud computing

Even though mobile cloud computing is envisioned as a promising service architecture for the future mobile applications, security and privacy is one of major challenges which may prevent its wide acceptance in practice. Specifically, to store data on the cloud, data owners no longer run and keep their data in their personal mobile devices. Instead, the data will be maintained by the "cloud". From users' point of view, putting sensitive data on the cloud and losing control of these data may increase the risk of being abused. These concerns originate from the fact that cloud servers are usually operated by commercial providers which may normally be out of users' trust domain. To provide privacy and enhance security for cloud users, there are quite a few security proposals recently, such as [3]–[7]. Specifically, [3]–[6] mainly focus on data auditing issue in cloud while [7] introduces a protocol to realize the confidentiality and access control of data based on the attribute based encryption (ABE). However, due to the dynamic network topology of mobile networks the existing security proposals presented which could achieve both of the confidentiality and data access control may not be suitable in mobile cloud computing . Specifically, the attribute based encryption or multi-recipient encryption which

978-1-4577-0248-8/11/$26.00 ©2011 IEEE

is more suitable for a static and small-scale network rather than for the mobile network, which is typically dynamic and potentially comprised of millions of mobile users who could join or leave the network arbitrarily. Moreover one user may possess many attribute and conversely one attributes may be possessed by many users which makes the data owner difficult to set up the correspondence between the users and attributes. These observations motivate us to propose a novel data service mechanism in mobile cloud computing.

In this work, we propose SDSM, a user-efficient and secure data service mechanism in mobile cloud computing, which enables the mobile users to enjoy a secure outsourced data services at a minimized security management overhead. The core idea of SDSM is that SDSM outsources not only the data but also the security management to the mobile cloud in a trusted way. To achieve this, we adopt an identity-based proxy re-encryption scheme which allows a mobile user to encrypt his data under his identity to protect his data from leaking and, at the same time, to delegate his data management capability to the mobile cloud. Furthermore the mobile user could delegate his access control capability to the cloud, which could grant the access of an authorized user by transforming the ciphertext encrypted with the data owner's identity to the one with the sharer's identity.

We summarize the contributions of this paper as follows:

- In this paper, we analyze the advantages and weaknesses of possible approaches to solve the confidentiality and access control issue in the mobile cloud computing.
- We propose a novel secure data service mechanism, SDSM, to efficiently achieve both of the secrecy and access control of data. Specifically, the mobile user can securely shift the data computing and distribution overhead to the cloud while the cloud has no idea about data content in the whole process. Additionally only authorized users can decrypt the ciphertext while unauthorized users would learn nothing about the data.
- we give the security and performance analysis on SDSM. In SDSM, the overhead of communication for the data sharing is reduced to the size of a re-encryption key. This could greatly reduce the cost of the user side which is charged based on the size of communications.

The remainder of this paper is organized as follows. Section II presents the related works. Section III describes the system model and some assumptions in SDSM which include the introduction of network model, system goals, and some preliminaries and notations. Section IV gives an overview of our secure data management mechanism. Section V shows the detailed protocol to achieve both the secrecy and the access control of data stored in the mobile cloud. Section VI presents some analysis on performance and security of our protocol. Finally, we conclude our paper in Section VII.

II. Related Works

So far, various proposals have been proposed to achieve secure remote storage [3]–[9]. In [8], S.Creece et al. discuss some privacy concerns in the design of cloud storage. As for the auditing of integrity of data, [3]–[6] present some efficient protocols to convince the users that their data is intact and retrievable in the cloud. Most of these approaches can be classified into two classes: MAC based schemes [3] and public signature based auditing schemes [4]–[6]. As for the access control of data, [10] and [11] introduce various approaches to achieve the data access control. However, they do not take the possible attacks launched by untrusted or semi-trusted cloud providers. In [7], it is further pointed out that, due to the semi-trusted of cloud servers, we should protect the data from being leaked to servers. In [7], S. Yu, et al. exploit a novel cryptographic approach, key policy-attribute based encryption scheme (KP-ABE), to achieve the secure data outsourcing storage and access control in the semi-untrusted cloud servers, and also apply re-encryption scheme in revocation phase to reduce the cost of data owner. However, we argue that in a dynamic mobile cloud, the ABE based approach may not be efficient to provide user access control due to frequent node revocations. The drawbacks of exploring the ABE as method to achieve access control are shown as follows:

1) the mobile user should know the list of sharers' attributes before encryption;
2) Sharers satisfying access policy may consist of a few persons, which makes the data owner difficult to implement the fine-grained access control of data.

To clearly depict a set of designated sharers may include many attributes. The increased of attributes will inevitably brings the size growth of the ciphertext in the ABE scheme. Even the size of ciphertext is constant, the system parameters will be also increased greatly. Therefore to achieve ABE-based access control solution may not be suitable for the mobile cloud.

Another potential approach to achieve the designated access of encrypted data is based on multi-recipient encryption [12], [13] (MRE), a public key cryptography primitive for one-to-many communications. Exploring the MRE, the size of ciphertext is less than the trivial n-recipient solutions which is just a concatenation of independently encrypted messages for n recipients using a single-recipient public key encryption algorithm. However predefining a set of sharers in advance makes it difficult to implement in a scalable mobile cloud. When changing the access policy of sharing, data owner has to retrieve the data from the cloud, decrypt ciphertext, re-encrypt the data corresponding to distinct user. The expensive cost on computation and communication in the updating of access policy makes it impractical in the mobile cloud.

III. System Model and Assumptions

A. Network Model

As shown in Figure 2, the network model is assumed to comprise of the following parties: data owners, cloud servers, and data sharers. Both of the data owners and data sharers have mobile device to easily access the Internet. To protect data from leaking to the third party, data owner forwards the encrypted files on the cloud servers to either for sharing or

for personal use. The data sharers, who want to access data files, will be authorized by the data owner to decrypt the file. Cloud servers are assumed to have abundant storage capacity and computation power and are assumed to be always online.

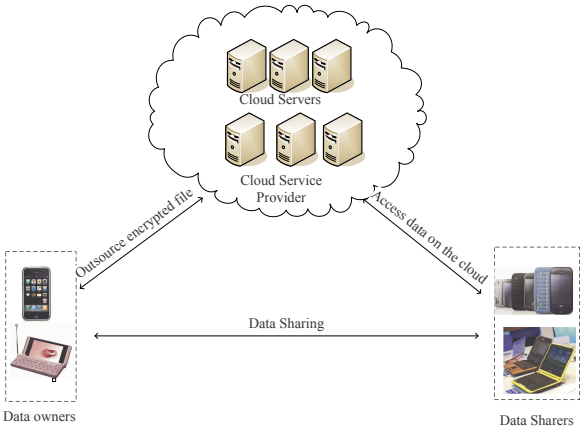

Fig. 2. Network model in our protocol

B. Security Model

In this work, we mainly consider the threats from the semi-trusted cloud server in the data storing and malicious sharer in the data sharing. Similar to [14], we consider the cloud servers to be semi-trusted. That is to say, cloud servers will honestly implement the proposed protocol in general, but try to find out as much secret information as possible based on users' inputs. Malicious sharers may try to access the data without permission by data owner. The goal of our protocol is to guarantee that only authorized sharer can access the data and conversely unauthorized sharer will learn nothing. Moreover the collusion attack of the malicious sharers and the semi-trusted cloud servers should also be considered. In some cases, the collusion attack launched by the sharers and the cloud servers even corrupts the data owner's secret key, which should be strictly resisted in our mechanism. We will give a more detailed description about the ability of the adversaries as follows.

Following the definition in cryptography, we assume the ability of the adversary $\mathcal{A}$ in our protocol could be adaptive. It means that the adversary $\mathcal{A}$ can gain some additional information to help them to compromise the system. In our protocol, each user entering the system has a unique identity and a secret key corresponding to his identity. The adversary $\mathcal{A}$ can corrupt some users' secret keys and manipulate the server to achieve some proxy re-encryption keys before he launches the attack. After the adversary $\mathcal{A}$ gets sufficient additional information, he will launch an attack. In the attack phase, the adversary $\mathcal{A}$ chooses one user id^* as his target goal. We will give some restrictions on choosing an attack target id^* as follows.

- The secret key of the attack target user id^* cannot be compromised by the attacker.
- Adversary $\mathcal{A}$ can not hold the re-encryption key which could be used to convert any ciphertext to the target user id^* for decryption. The reason is that if the adversary has the re-encryption keys of id^*, then he can directly transform any ciphertext into the one that id^* can decrypt.

The attack is successful if the adversary can differentiate two different ciphertexts under id^*'s identity.

C. System Goals

Our goals addressed in this paper are to achieve a secure data service mechanism of mobile users. We emphasize on the problem of confidentiality and access control of mobile network users' outsourced data in the cloud circumstances. A solution to this problem should have the following requirements:

- Protect the mobile user's data from leaking to the cloud.
- Guarantee the authorized sharers can access the data, while unauthorized sharers can not learn anything about data.
- Provide mobile users easy operations on setting and changing the access policies.
- Reduce the communication cost of mobile user.

D. Preliminaries and Notations

1) Bilinear pairing: Let $\mathbb{G}_1$ and $\mathbb{G}_T$ be a cyclic multiplicative group with the same prime order q, that is, $|\mathbb{G}_1| = |\mathbb{G}_2| = q$. Let g be a generator of $\mathbb{G}_1$. An efficient bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$, with the following properties:

- Bilinear: for all $g \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q^*$, $e(g^a, g^b) = e(g, g)^{ab}$.
- Non-degenerate: there exists $g \in \mathbb{G}_1$ such that $e(g, g) \neq 1$.
- Computable: there is an efficient algorithm to compute $e(g_1, g_2)$ for any $g_1, g_2 \in \mathbb{G}_1$.

Typically, we can implement the bilinear map using Weil or Tate pairing [15].

2) Notations: In SDSM, let $(\mathbb{G}_1, \mathbb{G}_T)$ be a pair of bilinear groups with the same prime order q. Let e be a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ which has the properties of *bilinearity*, *computability*, and *non-degeneracy*. In addition, we make use of two independent hash functions $\mathcal{H}_1$ and $\mathcal{H}_2$

- $\mathcal{H}_1 : \{0,1\}^* \to \mathbb{G}_1$
- $\mathcal{H}_2 : \mathbb{G}_T \to \mathbb{G}_1$

3) Problem assumptions: **Decisional Bilinear Diffe Hellman Assumptions (DBDH)**
Let $(\mathbb{G}_1, \mathbb{G}_T)$ be a pair of bilinear groups with an efficiently computable pairing. Given a randomly chosen $g \in \mathbb{G}_1$, as well as g^a, g^b and g^c (for unknown randomly chosen $a, b, c \in Z_q^*$). The DBDH problem is to decide, given a tuple of values $(g, g^a, g^b, g^c, T) \in \mathbb{G}_1^4 \times \mathbb{G}_T$, whether $T = e(g, g)^{abc}$ or if T is a random element of $\mathbb{G}_T$.

IV. OVERVIEW OF SDSM

We present SDSM to achieve the secure data management including the secrecy and data access control in the mobile

cloud computing. In our protocol, we will employ identity-based proxy re-encryption to realize the secrecy of data. The proxy re-encryption scheme (PRE) [16], [17] is to allow a semi-trusted proxy to transform the data encrypted with Alice's public-key into the one encrypted with Bob's public key. The most natural application of PRE is in the transmission of Email to allow Bob to read Alice's encrypted emails while she is on vacation. In the mobile cloud computing, we employ the PRE to realize the access control of data and offers the following benefits:

1) *Strong access control*: Only authorized user can decrypt the data. Data owner can distinguish the identity information of sharers.
2) *Flexibility*: Our protocol is flexible to operate and scalable with the growth of data sharers. Data owner only need to forward a re-encryption key to cloud which can complete the transformation of ciphertext. No requiring classify the sharers in attributes by data owner makes our protocol easy to operation. Moreover, due to the lower cost of communication, our protocol is fit for the scalability of network.
3) *Low overhead*: The cost of achieve, change and update access policy is relatively lower.

In the mobile network, each user needs to register with an identity in the system and thus obtains the secret key corresponding to his identity. Mobile users explore IB-PRE encryption algorithm to encrypt the data which guarantees the data secrecy of the cloud service, and the ciphertext can be transformed to the one that could be decrypted by others in the future. Before encryption, the file F is divided into n blocks $F = (m_1, m_2, \cdots, m_n)$. For each block m_i data owner encrypts it under his secret key. At the same time, data owner should generate the proxy re-encryption key to the authorized user. And different sharer's identity corresponding to different proxy re-encryption key is generated. The size of re-encryption key is almost similar to the size of a block m_i in file F. Given the proxy re-encryption key by the owner, the cloud can convert the ciphertext outsourced by the data owner to the ciphertext that can be decrypted by the sharer. As mentioned above, in our proposal the role of the cloud is twofold: a) providing secure storage for the mobile users, and b) serving as the secure proxy. From the perspective of the mobile user, the task of convert ciphertext is relinquished to the cloud, and the mobile user just only needs to upload a key whose size is far less than the whole file.

The general PRE framwork consists of six algorithms (Setup, KeyGen, Encrypt, Decrypt, RKGen, Re-encrypt) [16]:

-Setup(1^λ) on input a security parameter λ, the algorithm outputs both the system parameters (params) which are distributed to users, and the master secret key (msk) which is kept private.

-KeyGen(params, msk, ID) on input an identity $ID \in \{0,1\}^*$, $params$ and the master secret key, this algorithm outputs a decryption key sk_{id} corresponding to users identity.

-Encrypt(params, ID, m) on input a set of public parameters, an identity $ID \in \{0,1\}^*$ and a plaintext $m \in M$, this algorithm output c_{ID}, the encryption of m under the specified identity ID.

-RKGen(params, sk_{ID_1}, ID_1, ID_2) on input a secret key sk_{ID_1} (derived via the KeyGen algorithm) and identities $(ID_1, ID_2) \in \{0,1\}^*$, this algorithm outputs a *re-encryption key* $rk_{ID_1 \to ID_2}$.

-Re-encrypt(params, $rk_{ID_1 \to ID_2}, c_{ID_1}$) on input a ciphertext c_{ID_1} under identity ID_1, and a re-encryption key $rk_{ID_1 \to ID_2}$, outputs a re-encrypted ciphertext c_{ID_2}.

-Decrypt(params sk_{ID}, c_{ID}) decrypts the ciphertext c_{ID} using the secret key sk_{ID} and outputs m or $\perp$.

V. OUR PROTOCOL

We first introduce the concrete algorithm of SDSM. Figure 3 shows the detail of the inputs and outputs of six functions in our protocol. We assume the cloud servers and the mobile users are in the same system domain and sharing the uniform system parameters. Mobile users will encrypt their data first and then forward the ciphertext to the cloud servers. At the same time the mobile user delegates his access control capability to the cloud. The mobile cloud stores the encrypted data, and is delegated to transform the ciphertext encrypted with the data owner's identity to the one with the requester's identity. We will use five phases to depict our protocol as follows.

A. Setup

In this phase, by using the `setup` and `KeyGen` algorithms the system parameters and users secret key are built up. We set the system master secret key $msk = s$. Each mobile user registered in the system can obtain a private key corresponding to his identity, $sk = \mathcal{H}_1(\mathrm{ID})^s$. Note that the master key s is only used in the process of user registration. The data owner can share his data only given the identity of the sharers.

B. Data encryption

The data F is divided in to n fractions $F = (m_1, m_2, \cdots, m_n)$. For m_i data owner runs the `Encrypt` algorithm and generates $M_i = (g^r, m \cdot e(g^s, \mathcal{H}_1(ID)^r))$. After implementing the encryption of F, the mobile user uploads $F^{'} = (M_1, M_2, \cdots, M_n)$ to the cloud.

C. Data sharing

In this phase, data owner runs the `RKGen` algorithm and generates the proxy key $rk = (\mathcal{H}_1(\mathrm{ID}_A)^{-s}, IBE_{\mathrm{ID}_B}(X))$ to cloud, where X is randomly selected from G_T. The re-encryption key rk will be used by the cloud to transform the ciphertext $F^{'}$ to the ciphertext under sharer's public key. The data owner forwards rk to the cloud which means that the cloud is delegated to manage the data in behalf of the owner. The cloud can deploy the re-encrypt key rk to permit the authorized user to get the ciphertext decrypted with his own secret key.

D. Access data

When the sharer wants to access the file, he sends a request to the cloud server. The cloud determines the validity of the sharer by checking if it has a re-encryption key to the sharer. With the re-encryption key is existed, the cloud server can run the `RKGen` algorithm and achieve the re-encryption ciphertext $C = (c_1, c_2, c_3) = (g^r, m \cdot e(g^r, \mathcal{H}_2(X)), IBE_{Bob}(X))$. Then the sharer fetches the re-encrypted data from the cloud servers, and runs the `Decrypt` algorithm on M_i with his secret key to obtain the $m_i = \frac{c_2}{e(c_1, \mathcal{H}_2(x))}$. As doing so, the sharer gets the entire file $F = (m_1, m_2 \cdots, m_n)$.

As previously defined, let $(\mathbb{G}_1, \mathbb{G}_T)$ be a pair of bilinear groups with prime order q. Let e be a bilinear map, and let $\mathcal{H}_1$ and $\mathcal{H}_2$ be two independent hash functions.

$Setup(1^\lambda)$:
Given the security parameter 1^λ, this algorithm outputs the system params=$(\mathbb{G}_1, \mathbb{G}_T, g, g^s)$ ($s \xleftarrow{R} Z_q^*$) and the params master secret key msk= s.

$KeyGen(\mathrm{ID}_A, params, msk)$:
Given the user's identity ID_A, this algorithm computes the secret key corresponding to his identity ID_A sk: $sk = \mathcal{H}_1(\mathrm{ID}_A)^s$.

$Encrypt(params, \mathrm{ID}_A, m)$:
To encrypt the message m under users identity ID_A as the public key, this algorithm does the following: choose a random $r \in Z_q^*$, and calculated the ciphertext as $IBE_{\mathrm{ID}_A}(m) = (g^r, m \cdot e(g^s, \mathcal{H}_1(\mathrm{ID}_A)^r))$.

$RKGen(params, sk_{\mathrm{ID}_A}, \mathrm{ID}_A, \mathrm{ID}_B)$:
Given the identity $\mathrm{ID}_A, \mathrm{ID}_B$ and a random chosen $X \xleftarrow{R} \mathbb{G}_T$, this algorithm generates the re-encryption key as follows: $rk_{\mathrm{ID}_A \to \mathrm{ID}_B} = (\mathcal{H}_1(\mathrm{ID}_A)^{-s}, IBE_{\mathrm{ID}_B}(X))$.

$Reencrypt(params, rk_{\mathrm{ID}_A \to \mathrm{ID}_B}, C_{\mathrm{ID}_A})$:
Given an ciphertext for ID_A, and a re-encryption key from ID_A to ID_B, this algorithm outputs the ciphertext: $C_{\mathrm{ID}_B} = (c_1, c_2, c_3)$, and $c_1 = g^r; c_2 = m \cdot e(g^r, \mathcal{H}_2(X)); c_3 = IBE_{Bob}(X)$.

$Decrypt(params, sk_{ID_B}, C_{ID_B})$:
Given the ciphertext C_{ID_B}, the secret key of ID_B, this algorithm decrypts the ciphertext (c_1, c_2, c_3) and obtains m by computing $\frac{c_2}{e(c_1, \mathcal{H}_2(x))}$.

Fig. 3. proxy re-encryption scheme

E. Policy updating

In practice the mobile user may want to update the list of sharers with creating or revoking some sharers in a large and dynamic mobile network. Using our approach the mobile user might do this task even without retrieving and decrypting the ciphertext from cloud. To implement the updating or deleting sharers means to create new re-encryption keys, or delete old re-encryption keys in the system. For example, Alice could add a new sharer Bob and repeal the privilege of Charlie by doing the following procedures:

- Notify the proxy to delete the re-encryption key from Alice to Charlie. This would prevent the cloud to convert the encrypted data under Alice to the one under Bob.
- Create new re-encryption key $rk_{Alice \to Bob}$ and send it to the cloud. This would allow the cloud to convert the encrypted data under Alice to the one under Charlie.

VI. Security and Performance Analysis

A. Security analysis

Correctness: In SDSM, the data owner and authorized users can correctly decrypt the ciphertext.

The correctness of decryption for data owner is shown as follows.

$$(m \cdot e(g^s, \mathcal{H}_1(\mathrm{ID}_A))^r) / e(g^r, \mathcal{H}_1(\mathrm{ID}_A)^s) = m$$

The correctness of the protocol on the sharers will be shown as follows. Given the ciphertext $C_{\mathrm{ID}_B} = (g^r, m \cdot e(g^r, \mathcal{H}_2(X)), IBE_{\mathrm{ID}_B}(X))$, the sharer ID_B can recover X from $IBE_{\mathrm{ID}_B}(X)$ and then perform the follows

$$m = \frac{c_2}{(e(g^r, \mathcal{H}_2(X))}$$

The soundness of our protocol is given in the following theorem. Due to the space constraints, we will not display the formal proof which can be found in [16].

Theorem Our protocol is secure under adaptive adversary model as above defined based on the intractability of DBDH problem [16].

B. Performance analysis

We measure the performance of our protocol in Windows XP operation system. All experiments are conducted on the the Intel Pentium(R) Dual-Core E6300 with 2 GB RAM. During the operation of data, it involves the multiplication, exponentiation, and the pairing in a cyclic group. Based on cryptographic library MIRACL [18], we estimate the time consumption of these operations in Table I. To demonstrate our advantage in computation and communication overhead. For comparison, we also analyze the cost of trivial solution which makes the user to endure the heavy cost in distribution of data. In other words, the trivial solution allows the data owner to compute and forward the different ciphertext to different sharer by his own.

Due to the unlimited computing resources of cloud, in our performance analysis we will mainly consider the overhead on the side of mobile users including the cost in the encryption data and generating the re-encryption key.

In the process of encrypted data, we will estimate the overhead of processing each file block m_i with the encryption algorithm. The running time is denoted by $Operate(m_i)$. For

	Descriptions	Execution Time
T_{mul}	one multiplication in cyclic group	3.6 ms
T_{pair}	one pairing operation on Ellipse Curve	15 ms
T_{exp}	1024-bit modular exponentiation	8.59ms

TABLE I
EXECUTION TIME OF MULTIPLICATION, EXPONENTIATION AND PAIRING.

the entire file, the whole operation time will be determined by $n \cdot Operate(m_i)$. The number of file blocks n will be determined by the size of file. We consider all operation completed in the group Z_q (mostly $|q| = 1024$ bits). g^s and $H_1(Alice)$ can be assumed as constant values in the encryption procession. Thus the computation in the encryption of m_i includes 2 exponentiations, 1 multiplication and 1 pairing. So the time consumption in the data encryption is close to $n \cdot 15ms$.

The cost of generating a re-encryption key is similar to the encryption of m_i plus one exponentiation. The second element of the re-encryption key is generated by employing the identity based encryption scheme to encrypt a random selected element X. Thus the delay in generation of proxy re-encryption keys is approximately to $|k_{num}| \cdot 44ms$, where $|k_{num}|$ denotes the number of the data sharers. Since each file only needs to be pre-processed one time, the delay could be further reduced in the real practice.

We then discuss the communication overhead of the data owner to realize the data sharing with the authorized users. Table II shows the comparison of communication overhead between our protocol and the trivial solution. Our protocol shows a comparable advantage to achieve data sharing with lower overhead. In our protocol, to achieve the access control of file F, data owner only requires forwarding a re-encryption key of a small size $2q$ ($q \ll |F|$) instead of the ciphertext of F.

	Communication overheads
Trivial approach	$O(\|F\|)$
Our protocol	$O(q)$

TABLE II
COMPARISON OF COMMUNICATION OVERHEAD.

VII. CONCLUSION

In this paper, we presented a secure data mechanism to solve the problem of data secrecy and privacy in mobile cloud computing. We first summarize the possible approaches to realize the access control in cloud computing, and show that the situation when mobile users may arbitrarily join or leave the mobile network makes these approaches not suitable to be used in mobile cloud computing. Afterwards, in this paper we explored identity based proxy re-encryption scheme to make mobile users easily implement fine-grained access control of data and also guarantee the data privacy in the cloud. At the same time, the cost of updating of access policy and communication is also reduced in our mechanism.

ACKNOWLEGEMENT

This research is supported by National Natural Science Foundation of China (Grant No. 61003218 and No. 61033014), Doctoral Fund of Ministry of Education of China (Grant No.20100073120065), Natural Science Foundation of Hohai University (Grant NO. 2009427811), and Science and Technology Commission of Shanghai (Grant No.10511501503).

REFERENCES

[1] Juniper, "Mobile cloud computing: $9.5 billion by 2014," Juniper, http://www.readwriteweb.com/archives, Tech. Rep., 2010.

[2] ABI, "Mobile cloud computing subscribers to total nearly one billion by 2014," ABI, http://www.abiresearch.com/press/1484/, Tech. Rep., 2009.

[3] K. D. Bowers, A. Juels, and A. Oprea, "Hail: a high-availability and integrity layer for cloud storage," in *ACM Conference on Computer and Communications Security*, 2009, pp. 187–198.

[4] R. Curtmola, O. Khan, R. C. Burns, and G. Ateniese, "Mr-pdp: Multiple-replica provable data possession," in *ICDCS*, 2008, pp. 411–420.

[5] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *INFOCOM*, 2010, pp. 525–533.

[6] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in *ESORICS*, 2009, pp. 355–370.

[7] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *INFOCOM*, 2010, pp. 534–542.

[8] S. Creese, P. Hopkins, S. Pearson, and Y. Shen, "Data protection-aware design for cloud services," in *CloudCom*, 2009, pp. 119–130.

[9] L. Wei, H. Zhu, Z. Cao, W. Jia, and A. Vasilakos, "SecCloud: Bridging Secure Storage and Computation in Cloud," in *Distributed Computing Systems Workshops (ICDCSW), 2010 IEEE 30th International Conference*, 2010.

[10] R. Sandhu, E. Coyne, H. Feinstein, and C. Youman, "Role-based access control models," *Computer*, vol. 29, no. 2, pp. 38–47, 2002.

[11] R. Sandhu and P. Samarati, "Access control: principle and practice," *Communications Magazine, IEEE*, vol. 32, no. 9, pp. 40–48, 2002.

[12] K. Kurosawa, "Multi-recipient public-key encryption with shortened ciphertext," in *Public Key Cryptography*, 2002, pp. 48–63.

[13] M. Bellare, A. Boldyreva, and J. Staddon, "Randomness re-use in multi-recipient encryption schemeas," in *Public Key Cryptography*, 2003, pp. 85–99.

[14] S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Over-encryption: Management of access control evolution on outsourced data," in *VLDB*, 2007, pp. 123–134.

[15] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," *SIAM Journal on Computing*, vol. 32, no. 3, pp. 586–615, 2003.

[16] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in *ACNS*, 2007, pp. 288–306.

[17] J. Shao and Z. Cao, "Cca-secure proxy re-encryption without pairings," in *Public Key Cryptography*, 2009, pp. 357–376.

[18] M. Scott, "Implement cryptographic pairings," in *The first International Conference on Pairing-based Cryptography (Pairing'07)*, Tokyo, Japan, July 2-4, 2007.