I.J. Wireless and Microwave Technologies **2012, 1, 53-59**
Published Online February 2012 in MECS (http://www.mecs-press.net)
DOI: 10.5815/ijwmt.2012.01.08

Available online at http://www.mecs-press.net/ijwmt

Efficient Proxy Re-encryption with Private Searching in the Untrusted Cloud

[a 1]Xi Chen, [a 2]Yong Li

[a] *Key Laboratory of Communication & Information Systems (Beijing Jiaotong University), Beijing Municipal Commission of Education, Beijing 100044, China*

Abstract

As promising as cloud computing is, this paradigm brings forth new security and privacy challenges when operating in the untrusted cloud scenarios. In this paper, we propose a new cryptographic primitive Proxy Re-encryption with Private Searching (PRPS for short). The PRPS scheme enables the data users and owners efficiently query and access files storaged in untrusted cloud, while keeping query privacy and data privacy from the cloud providers. The concrete construction is based on proxy re-encryption, public key encryption with keyword search and the dual receiver cryptosystem. The scheme is semantically secure under the BDH assumption.

Index Terms: public key encryption with keyword search; proxy re-encryption; untrusted cloud; private searching

1. Introduction

Cloud computing is an important trend which is beginning to fulfill the early promise of the Internet and creating unanticipated change in computing paradigm. However, a significant barrier to the adoption of cloud computing is that data owners fear of confidential data leakage and lose of privacy in the cloud [1]. These concerns originate from the fact that cloud providers are usually operated by commercial providers which are very likely to be outside of the trusted domain of the data owners or users. Data confidentiality against cloud providers is hence frequently desired when data owners outsource data for storage in the cloud [2].

Our work is motivated by the following scenario. Data owners, cloud storage providers and data users are separated geographically. Data owner stores his files in an encrypted form in the untrusted cloud, and retrieves them wherever and whenever he wants. The user sends a query for files containning certain keywords to the cloud provider. The desired requirements are: 1) The user can decrypt the files uploaded by the data owner with his private key; 2) The cloud provider can search whether the encrypted files contain some keywords; 3) The

Corresponding author:
E-mail address: [1]09120113@bjtu.edu.cn, [2] li.yong9@gmail.com

cloud provider oughts to keep blind to the files content and the query keywords of the user; 4) The user could finish query and decryption with a thin client which demands computing overhead as small as possible.

1.1. Related work

Proxy Re-Encryption (PRE). PRE is a cryptographic primitive, where a (potentially untrusted) proxy is given a re-encryption key $rk_{1\to 2}$ that allows it to translate a message m encrypted under public key pk_1 into a ciphertexts under a public key pk_2 , without being able to see anything about the encrypted messages. In [3], Ateniese *et al.* proposed a single-use, unidirectional, but not transparent Proxy Re-Encryption schemes based on bilinear maps.

Public key encryption with keyword search (PEKS). In PEKS scheme, Alice creats a trapdoor with her private key and a keyword, and sends it to S. S uses a test algorithm with inputing encrypted keyword, trapdoor and user's public key . If matches, it outputs 1 and 0 otherwise. PEKS supports that a user could search for some files containning certain keywords in untrusted storage servers, and at the same time, the servers keep blind to the privacy of file and the keyword. In [4], Boneh *et al.* proposed a public key encryption with keyword search scheme.

Dual receiver cryptosystem. Diament *et al.* [5] first introduced the notion of an effcient dual receiver cryptosystem, which enables a ciphertext to be decrypted by two independent receivers. The main disadvantage of the dual receiver cryptosystem is that the server needs to send an auxiliary private key to a client for decrypting a partial ciphertext, which is insecure in the real environment [6].

Liu *et al*. [6] improved the PEKS by inspiring the idea of dual receiver cryptosystem, and proposed an effcient privacy preserving keyword search scheme. However, this scheme is one specific case applicable in the setting that the data owner and data user is the *same one*. Shao *et al.* [7] introduced the concept of proxy re-encryption with keyword search (PRES), in particular the concept of bidirectional PRES, against the chosen ciphertext attack. Note that the third party is trusted, and this scheme improved the security level with the sacrifice of efficiency.

1.2. Our contributions

We proposed a new cryptographic primitive, Proxy Re-encryption with Private Searching (PRPS), and the new PRPS construction combines technologies from PRE, PEKS and dual receiver cryptosystem. The PRPS scheme is able to protect the data privacy and the users' queries privacy simultaneously during the search process. And it is provably secure under the BDH assumption in random oracle model. In addition, the PRPS scheme enables the decrease of computing overhead for the user and reduces the modification of encrypted storage when different users accessing the cloud provider.

2. Preliminaries

Let G_1 and G_2 be two cyclic groups of some large prime order q . We view G_1 as an additive group and G_2 as a multiplicative group.

Definition 2.1 (Bilinear Maps): A bilinear map $e: G_1 \times G_1 \to G_2$ is a map with the following properties: (1) Bilinearity: for any integers $x, y \in [1, q]$, we have $e(g^x, g^y) = e(g,g)^{xy}$. (2) Computability: given $g, h \in G_1$, there is a polynomial time algorithms to compute $e(g,h) \in G_2$. (3) Non-degeneracy: if g is a generator of G_1, then $e(g,g)$ is a generator of G_2 .

Definition 2.2 (BDH Problem): Given a random element $g \in G_1$, as well as g^x, g^y and g^z, for some $x, y, z \in Z_q^*$, compute $e(g,g)^{xyz} \in G_2$.

3. Proxy Re-encryption with Private Searching

Definition 3.1 Proxy Re-encryption with Private Searching (PRPS) scheme consists of seven randomized polynomial time algorithms as follows:

- Key Generation (KG): takes a suffciently large security parameter K_1 as input, and produces a public/private key pair (A_{pub}, A_{priv}) for a data owner A. We write $KG(K_1) \rightarrow (A_{pub}, A_{priv})$. Let K_2 be a suffciently large security parameter, we write $KG(K_2) \rightarrow (S_{pub}, S_{priv})$ for the cloud provider S, where S_{pub}, S_{priv} are public /private key respectively. Let K_3 be a suffciently large security parameter, we write $KG(K_3) \rightarrow (U_{pub}, U_{priv})$ for the data user U, where U_{pub}, U_{priv} are public/private key respectively.
- Encryption (E): this algorithm is performed by data owner A to encrypt the keyword $W_i (i \in Z^+)$ and message m. Correspondingly, two parts, KWEnc and EMBEnc constitutes Encryption.

  1) KWEnc: is a public key encryption algorithm that takes a public key A_{pub} and a key word $W_i (i \in Z^+)$ as inputs, and produces $W_i^{'}s$ ciphertext $C_{W_i} \in C_w$. We write $KWEnc(A_{pub}, W_i) \rightarrow C_{W_i}$

  2) EMBEnc: is a public key encryption algorithm that takes public keys S_{pub}, A_{pub} and message $m \in M$ as inputs, and produces $m's$ ciphertext C_m. We write $EMBEnc(S_{pub}, A_{pub}, m) \rightarrow C_m$.
- Re-Encryption Key Generation (RG): A data owner takes private key A_{priv} and user's public key U_{pub} as inputs, and produces the re-encryption key $rk_{A \rightarrow U}$. We write $RG(A_{priv}, U_{pub}) \rightarrow rk_{A \rightarrow U}$.
- TCompute： User takes private key U_{priv} and a keyword W_j $(j \in Z^+)$ as inputs, and produces $W_j^{'}s$ trapdoor T_{W_j}. We write $TCompute(U_{priv}, W_j) \rightarrow T_{W_j}$.
- Re-Encryption (R): The cloud provider takes re-encryption key $rk_{A \rightarrow U}$, ciphertext C_m and some intermediate result $\square$ as the inputs, and produces ciphertext $C_m^{'}s$ re-encrypted ciphertext C_U. We write $\mathrm{Re}-Encryption(\square, rk_{A \rightarrow U}, C_m) \rightarrow C_U$.
- Test：The cloud provider takes re-encryption key $rk_{A \rightarrow U}$, an encrypted keyword C_{W_i} and a trapdoor T_{W_j} as inputs, and produces "1" if $W_i = W_j$ or "0" otherwise. This algorithm is to check whether the ciphertext C_{W_i} matches the trapdoor T_{W_j}.
- Decryption (D): The user takes private key U_{priv} and re-encrypted ciphertext C_U as inputs, and outputs the plaintext m.

Definition 3.2 (Semantic Security of KWEnc): Given a public key encryption algorithm KWEnc which encrypts keywords using A_{pub}, let $\mathcal{A}_1$ be a polynomial time IND-CPA adversary that can adaptively ask for the trapdoor T_{W_i} for any keyword $W_i \in W$ of its choice. $\mathcal{A}_1$ first chooses two keywords W_0 and W_1, which are not to be asked for trapdoors previously, and sends them to KWEnc. And then KWEnc picks a random element $b_1 \in \{0,1\}$ and gives $\mathcal{A}_1$ the ciphertext $C_{W_{b_1}} \leftarrow KWEnc(A_{pub}, W_{b_1})$. Finally, $\mathcal{A}_1$ outputs a guess $b_1^{'} \in \{0,1\}$ for b_1. We define the advantage of $\mathcal{A}_1$ in breaking KWEnc as $Adv_{\mathcal{A}_1}(k) = \left| \Pr[b_1 = b_1^{'}] - \frac{1}{2} \right|$. KWEnc is semantically secure if for any polynomial time adversary $\mathcal{A}_1$, $Adv_{\mathcal{A}_1}(k)$ is negligible.

Definition 3.3 (Semantic Security of EMBEnc): Given a public key encryption algorithm EMBEnc which encrypts the message using A_{pub} and S_{pub}. Let $\mathcal{A}_2$ be a polynomial time IND-CPA adversary that can adaptively ask for the ciphertext for any message $m_i \in M$ of its choice. We use subscript T to denote the target user, x to denote the adversarial users, and h to denote the honest users (other than T). The input marked with a '*' is optional. $\mathcal{A}_2$ first chooses two messages m_0 and m_1, which are not to be asked for the ciphertext previously, and sends them to EMBEnc. And then EMBEnc picks a random $b_2' \in \{0,1\}$ and gives $\mathcal{A}_2$ the ciphertext $C_{m_{b_2}} = EMBEnc(A_{pub}, S_{pub}, m_{b_2})$. Finally, $\mathcal{A}_2$ outputs a guess $b_2' \in \{0,1\}$ for b_2. That is, for all PPT algorithms A_k,

$$
\begin{aligned}
\Pr[(pk_T, sk_T) &\leftarrow KG(1^k), \{(pk_x, sk_x) \leftarrow KG(1^k)\},\\
\{rk_{x \to T} &\leftarrow RG(pk_x, sk_x, pk_T, sk_T^*)\},\\
\{(pk_h, sk_h) &\leftarrow KG(1^k)\},\\
\{rk_{T \to h} &\leftarrow RG(pk_T, sk_T, pk_h, sk_h^*)\},\\
\{rk_{h \to T} &\leftarrow RG(pk_h, sk_h, pk_T, sk_T^*)\},\\
(m_0, m_1, \mapsto) &\leftarrow A_k(pk_T, \{(pk_x, sk_x)\}, \{pk_h\}, \{rk_{x \to T}\}, \{rk_{T \to h}\}, \{rk_{h \to T}\}),\\
b_2 &\leftarrow \{0,1\}, b_2' \leftarrow A_k(\mapsto, EMBEnc(pk_T, m_{b_2})):\\
b_2 &= b_2'] < 1/2 + 1/poly(k)
\end{aligned}
$$

We define the advantage of $\mathcal{A}_2$ in breaking EMBEnc as $Adv_{\mathcal{A}_2}(k) = \left|\Pr[b_2 = b_2'] - \frac{1}{2}\right|$. We say that EMBEnc is semantically secure if for any polynomial time adversary $\mathcal{A}_2$, $Adv_{\mathcal{A}_2}(k)$ is negligible.

Definition 3.4 (Semantic Security of PRPS): Given an PRPS scheme consisting of KWEnc and EMBEnc, it takes a security parameter K as input and runs the key generation algorithm KG to generate the public/private key pairs (A_{pub}, A_{priv}), (S_{pub}, S_{priv}) and (U_{pub}, U_{priv}). Given an adversary $\mathcal{A}$ consisting of two polynomial time algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$, $\mathcal{A}_1$ initiates attacks on KWEnc and $\mathcal{A}_2$ initiates attacks on EMBEnc. We say that the PRPS Scheme is semantically secure if for any adversary $\mathcal{A}$, $Adv_{\mathcal{A}}(k) = Adv_{\mathcal{A}_1}(k) + Adv_{\mathcal{A}_2}(k)$ is negligible.

4. Construction for PRPS

We assume that the scheme is composed of the following entities, the data owner, data users, and cloud providers. To access data files shared by the data owner, data users download data files of their interest from cloud providers and then decrypt. The users are assumed to have the only access privilege of data file reading. The cloud providers are assumed to have abundant storage capacity and computation power.

In our scheme, cloud providers are viewed as "*honest but curious*", which means they follow the proposed protocol in general, but try to find out as much secret information as possible. Cloud providers might collude with malicious users for the purpose of harvesting file contents when it is highly beneficial. Communication channel between the data owner/users and cloud providers are assumed to be secured. Users may work independently or cooperatively.

The main design goal is to help the data users achieve efficient private querying and downloading the encryted files stored in cloud providers. The data owner won't need to re-encryt the files in cloud provider for different users. We also want to prevent cloud providers from being able to learn both the data file contents and user queries information.

Suppose data owner A is about to store an encrypted file with keywords $W_1, \ldots W_l$ on a cloud storage S, where $l \in Z^+$. Keywords may be words in headline or stored date, and are relatively small. A encrypts the file message using his public key A_{pub}, the cloud storage's public key S_{pub}. And then A encrypts keywords $W_1, \ldots W_l$ using his public key A_{pub}. The file deposited in the cloud storage S by the data owner A is as follows:

$$MSG_{U2S} = [EMBEnc(A_{pub}, S_{pub}, m), KWEnc(A_{pub}, W_1), \ldots, KWEnc(A_{pub}, W_l)]$$

where $EMBEnc$, $KWEnc$ are public key encryption algorithms. Finally, A appends to the encrypted file message with all the encrypted keywords and sends MSG_{U2S} to S.

Given a suffciently large security parameter $K \in Z^+$, two groups G_1 and G_2 of prime order q, and a bilinear map $e: G_1 \times G_1 \rightarrow G_2$, $g, h \in G_1$, $Z = e(g,g) \in G_2$, where g is a generator of G_1. Then it chooses two hash functions H_1, H_3 : $\{0,1\}^* \rightarrow G_1^*$, hash function $H_2: G_2 \rightarrow \{0,1\}^{\log q}$, and hash function $H_4: G_2 \rightarrow \{0,1\}^n$ for some n, where H_1, H_2, H_3 and H_4 are random oracles. Finally, it picks three random elements $a, b, c \in Z_q^*$ and computes g^a, g^b and g^c. The plaintext space includes $M \in \{0,1\}^n$ and $W \in \{0,1\}^*$. The ciphertext space includes $C_M \in G_1^* \times \{0,1\}^n$ and $C_W \in G_2$.

- Key Generation (KG): The data owner A's public key is $A_{pub} = g^a$ with the corresponding private key $A_{priv} = a$; the user U's public/private key is $U_{pub} = g^b$, $U_{priv} = b$ respectively. The cloud provider S's public key is $S_{pub} = g^c$ with the corresponding private key $S_{priv} = c$.
- Encryption (E): This encryption algorithm consists of KWEnc and EMBEnc. The data owner first picks a random element $r \in Z_q^*$.

  1) KWEnc(E_1) ： To encrypt m's keywords $W_1,...,W_k$ $(k \in Z^+)$ under a data owner's public key g^a and a random element r, it computes $H_2(e(g^a, H_1(W_i))^r)$, where $W_i \in \{W_1,...,W_k\}$, sets the ciphertext $C_{W_i} = H_2(e(g^a, H_1(W_i))^r)$.

  2) EMBEnc (E_2): To encrypt the file message m under data owner's public key g^a, cloud provider's public key g^c and random element r, it picks a random element $\sigma \in \{0,1\}^n$, and computes $u_1 = h^r$, $u_2 = \sigma \oplus H_4(e(h^a, g^c)^r)$, $u_3 = m \cdot e(H_3(\sigma), g^a)^r$, and sets the ciphertext $C_m = (u_1, u_2, u_3)$.
- Re-Encryption KeyGeneration (RG): Data owner A delegates to user U by publishing the re-encryption key $rk_{A\rightarrow U} = g^{abr}$, computed with U's public key g^b.
- Tcompute: To retrieve the file containing keyword $W_j (j \in Z^+)$, user computes the trapdoor $T_{W_j} = H_1(W_j)^{1/b}$ using his/her private key $U_{priv} = b$, then sends the trapdoor to the cloud provider.
- Re-Encryption (R): to change the ciphertext $C_m = (u_1, u_2, u_3)$ for A into a ciphertext $C_U = (u_3, u_4)$ for U under the re-encyption key $rk_{A\rightarrow U} = g^{abr}$, it computes $u_4 = e(H_3(\sigma), rk_{A\rightarrow U}) = e(H_3(\sigma), g^{abr})$. The cloud provider sends C_U to the user.

Note. Since $\sigma = u_2 \oplus H_4(e(h^a, g^c)^r) = u_2 \oplus H_4(e(g^a, h^r)^c)$, the cloud provider can compute the intermediate value σ with its private key c.

- Test：To determine whether a given file contains keyword W_j ,the cloud provider tests whether $C_{W_i} = H_2(e(rk_{A\rightarrow U}, T_{W_j}))$. If so, $Test(rk_{A\rightarrow U}, C_{W_i}, T_{W_j})$ outputs 1, and 0 otherwise.

  Note. If $W_i = W_j$, since $C_{W_i} = H_2(e(g^a, H_1(W_i))^r)$, then
  $C_{W_i} = H_2(e(g^a, H_1(W_j))^r) = H_2(e(g^{abr}, H_1(W_j)^{1/b})) = H_2(e(rk_{A\rightarrow U}, T_{W_j}))$
- Decryption (D) : Given the ciphertext $C_U = (u_3, u_4)$, it computes $m = u_3 / (u_4)^{\frac{1}{U_{priv}}} = u_3 / (u_4)^{\frac{1}{b}}$ to recover the message m.

  Note that: $\dfrac{u_3}{(u_4)^{\frac{1}{b}}} = \dfrac{m \cdot e(H_3(\sigma), g^a)^r}{(e(H_3(\sigma), g^{ab})^r)^{\frac{1}{b}}} = \dfrac{m \cdot e(H_3(\sigma), g^a)^r}{e(H_3(\sigma), g^a)^r} = m$.

5. Security Analysis

Lemma 5.1 (Privacy for Keyword) Let H_1 be a random oracle from $\{0,1\}^*$ to G_1^* and H_2 be a random oracle from G_2 to $\{0,1\}^{\log q}$. Suppose $\mathcal{A}_1$ be an IND-CPA adversary that has the advantage ϵ_1 in breaking KWEnc. Suppose $\mathcal{A}_1$ makes at most $q_{H_2} > 0$ hash queries to H_2 and at most $q_T > 0$ trapdoor queries. Then there is an algorithm B_1 that solves the BDH problem with the advantage at least $\epsilon_1^{'} \geq 2\epsilon_1 / \{e \cdot q_{H_2} \cdot (1 + q_T)\}$, and a running time $O(time(\mathcal{A}_1))$.

Lemma 5.2 (Privacy for Message) Let H_3 be a random oracle from $\{0,1\}^*$ to G_1^* and H_4 be a random oracle from G_2 to $\{0,1\}^n$. Let $\mathcal{A}_2$ be an IND-CPA adversary that has the advantage ϵ_2 against EMBEnc. Suppose $\mathcal{A}_2$ makes $q_{H_3} > 0$ hash function queries to H_3 and $q_R > 0$ queries to $Request$. Then there is an algorithm B_2 that solves the BDH problem with the advantage at least $\epsilon_2^{'} \geq 2\epsilon_2 / q_{H_3} q_R$ and a running time $O(time(\mathcal{A}_2))$.

Theorem 5.1 (Security for PRPS) Suppose the hash functions H_1, H_2, H_3 and H_4 are random oracles. Let $\mathcal{A}$ be an IND-CPA adversary consisting of two polynomial time algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$. Let $\mathcal{A}_1$ be an IND-CPA adversary that has the advantage ϵ_1 in breaking KWEnc. Suppose $\mathcal{A}_1$ makes $q_T > 0$ trapdoor queries and $q_{H_2} > 0$ hash queries to H_2. Let $\mathcal{A}_2$ be an IND-CPA adversary that has the advantage ϵ_2 against EMBEnc. Suppose $\mathcal{A}_2$ makes $q_{H_3} > 0$ hash function queries to H_3 and $q_R > 0$ queries to $Request$. Let $\mathcal{A}$ be an IND-CPA adversary that has the advantage $\epsilon = \epsilon_1 + \epsilon_2$ against the PRPS scheme. Then there is an algorithm $\mathcal{B}$ that solves the BDH problem with the advantage at least:

$$Adv_{\mathcal{B}} \geq 2\epsilon_1 / \{e \cdot q_{H_2} \cdot (1 + q_T)\} + 2\epsilon_2 / q_{H_3} q_R$$

Here $e \approx 2.71$ is the base of the natural logarithm. The running time of $\mathcal{B}$ is $O(time(\mathcal{A}))$.

Due to page limitation, the details of formal security proof and some remarks are provided in the full version.

6. Conclusions

In this paper, we propose an efficient proxy re-encryption with private searching (PRPS) scheme in the untrusted cloud. We exploit proxy re-encryption and uniquely combining it with techniques of public key encryption with keyword search and dual receiver cryptosystem. PRPS allows users and data owners to query and access files storaged in untrusted cloud provider, while maintainning query privacy and data privacy. It allows user to decrypt the files efficiently. The PRPS scheme is proven semantically secure in the random oracle model.

Acknowledgements

This work is partially supported by National High Technology Research and Development Program of China (863 Program) under Grant No. 2009AA01Z423 and the Fundamental Research Funds for the Central Universities under Grant No. 2009JBM004.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A berkeley view of cloud computing," University of California, Berkeley, Tech. Rep. USB-EECS-2009-28, Feb 2009.

[2] Shucheng Yu, Cong Wang, Kui Ren and Wenjing Lou, "Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing," Proceedings of INFOCOM 2010. IEEE,2010.

[3] G. Ateniese, K. Fu, M. Green, S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," ACM Transactions on Information and System Security (TISSEC). 9 (1) (2006) 1–30.

[4] D. Boneh, G. Crescenzo, R. Ostrovsky, and G. Persiano, “Public Key Encryption with Keyword Search,” Proceedings of Eurocrypt 2004, LNCS 3027, Springer-Verlag. 2004. pp. 506-522.
[5] T. Diament, H. K. Lee, A. D. Keromytis and M. Yung, “The Dual Receiver Cryptosystem and its Application,” Proceedings of the ACM CCS 2004, pp. 330-343.
[6] Qin Liu, Guojun Wang, Jie Wu, “An Efficient Privacy Preserving Keyword Search Scheme in Cloud Computing,” CSE’09,vol.2, 2009 , pp. 715 - 720.
[7] J. Shao, Z. Cao, X. Liang, H. Lin, “Proxy re-encryption with keyword search,” Inf. Sci. 180 (2010) 2576–2587.