基于双重加密的敏感数据限时访问研究

陈 伟，王 燚，秦志光，刘鑫忠

(电子科技大学信息与软件工程学院 成都 610054)

【摘要】在云外包存储的背景下，针对外包存储中共享敏感数据的定时删除问题，提出基于双重加密的敏感数据限时访问方案。首先对称加密待共享的敏感数据文件，随后对加密后的文件进行随机分割提取，形成提取密文分量和封装密文分量；然后采用限时属性基加密算法对对称密钥和提取密文分量进行加密，生成访问控制对象；最后将访问控制对象同封装密文分量一同上传至云服务器。通过该方案，授权用户能够在限时属性基加密的访问时限窗口中解密访问控制对象，获取对称密钥和提取密文分量，并合成原始密文，恢复明文。访问时限窗口过期后，任何用户都无法属性基解密访问控制对象，获取对称密钥，恢复明文，从而实现敏感数据的定时删除。通过敌手攻击模型，分析并证明了该方案的安全性。

关 键 词 属性基加密； 密文提取； 外包存储； 敏感数据； 定时删除

中图分类号 TP399 **文献标志码** A doi:10.3969/j.issn.1001-0548.2017.03.018

Research on Timed Access of Sensitive Data Based on Dual Encryption

CHEN Wei, WANG Yi, QIN Zhi-guang, and LIU Xin-zhong

(School of Information and Software Engineering, University of Electronic Science and Technology of China Chengdu 610054)

Abstract A timed access solution of sensitive data based on dual encryption scheme is proposed to solve the problem of timed deletion of shared sensitive data stored in outsourcing storage. In our solution, the shared sensitive data file is encrypted by symmetric encryption, and the encrypted file is randomly divided to form the extracted cipher component and the encapsulated cipher component. Then, the attribute based timed encryption algorithm is used to encrypt the symmetric key and extracted cipher component, which can generate access control object. Finally, the encapsulated cipher component is combined with the access control object and sent into the cloud server. By this scheme, the authorized user can decrypt the access control object in the time limitation, obtain the symmetric key and extract cipher component, compose the original ciphertext, and recover the plaintext. Once access windows period expire, any users are unable to decrypt the access control object, get the symmetric key, recover the plaintext, so as to realize the timed destruction of sensitive data. The security of the scheme is analyzed and proved by the adversary attack model.

Key words attribute based encryption; ciphertext extraction; outsourcing storage; sensitive data; timed deletion

随着信息化的日益推进，越来越多的数据在网络上产生。云计算的兴起为数据存储提供了极大的方便，于是，越来越多的人选择将数据存放在云服务器中，并且将数据共享给其他好友。考虑到数据的敏感性和隐私性，用户更愿意这样的共享是有时限的，而不是永久性的共享。从时间的维度，一个典型的数据生命周期包括数据生成、数据发布、数据使用和数据删除，数据使用后需要销毁来保障敏感信息不被泄露，因此，需要一种适当的机制来实现数据使用后的删除；从数据拥有者的角度，他往往不希望数据一直被暴露在不可信的云服务器中，因此需要设计一种方案来满足数据拥有者的需求，在他不希望数据被共享时，将数据删除。综合考虑上述两点，删除是保障数据隐私性的一个重要手段。

找到敏感数据存储在云服务器中所有的备份文件，然后逐一删除，实现了物理层面的删除。然而，云存储的结构相对复杂，一般都采用多重备份的方式来保障数据的高可用性，要找到云中所有的数据备份相对比较困难。退而求其次，寻找一种限时的加密方案，当数据拥有者设定的访问时间过期后，即使数据使用者获得了共享的数据文件，也不能解密文件的内容，从而在逻辑层面上解决了敏感数据的

收稿日期：2015－12－18；修回日期：2016－05－09
基金项目：国家自然科学基金(61520106007)
作者简介：陈伟(1978－)，男，博士，副教授，主要从事网络与信息安全方面的研究.

定时删除问题。

1 相关工作

文献[1]提出了Vanish隐私数据自毁系统，通过将加密密钥秘密分享到分布式Hash表DHT (distributed hash table) 网络中，利用DHT网络的定期自动覆盖功能实现密钥的删除，从而实现密文不可读；文献[2]提出的Safe Vanish方案在Vanish的基础上，可以抵抗“跳跃攻击”和“嗅探攻击”；文献[3]提出了SSDD(secure self-destruction scheme for electronia data)方案，文献[4-5]提出了ISDS(IBE-based secure document self-destruction)方案和ISS (IBE-based secure self-destruction)方案，分别对文献[1]的方法加以不同角度的改进。此外，文献[6]提出了基于身份加密和DHT网络的定时发布加密方案。DHT网络上的节点虽然有自我更新的特性，可以用来自动覆盖和删除部分数据，但是数据删除的确定性、及时性会受到DHT网络的性能和更新周期的钳制。

在某些应用场景下，数据拥有者希望其数据在预设的某一个时间点访问生效，而不是已进入云服务器就立即生效，于是文献[7]提出了一个解决限时加解密问题的算法。文献[8-12]设计了一系列定时发布加密算法(timed-release encryption, TRE)，这些算法需要共享时间服务器来取得绝对的时间参考和预设的时间发布密钥，一旦到了预设的时间点，授权用户就可以从时间服务器获得或重构解密密钥以解密受保护的隐私数据。文献[13]进一步给出了一种基于时间的代理重加密方案，该方案在私钥中嵌入时间戳，当时间过期后，私钥不可用，从而实现数据的访问撤销。这一系列算法虽然解决了敏感数据在预定义的时间之后才能被授权用户解密，但没有解决敏感数据生命周期结束后的安全隐私保护问题。

为了实现授权期的细粒度访问控制，以及敏感数据在过期后不被任何人访问，本文设计了基于双重加密的敏感数据限时访问方案(timed access of sensitive data, TASD)。该方案首先将明文文件进行一次对称加密，实现对文件的机密性保护；然后对密文文件实施随机提取(random ciphertext extra, RCE算法)，再对对称密钥和提取的小部分文件进行一次限时属性基加密(attribute-based timed encryption, ATE算法)，通过随机提取破坏文件完整性，并在属性加密中引入一个时间参数，这样既实现了细粒度的访问控制，又实现了过期后对称密钥和提取文件分量的不可恢复，最终实现敏感数据不被任何人访问。

2 模型与假设

2.1 系统模型

本文主要关注如何通过到期时间的设置来完成云服务器中敏感数据到期后不可访问：到期前，通过属性基加密实现细粒度的访问控制；生命周期中，通过文件提取和对称加密保证敏感数据的保密性；到期后，通过属性加密中的时间参数来实现密钥的不可获得。系统的模型如图1所示。

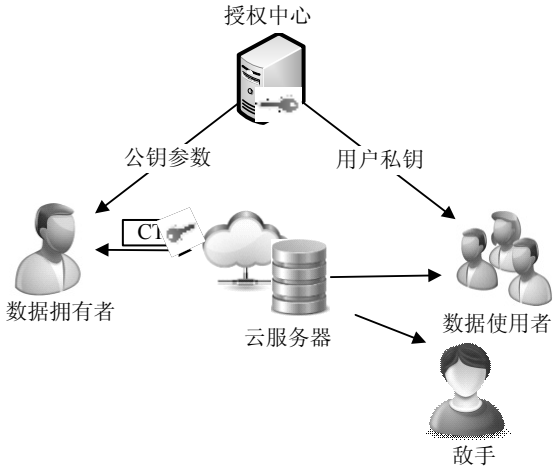

图1　系统模型

系统中存在6种参与实体分别是：1) 数据拥有者，需要将自己的数据上传到云服务器中，这些数据中含有某些敏感和隐私信息，并且这些数据将会与他们的朋友进行共享。共享数据将会被外包存储到云服务器中。数据拥有者也将对上传数据进行加密，并定义数据可以被访问的时间间隔。2) 授权中心，这是一个对所有参与者均可信的参与实体，负责生成、分发和管理所有的私钥。3) 时间服务器，这个参与实体与其他所有参实体都没有任何交互，只是一个时间参考服务器，负责精确的时间发布，保证所有实体使用的时间是一致的。4) 数据使用者，通过身份认证后可在授权时间内访问数据拥有者存储在云服务器上的敏感数据。5) 云服务器，它包含有几乎无限大的存储空间，用来存储和管理系统中所有敏感数据，其他所有实体都可以将他们的数据外包到云服务器上。6) 敌手，引入这个实体是为了建立系统安全模型，然后进行安全性分析。本文中假设敌手是一个多项式时间敌手。

针对上述应用场景中的共享数据的安全自毁，设计TASD方案。该方案将会实现在授权期内敏感数

据的细粒度访问控制，在授权期结束后，任何数据使用者将无法获取共享数据。

2.2 贡献

1) 共享的敏感数据在生命周期内的可用性。如果处在生命周期时间段内，共享数据应该能够正确地被指定的数据使用者解密获得。

2) 实现细粒度的访问控制。通过引入属性加密，对用户分类，在敏感数据的生命周期中，数据拥有者可以预先定义可以访问共享数据的用户。

3) 过期后不能被任何人访问。方案可以实现前向安全，即便敌手构造了数据拥有者定义的属性集，一旦过期共享数据对任何人不可得，从而对共享数据的机密性和隐私提供生命周期内的安全保护。

4) 密钥管理简单高效。方案采用属性加密这一非对称加密来实现细粒度的访问控制，因此具有更简单高效的密钥管理机制。

5) 能够抵御多项式敌手的攻击。即便敌手在生命周期内伪造出了正确的属性集，获得了数据，由于密文经过随机提取，敌手将无法还原共享数据。

3 TASD方案

首先对系统的工作流程进一步细化，阐述一个明文文件如何加密、提取，最终上传到云服务器，以及云服务器的数据如何恢复成为一个完整的明文文件；然后介绍这个系统中的关键算法，给出其工作原理，并进行形式化描述。

为了叙述方便，对算法必须使用的公式、符号进行定义说明：

1) 有限域加法循环群 G_1 和乘法群 G_2，均为大素数 q 阶，g 为 G_1 的生成元，一个双线性函数 $e:(G_1, G_1)\to G_2$；

2) 哈希函数：$H:\{0,1\}^*\to G_1$，$H_1=G_2\to\{0,1\}^*$；

3) 将数据拥有者定义的用户属性集合抽象成为 Ω，数据拥有者和数据使用者都可得；

4) 对称加密算法E，对称加密密钥 A_k 称为访问密钥。

3.1 系统描述

1) 系统初始化：在这一阶段，数据使用者选择一个安全参数，并定义访问该文件需要具有的属性集 Ω，给定一个系统主密钥 $x\in Z_q^*$，由授权中心产生系统公钥 $(g,y=xg)\in G_1^2$，时间服务器周期性的发布时间密钥更新信息 $xH(T)$，其中 $T\in\{0,1\}^*$ 表示当前时刻。

2) 密文提取：数据拥有者首先从授权中心获得一个对称密钥，称为访问密钥 A_k，用 A_k 对明文文件进行加密，得到密文文件C。通过RCE算法将密文C分解成为两部分 C_e 和 C_o，其中 C_e 将会被加密存储到云服务器上，而 C_o 也将会存储在云服务器上，在 C_e 和 C_o 之间建立映射关系。

图2为系统的加/解密过程图，向下表示为加密或分解，向上表示为解密或合并。

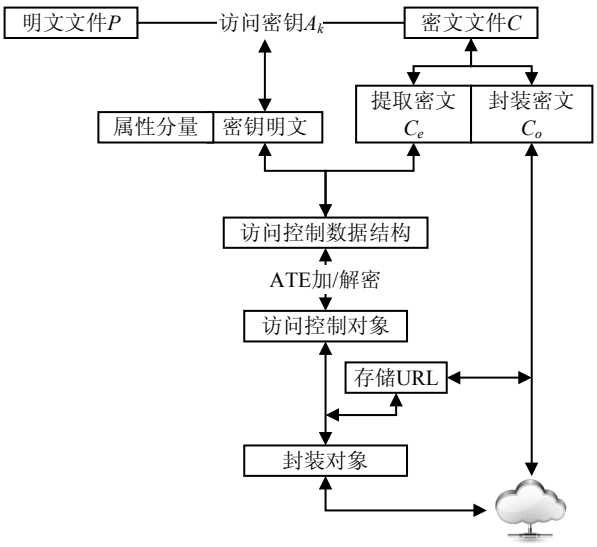

图2 系统加解密过程图

如图2所示，对访问密钥 A_k 和提取密文 C_e 实施ATE算法，得到一个访问控制对象(access control object, ACO)，这一步需要有授权中心和时间服务器的参与。封装密文 C_o 存储在云服务器中的URL会被记录下来，与ACO一起打包形成新的封装对象存储在云服务器中，并传给数据使用者。

解密过程与上述加密过程相反。数据使用者获取封装对象，分离分装对象将会获得封装密文的URL，进而获得封装密文 C_o；授权中心将根据时间服务器的当前时间和用户的属性对其进行检查，判断是否为其生成解密密钥；如果满足解密条件，将会获得访问密钥 A_k 和提取密文 C_e；进一步地合并和解密就可以获得最初的明文文件。

3) 限时属性基加密：将时间参数作为一个属性加入密钥生成的运算，为了保证在过期后密钥不可得，就要保证系统时间在设定的时间参数之前，授权中心会执行这样的比较，根据比较结果，决定是否发布密钥给数据使用者。

4) 授权期间的访问控制：系统中仍然保留着用户属性，数据拥有者可以预先定义数据使用者应该具有的属性，这些属性最终会反映在密钥中。只有满足数据拥有者定义的属性，数据使用者才可以获

得ACO。访问控制策略的实施在属性层面，因此是细粒度的。

3.2 随机密文提取算法

密文分量的提取采用随机矩阵的方式来进行。随机构造一个随机提取矩阵，存储于授权中心，用来进行密文分量提取。数据拥有者在加密了明文文件之后，进行密文分量提取。RCE算法的输入为一个 $n\times n$ 秘密随机矩阵以及原始的密文文件，输出为两个文件分别称为提取密文 C_e 和封装密文 C_o，该算法的工作流程如图3所示。

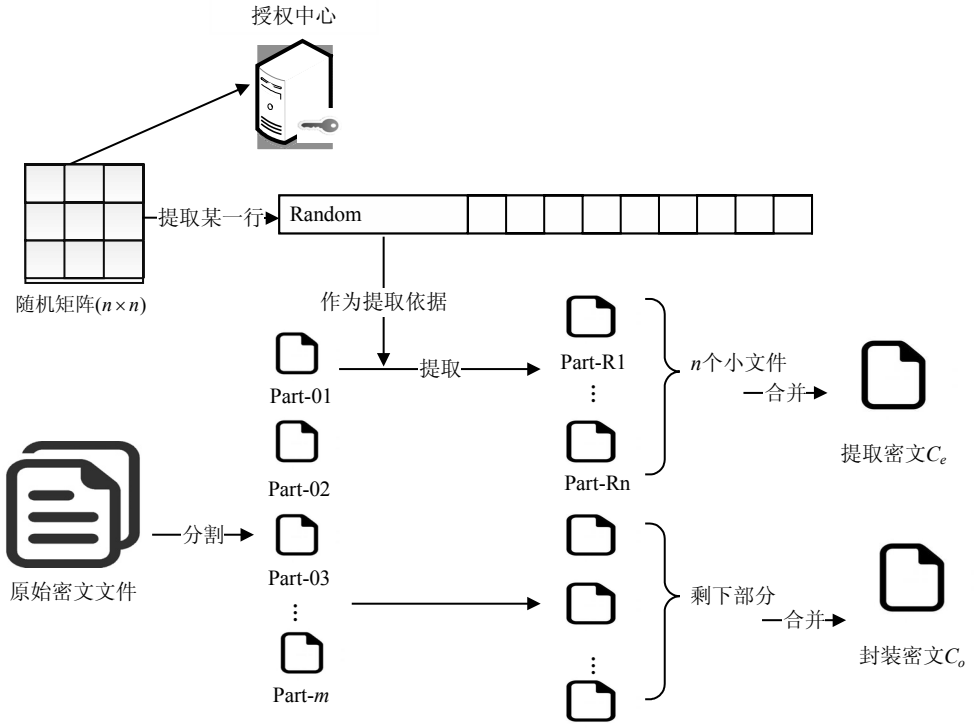

图3　RCE算法的工作流程

RCE算法的工作步骤如下：

1) 按照要求生成随机矩阵，该矩阵将会由授权中心秘密地保存，随机矩阵的引入，是为了省去每一次随机提取向量的生成，从而提高效率；要求矩阵中没有重复的行，矩阵中所有的元素都必须是充分随机的，得到一个 $n\times n$ 秘密矩阵；

2) 确定一个分割粒度 g (如2 KB)作为密文分量的大小，根据密文文件的大小 f，分割粒度的大小也应该有所不同，按照这一分割粒度将文件分成分割成 m 个密文分量，其中 $m=f/g$，将会输出 m 个密文分量；

3) 将步骤2)中输出的 m 个密文分量从1开始标号，最后一个文件块标号为 m，标号是为了给随机提取提供依据，将会输出标了号的 m 个密文分量；

4) 在随机矩阵中随机地选取某一行，称为随机密文提取向量，将该向量中的元素值作为提取密文分量的依据，将会输出一个随机密文提取向量；

5) 根据步骤4)输出的随机密文提取向量中元素值，在 m 个密文分量中的对应位置，选取密文分量，提取的密文分量的个数就是向量中元素的个数 n，将输出两组密文分量，分别是提取出的密文分量和剩余的密文分量；

6) 将步骤5)输出的 n 个提取出的密文分量和 $(m-n)$ 个剩余的密文分量进行合并，最后分别输出提取密文 C_e 和封装密文 C_o。

经过上面的操作，两部分文件都将失去可读性，攻击者拿到了大块的 C_o 但却无法识别其中内容，也没有意义；小部分 C_e 可以通过属性加密之后，再上传到云服务器，这样将属性加密这一非对称加密应用于少量数据加密，保证了加解密效率和数据的机密性。

3.3 基于属性的定时加/解密算法

为了实现在特定时间内数据可以被正常访问，过期后不能再被访问，本文设计ATE算法。该算法的输入为访问控制数据结构A，输出为访问控制对象ACO。

1) 计算 $\omega_E=H(\Omega)+H(T)$，其中 T 是由数据拥有者按定义的一个时间段，开始于预期发布时间，

止于数据的过期时间；

2) 随机选择一个变量 $r \in Z_q^*$，计算双线性对 $\omega = e(y,\omega_E)^r = e(xg,\omega_E)^r = e(g,\omega_E)^{xr}$；

3) 计算访问控制数据结构A的密文 $C_A =< rg, A \oplus H_1(\omega) >$。

通过上述运算将当前时间反映在访问控制数据结构密文 C_A 中，这样的定义可以在解密时保证时间过期后，无法完成解密。

解密访问控制对象ACO将执行如下过程：

1) 用户从授权中心获得私钥 $k_\Omega = xH(\Omega) \in G_1$，获得当前时间 T_1，并计算 $xH(T_1)$，此时的 T_1 是一个时间点，若 $T_1 \in T$，则可以得到正确的 $xH(T_1)$，否则将无法得到结果或者得到错误的结果；

2) 计算 $\omega_D = xH(\Omega) + xH(T_1)$，即 $\omega_D = x\omega_E$，$\omega^* = e(rg,\omega_D)$。由于 $\omega_D = x\omega_E$，则 $\omega^* = \omega$，从而 $A = C_A \oplus H_1(\omega^*)$。

4 安全性分析

文献[6-9]的方案主要考虑追溯性攻击，均假设敌手不是实时的，存储在云服务器中的封装对象在生命期内不受攻击。本文假设云服务器不可信，封装对象在任意时刻都可能受到敌手攻击。主要分析如下两个方面的敌手攻击：一是云服务器的密码分析攻击；二是针对分割文件算法的攻击。现在考虑两者的并发攻击，可以分为如下3个阶段：

1) 在封装对象预期发布时间之前

假设敌手从云服务器获取封装密文 C_o，又获取了封装对象。敌手如果想要获取完整的明文数据，就必须解密封装对象来获取对称密钥。但是，当前时间由可信的时间服务器给出，当前时间小于预期发布时间，因此无法正确的构造时间更新密钥 $xH(T_1)$，就不能还原原始对称密钥。因此，敌手在该阶段的攻击无效。

2) 在封装对象的生命周期内

敌手可以顺利地获取访问控制数据结构，并且可以正确的构造出时间更新密钥 $xH(T_1)$，并且敌手通过伪造用户属性，从授权中心获取了私钥，解密得到提取密文 C_e，并且能够还原原始对称密钥 A_k。但由于密文文件的合成需要知道一个随机提取矩阵，这个矩阵对于敌手来说不可得，因此它无法获得密文文件，也就无法解密获得原始明文文件。因此敌手在该阶段的攻击无效。

3) 在封装对象过期后

敌手不能正确的构造时间更新密钥 $xH(T_1)$，攻击无效。

综上，TASD方案是安全的。

5 结 束 语

在云服务环境中，由于云服务器复杂的环境，共享的敏感数据需要在其使用过后得到确定性的销毁，从而保护数据拥有者的敏感和隐私信息。本文提出了一种TASD方案，引入随机密文提取以及限时属性基加密算法，将密文文件分割提取后，在此进行属性加密。通过限时属性基加密算法，可以实现细粒度的访问控制以及敏感数据的过期自毁，从而为共享的敏感数据提供全生命周期的隐私保护。安全性分析表明，该方案在合适的前提假设下，在不同的时间段，可以抵御来自多项式敌手的攻击。因此，该方案是安全的。

本文是基于双重加密的敏感数据限时访问的初步研究成果，是属性加密机制的实际应用，是数据生命周期安全性研究工作的一部分。为了进一步保证数据拥有者数据的安全性，下一步的研究工作将主要有：1) 设计更加合理的访问策略，实现生命周期内的细粒度的访问控制；2) 站在数据拥有者的角度，设计一种高效的动态更新删除时间的机制。

参 考 文 献

[1] GEAMBASU R, KOHNO T, LEVY A, et al. Vanish: Increasing data privacy with self-destruction data[C]//Prof of 18th USENIX Security Symp. Berkeley, USA: USENIX Association, 2009: 299-315.

[2] ZENG Ling-fang, SHI Zhan, XU Sheng-jie, et al. SafeVanish: an improved data self-destruction for protecting data privacy[C]//Prof of the 2nd Int Conf on Cloud Computing Technology and Science. Piscataway, NJ: IEEE, 2010: 521-528.

[3] WANG Guo-jun, YUE Fang-shun, LIU Qin. A secure self-destructing shemefor electronic data[J]. Journal of Computer and System Sciences, 2013, 79(2): 279-290.

[4] XIONG Jin-bo, YAO Zhi-qiang, MA Jian-feng, et al. A secure document self-destruction scheme with identity based encryption[C]//Prof of the 5th Int Conf on the Intelligent Networking and Collaboratives Systems. Piscataway, NJ: IEEE, 2013: 239-243.

[5] 熊金波, 姚志强, 马建峰, 等. 面向网络内容隐私的基于身份加密的安全自毁方案[J]. 计算机学报, 2014, 37(1): 139-150. XIONG Jin-bo, YAO Zhi-qiang, MA Jian-feng. et al. A secure self-destruction scheme with IBE for the internet content privacy[J]. Chinese Journal of Computers, 2014, 37(1): 139-150.

[6] 姚志强, 熊金波, 马建峰, 等. 云计算中一种安全的电子文档自毁方案[J]. 计算机研究与发展, 2014, 51(7):

1417-1423.
YAO Zhi-qiang, XIONG Jin-bo, MA Jian-feng, et al. A secure electronic document self-destruction scheme in cloud computing[J]. Journal of Computer Reseach and Development, 2014, 51(7): 1417-1423.

[7] RIVEST R L, SHAMIR A, WAGNER D A. Time-lock puzzles and timed-released crypto[EB/OL]. [2015-02-14]. http://dl.acm.org/citation.cfm?id=888615.

[8] CHAN A F, BLAKE I F. Scalable, server-passive, user-anonymous timed release cryptography[C]//Proc of the 25th on Distributed Computing Systems. Piscataway, NJ: IEEE, 2055: 504-513.

[9] CHALKIAS K, HRISTU-VARSAKELIS D, STEPHANIDES G. Improved anonymous timed-release encryption[C]// LNCS 4734: Proc of the 12th European Symp on Research in Computer Security. Berlin: Springer, 2007: 311-326.

[10] DENT A W, TANG Q. Revisiting the security model for timed-release encryption with pre-open capability[C]// LNCS 4779: Proc of the Information Security. Berlin: Springer, 2007: 158-174.

[11] KIKUCHI R, FUJIOKA A, OKAMOTO Y, et al. Strong security notions for timed-release public-key encryption revisited[C]//LNCS 5324: Proc of the Provable Security. Berlin: Springer, 2012: 88-108.

[12] CHOW S S, YIU S M. Timed-release encryption revisited[C]//LNCS 5324: Proc of the Provable Security. Berlin: Springer, 2008: 38-51.

[13] LIU Qin, WANG Guo-jun, WU Jie. Timed-based proxy re-encryption scheme for secure data sharing in a cloud computing environment[J]. Information Sciences, 2014, 258(3): 355-370.

编 辑 税 红