Identity-based encryption with outsourced revocation in cloud computing

摘要: 基于身份的加密(IBE)简化了公钥基础设施(PKI)上的公钥和证书管理，是公钥加密的重要替代方案。然而,IBE的一个主要效率缺陷是用户撤销期间在私钥生成器(PKG)上的开销计算。在传统的PKI环境中，有效的撤销已经得到了很好的研究，但是繁琐的证书管理正是IBE努力减轻的负担。

本文针对身份撤销的关键问题，首次将外包计算引入到IBE中，提出了一种服务器辅助下的可撤销IBE方案设置。在密钥发布和密钥更新过程中，我们的方案将大多数与密钥生成相关的操作卸载给密钥更新云服务提供商，只留下一定数量的简单操作供PKG和用户在本地执行。这个目标是通过使用一种新的抗合缝技术来实现的:我们为每个用户使用一个混合的私钥，其中包含一个AND门来连接和绑定标识组件和时间组件。此外，我们还提出了另一种构造方法，该构造方法在最近规范化的引用委托计算模型下是安全的。最后，我们提供了广泛的实验结果，以证明我们的建设的效率。

索引术语——基于身份的加密、撤销、外包、云计算。

1.介绍

基于身份的加密(IBE)是公钥加密(public key Encryption, PKI)的一种有趣的替代方案。公钥加密的目的是通过使用人类可理解的身份(如惟一名称、电子邮件地址、IP地址等)作为公钥，简化基于证书的公钥基础设施(PKI)中的密钥管理。因此,发送方使用IBE不需要查找公钥和证书，而是直接用接收者的身份加密消息。因此，从私钥生成器(private key Generator, PKG)获取与相应身份相关联的私钥的接收者能够解密这些密文。

尽管IBE允许任意字符串作为公钥，这被认为是比PKI更有吸引力的优势，但它需要一种有效的撤销机制。具体来说，如果某些用户的私钥受到攻击，我们必须提供一种方法将这些用户从系统中撤销。在PKI设置中，撤销机制是通过对证书附加有效期或使用相关技术组合[1][2][3]来实现的。然而，繁琐的证书管理正是IBE努力减轻的负担。据我们所知，虽然在PKI中已经对撤销进行了深入的研究，但是在IBE设置中，我们所知的撤销机制并不多。在[4]中，Boneh和Franklin建议用户定期更新他们的私钥，发送者使用与当前时间段连接的接收者身份。换句话说，所有的用户，无论他们的密钥是否被撤销，都必须定期与PKG联系，以证明他们的身份，并更新新的私钥。它要求PKG是在线的，所有事务都必须维护安全通道，随着用户数量的增加，这将成为IBE系统的瓶颈。

2008年，Boldyreva, Goyal和Kumar[5]提出了一个可撤销的IBE计划。该方案基于模糊IBE原语[6]的思想，利用二叉树数据结构记录叶节点的用户身份。因此，PKG下的键更新效率可以从线性显著降低到这种二叉树的高度(即用户数量的对数)。然而,我们指出,虽然二叉树引入能够达到一个相对高绩效,这将导致其他问题:1)包裹必须生成一个密钥对路径上的所有节点的身份叶子节点到根节点,导致复杂性对数的用户数量在系统发布一个私有密钥。2)系统中私钥的大小随着用户数量的对数增长，使得用户在私钥存储中存在困难。3)随着系统用户数量的增加，PKG需要维护一个节点较多的二叉树，这给全局系统带来了另一个瓶颈。

随着云计算的发展，用户可以从基于云的服务(如Amazon的EC2和Microsoft的Windows Azure)上购买按需计算。因此，它希望在IBE撤销中引入此类云服务的新工作范式，以解决上述效率和存储开销问题。一种简单的方法是将PKG s主密钥交给云服务提供商(csp)。然后，csp可以使用传统的密钥更新技术[4]简单地更新所有的私钥，并将私钥传输回未撤销的用户。然而，这种天真的方法基于一个不现实的假设，即csp是完全可信的，并且允许访问IBE系统的主密钥。相反，在实践中，公共云很可能位于相同的用户信任域之外，并且对用户的个人隐私很好奇。因此，如何设计一种安全的可撤销IBE方案，在不可信的CSP下减少PKG上的开销成为了一个难题。

在本文中，我们将外包计算引入到IBE撤销中，并在我们所知的范围内首次正式给出了外包可撤销IBE的安全定义。我们提出了一种方案，在密钥分发和密钥更新期间卸载所有与密钥生成相关的操作，只留下一定数量的简单操作供PKG和符合条件的用户在本地执行。在我们的方案中，与[4]中的建议一样，我们通过更新未撤销用户的私钥来实现撤销。但与这工作[4]非常连接时间与身份密钥生成/更新,需要重新发放整个unrevoked用户的私钥,我们提出一个新颖的collusion-resistant关键发行技术:我们使用了混合的为每个用户的私钥,一个和门是两个子组件连接和约束,即身份组件和组件。首先，用户能够获得标识组件和默认时间组件(即以PKG作为他/她的私钥发出密码匙)。然后，为了保持解密性，未撤销用户需要定期向新引入的名为密钥更新云服务提供商(Key Update Cloud Service Provider, KU-CSP)的实体请求时间组件的密钥更新。

与以前的工作[4]相比，我们的方案不需要重新发布整个私钥，只需要在专门化实体KU-CSP上更新它的轻量级组件。我们还规定1)在KU-CSP的帮助下，用户在key-update中不需要与PKG联系，即允许PKG在将撤销列表发送给KU-CSP后离线。2)用户与KU-CSP之间的密钥更新不需要安全通道或用户认证。

此外，我们考虑用半诚实的KU-CSP实现可撤销的IBE。为了实现这一目标，我们提出了一个安全增强的建设下，最近正式的参考计算委托(RDoC)模型[7]。最后，我们提供了广泛的实验结果，以证明我们的建议建设的效率。

本文组织如下。在第二节中，我们描述了我们的方案的初步内容。在第三节中，我们给出了该方案的系统模型和安全定义。第四节给出了建议的构造及其安全性分析。在第五节中，我们提出了RDoC模型下的安全增强构造。第六节给出了一个广泛的实验结果，以证明我们提出的结构的效率。最后，通过对第七节相关工作的回顾，得出第八节的结论。

2.准备

在本节中，我们将简要回顾一些加密背景和基于身份的加密。

1. 加密的背景

定义1 :(双线性映射）设G，GT为素数阶q的循环群，乘法地编写群动作。g是G的生成器。设e：G GGT是具有以下属性的地图：

非简并：存在g1，g2 G，其中e（g1，g2）= 1，换句话说，地图不会发送G中的所有对

定义2：（DBDH问题）决定Bilinear DiffieHellman（DBDH）的问题是，给定g，gx，gy，gz G为通常涉及两个实体，即PKG和用户（包括发送方和接收方）的IBE方案由以下四种算法组成。

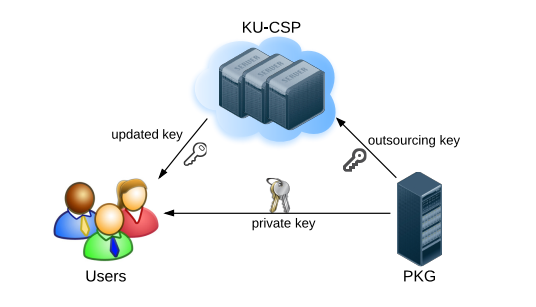


图1所示。具有外包撤销的IBE系统模型

B. 基于身份的加密

IBE方案通常涉及两个实体PKG和用户(包括发送方和接收方)，它由以下四种算法组成。

Setup(λ): 设置算法将安全作为输入参数λ和输出公钥PK和主密钥可注意包裹的主密钥是保密的。

KeyGen(MK; ID) : 私钥生成算法由PKG运行，PKG以主密钥MK和用户身份ID作为输入。它返回与标识ID对应的私钥SKID。

Encrypt(M; ID0) : 加密算法由发送方运行，接收方的身份ID0和要加密的消息M作为输入。输出密文CT。

Decrypt(CT; SKID0) : 解密算法由接收端运行，接收端以密文CT和其私钥SKID0作为输入。它返回消息M或错误.

IBE格式必须满足一致性的定义。具体来说，当算法KeyGen生成的私钥SKID被给定ID作为输入时，则解密(CT;其中CT =加密(M;ID)。

IBE的动机是简化证书管理。例如，当Alice在bob@company.com向Bob发送电子邮件时，她只是使用加密方法对邮件进行加密Bob的电子邮件地址“bob@company.com”，但不需要获得Bob的公钥证书。当Bob收到加密的电子邮件时，他在PKG上验证自己以获得其私钥，并使用该私钥阅读电子邮件。

3.问题陈述

A. 系统模型

我们在图1中给出了外包可撤销IBE的系统模型。与典型的IBE方案相比，采用了KU-CSP实现了对受损用户的撤销。实际上，KU-CSP可以设想为由第三方运行的公共云，通过网络将基本计算能力作为标准服务提供给PKG。通常，KU-CSP驻留在远离用户或PKG的地方，但是通过提供对基础设施的灵活的、甚至是临时的扩展，提供了一种减少PKG计算和存储成本的方法。当撤销被触发时，未撤销的用户必须要求KU-CSP更新其私有密钥的轻量级组件，而不是从[4]中的PKG重新请求私有密钥。虽然KU-CSP的部署涉及到很多细节，但在本文中，我们只是从逻辑上将其设想为一个计算服务提供商，并关注如何在不可信的KU-CSP下设计安全方案。

基于所提出的系统模型，我们可以定义外包可撤销的IBE方案。与传统的IBE定义相比，KeyGen;将加密和解密算法重新定义如下，以集成时间组件。请注意，在我们的定义中使用了两个列表RL和T L，其中RL记录已撤销用户的身份，而T L是过去和当前时间段的一个链表。

KeyGen（MK，ID，RL，TL）：PKG运行的密钥生成算法作为输入master主密钥MK，身份ID，撤销列表RL和时间列表T L.如果是ID RL，算法是中止。否则，它将私钥SKID =（IK [ID]，TK [ID] Ti）发送给用户，其中IK [ID]是私钥SKID的标识组件，TK [ID] Ti是其当前时间段的时间分量钛。另外，该算法将外包密钥OKID发送到KU-CSP。

加密（M，ID，Ti，PK）：发送方运行的加密算法将消息M，标识ID和时间段Ti作为输入。它输出密文CT。

解密（CT，SKID）：接收机运行的解密算法作为输入 在身份ID和时间段Ti和私钥SKID =（IK [ID]，TK [ID] Tj）下加密的密文CT。如果ID = ID且Ti = Tj，则输出原始消息M，否则输出。

另外，定义了两种算法，通过更新未撤销用户的私钥，实现KU-CSP的撤销。

撤销（RL，TL，{IDi1，...，IDik}）：PKG运行的撤销算法作为输入 - 撤销列表RL，时间列表TL和要撤销的身份集{IDi1，IDi2， 。。。，IDik}。它输出更新的时间段Ti + 1以及更新的撤销列表RL和时间列表TL。

KeyUpdate（RL，ID，Ti + 1，OKID）：由KU-CSP运行的密钥更新算法作为输入 - 撤销列表RL，身份ID，时间段Ti + 1和身份ID的外包密钥OKID 。如果用户的身份ID不属于RL，则输出私钥TK [ID] Ti + 1中的用户更新时间分量，否则输出。

在本文中，我们讨论用户撤销，即即使已经发布了私钥，如何剥夺用户的可解密性。为此，我们以巧妙的方式将一段时间嵌入私钥中以便撤销。具体地，在第II-B部分所示的相同示例中，Alice在我们的设置中不仅使用Bob的电子邮件地址“bob@company.com”加密消息，而且还使用当前时间段（例如，“Thu Jul 18 2013”​​）加密消息。当收到加密的电子邮件时，Bob然后从PKG获得由身份组件和时间段组件组成的私钥。通过这两个适当的组件，可以阅读电子邮件。

假设Bob受到了损害。然后，KU-CSP用新的时间段（例如，“2013年7月19日星期五”）更新所有其他用户的时间分量。从那时起，发送给Bob的消息应该使用Bob的电子邮件地址和更新的时间段进行加密。由于Bob没有与更新的时间段相对应的时间组件，因此Bob不能解密以下加密的消息，即使它们是针对他的。

设计外包可撤销IBE方案的挑战是如何防止Bob与其他未被发现的不诚实用户之间的勾结。具体来说，不诚实的用户（名为Eve）可以与Bob分享她更新的时间组件（即“2013年7月19日星期五”），并帮助Bob解密密文，即使Bob刚刚拥有前一个（即“Thu Jul 18 2013”）。我们将在稍后展示如何避免这种勾结。

我们假设所提出的系统模型中的KU-CSP是半固定的。具体来说，它将遵循我们的协议，但试图根据其拥有情况找出尽可能多的秘密信息。因此，两种类型的对手应被视为如下。

I型对手。它被定义为具有身份ID但在时间段Ti之前被撤销的好奇用户。这样的对手试图通过与其他用户勾结来从Ti（例如时间段Ti，Ti + 1，...）处或之后从他/她获得的密文获得有用信息，即使它们未被撤销。因此，允许为协作用户请求包括身份组件和更新时间组件的私钥。我们指出，假设KU-CSP是半信任的，I型攻击者无法获得任何用户的外包密钥。

II型对手。它被定义为一个好奇的KU-CSP，其旨在从时间段Ti处的某些目标身份ID的密文中获取有用信息。这样的对手不仅拥有系统中所有用户的外包密钥，而且还能够通过与具有身份ID的任何其他用户串通来获取用户的私钥。需要注意的是，为了使这种攻击合理，我们必须限制ID = ID。

综上所述，我们可以根据图2中的设置，分别为type-I和type-II对手定义CCA安全游戏。假设Ai 是 type-i型对手，i取值范围是I ，II。然后，利用外包撤销方案攻击IBE的优势，



定义3:对于第i类和第ii类攻击者，如果没有多项式有界的对手在安全博弈中对挑战者具有不可忽略的优势，则采用外包撤销方案的基于身份的加密在语义上可以抵御自适应选择密文攻击(IND-ID-CCA)。

最后,除了CCA的安全,我们也指定1)外包撤销的IBE方案IND-ID-CPA安全(或语义安全对抗chosen-plaintext攻击)如果没有多项式时间的对手有可观的优势在修改游戏中i型和ⅱ型的对手,解密的甲骨文在阶段1和阶段2删除;2)对于i类和ii类对手，如果在修改后的博弈中没有多项式时间对手具有不可忽略的优势，且在设置前提交了挑战标识和时间周期，则采用外包撤销方案的IBE在选择模型中是安全的。

1. 有效的撤销外包IBE
2. **直觉**

为了实现有效的撤销,我们部分私钥更新的理念引入到提出建设、运营两个方面:1)我们利用混合为每个用户私钥在我们的系统中,有一个和门连接两个子组件即身份本土知识组件和组件TK分别的时间。IK是由key-update中的PKG生成的，TK是由key-update中新引入的KU-CSP更新的;2)在加密,我们采取作为输入用户身份标识以及时间限制解密,更准确地说,用户被允许执行成功解密当且仅当身份和时间段内嵌在他/她的私钥与密文是相同的。利用这种技术，我们可以通过KU-CSP更新私钥的时间组件来撤销用户的解密性。

**CCA Security Game for Type-I Adversary**

Setup:挑战者运行Setup(λ)获取密钥对(PK;MK)和输出PK

Phase 1:挑战者初始化一个空表列表L和一个空集s，对手提供以下的预言。

•私钥提取oracle。接收到ID后，运行KeyGen获取SKID = (IK[ID];TK[ID]Ti)和OKID。添加条目后(ID;打滑;输入L，输出IK[ID]。

•更新密钥提取oracle。在接收到ID和Tj时，如果存在(ID;打滑;在L中，集合S = S [f(ID;Tj) g。相应地，运行KeyUpdate并输出更新后的密钥tk [ID]Tj。

•解密oracle。接收到ID、Tj和CT后，运行解密并输出结果明文M。

挑战:对手输出两个等长的明文M0;M1, T\*和ID\*的限制(ID\*;T\*) 不属于S。  
挑战者随机选择一个比特b1属于(0,1)g和设置CT\*=加密(MbI;ID\*;T\*;PK)。

Phase 2:对手自适应问题更多的查询在阶段1的限制(ID\*;T\*)不能查询更新关键提取oracle。

猜测:最后，对手输出猜测bI’属于{0,1};如果bI’= bI，则1g获胜。

**CCA Security Game for Type-II Adversary**

设置:它与类型i对手的CCA安全游戏中的设置阶段相同。

阶段1:挑战者初始化一个空表列表L，两个空集U和i。外包密钥提取oracle。

挑战者在接收到ID后，运行KeyGen获取SKID = (IK[ID];TK [ID] Ti)和OKID。添加条目后(ID;打滑;输入L，输出OKID。私钥提取oracle。接收到ID后，如果存在条目(ID;打滑;在L中，设置U = U [fIDg]并将IK[ID]返回给对手。更新密钥提取oracle。在接收到ID和Tj时，如果存在条目(ID;打滑;在L中，检查ID是否为2u，如果没有设置I = I [fTjg]。然后，运行KeyUpdate并输出TK[ID]Tj。

解密oracle。它与CCA安全游戏中针对i类对手的解密oracle相同。

挑战:对手输出两个等长的明文M0;M1, T和ID的限制条件是T 2= I和ID 2= u挑战者随机选取一位bII 2 f0;1g，设置CT =加密(MbII;ID;T;PK)。

阶段2:对手自适应地发出更多的查询，就像阶段1一样，限制i) ID不能在私钥提取oracle中查询;(二)(ID;无法在更新的密钥提取oracle中查询。

猜测:最后，对手输出猜测b0 II 2f0;如果b0 = bII，则1g获胜。

Fig. 2. CCA Security Game for Type-I and Type-II Adversary

此外，我们注意到它不能简单地为所有用户使用相同的更新时间组件，因为被撤销的用户能够通过与未撤销的用户勾结来重新构建自己的能力。为了消除这种共谋，我们随机为每个标识ID生成一个外包密钥，该密钥本质上决定了两个子组件之间的匹配关系。此外，我们让KU-CSP维护一个列表UL来记录用户的身份及其对应的外包密钥。在密钥更新中，我们可以使用OKID更新用于标识ID的时间组件TK[ID]T。即使能够获得身份ID0的TK[ID0]Ti+1，被撤销的用户仍然无法解密在Ti+1下加密的密文。

**B .提出建模**

下面是我们提出的基于6点的建模。

• Setup(λ) : 设置算法由PKG运行，选择随机发生器和随机整数 Z q，设置。然后，PKG随机选取一个元素和两个哈希函数。最后输出公钥PK = (g,g1,g2,H1,H2)，主键MK = x。



• KeyGen(MK; ID; RL; T L; PK) : 对于每个用户对身份ID的私钥请求，PKG首先检查请求身份ID是否存在于RL中，如果存在，则终止密钥生成算法。接下来,包裹随机选择并设置它随机选择rID ，并计算 然后，PKG从T L中读取当前时间段Ti(如果T L为空，我们要求PKG首先创建当前时间段)。随机选取，计算其中， 。最后，输出和



• Encrypt(M; ID; Ti; PK)：假设用户希望在身份ID和时间段Ti下加密消息M。他/她选择一个随机值，计算出, ,, 。最后，将密文发布为CT =



• Decrypt(CT; SKID; PK) : 假设密文CT在ID和Ti下加密，用户有私钥SKID = ，其中和。然后计算：

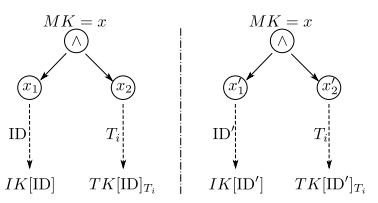
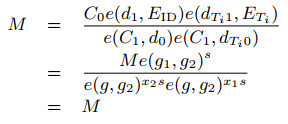


图3所示： 为两个不同用户生成私钥的比较

• Revoke: 如果用户的身份在集合中在Ti时间段内被撤销，PKG更新撤销列表为 通过将新创建的时间段Ti+1链接到原始列表T L上，最终将更新后的撤销列表以及新时间段Ti+1发送到KU-CSP的副本。



• KeyUpdate(RL; ID; Ti+1; OKID) : 当收到ID上的keyupdate请求时，KU-CSP首先检查撤销列表RL中是否存在ID，如果存在，判断KU-CSP是否返回，键更新被中止。否则，KU-CSP获取对应的条目。然后随机选取，计算, 。最后输出



最后，我们强调我们构造的思想是通过更新私钥中的时间组件来实现撤销。因此，关键是防止被撤销的用户与其他用户合谋重新构造其私钥。就像直觉中声明的那样，由于每个用户在x上的随机分割，这种共谋攻击在我们提出的构造中是具有抵抗性的。具体来说,如图3所示^是一个和门连接两个子组件,如果两个不同的用户呼吁他们的私钥,包裹将获得两个随机分裂(x1;x2)和;x1+x2 = x mod q, = x mod q. x1和分别产生ID和的单位分量，x2和分别产生时间分量。由于x1和x2以及和之间存在补码，因此单位分量和时间分量应该相应地在私钥中进行验证。通过这样的验证，即使好奇的用户获得了其他用户的时间组件，他/她也无法伪造一个有效的私钥来成功执行解密。



***C. 关键服务程序***

根据我们的算法构建，如图4所示，在提出的IBE方案中，外包撤销工作的密钥发布、密钥更新和撤销等密钥服务流程如下:

• Key-issuing.我们要求PKG在本地维护一个撤销列表RL和一个时间列表T L。接收到ID上的私钥请求后，PKG运行KeyGen(MK;ID;RL;T L;PK)获取私钥SKID和外包密钥OKID。最后，将SKID发送给user和(ID;OKID)分别到KUCSP。如直觉所述，对于每个条目从PKG发送，KU-CSP应该将其添加到本地维护的用户列表UL中。



• Key-update.如果某些用户在Ti时间段内被撤销，则每个未撤销的用户需要向KU-CSP发送key-update请求，以保持解密性。KU-CSP接收到身份ID请求后，运行Key-Update(RL;ID;Ti + 1,OKid);获取TK[ID]Ti+1。最后，将该时间组件发送回能够更新其私钥的用户SKID = (IK[ID];TK (ID)Ti + 1)。

• Revocation.与密钥更新类似，如果被撤销的用户发送一个关于标识ID的密钥更新请求，KU-CSP将运行KeyUpdate(RL;ID;Ti + 1;OKID)。然而，由于KU-CSP将返回。因此，这样的键更新请求将被中止。



***D. 安全分析***

定理1:假设−DBDH假设在G和哈希函数中H1和H2是随机的预测。假设对手最多制造出和查询哈希函数H1,H2、私钥、更新密钥和外包密钥提取oracle。我们用表示G中单次基于多指数运算的时间成本，那么提出的外包撤销方案IBE为感觉很安全的IND-ID-CPA



证明:假设对手AI和AII在攻击I类和II类对手的IND-ID-CPA安全意义上提出的IBE方案时分别具有I和II优势。我们将构建两个模拟器SI和SII，分别使用AI和AII作为子算法，以不可忽略的概率解决BDH决策问题。

假设DBDH问题中的挑战者在SI和SII的视图之外抛了一个均匀的二元硬币。如果 µ= 0，则给出SI和SII ;否则， 表示随机。要求SI和SII输出一个值0作为猜测值。然后我们提供如下模拟。

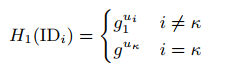


**针对I型对手的SI仿真**

设置:SI设置g1 = X;g2 = Y，发送公钥PK = (g;g1;g2)AI。

阶段1:SI初始化一个空的表列表L，一个空的集S. AI可以发出以下类型的查询。

• H1-query. 如果随机挑选并维护一个列表L1来存储散列oracle H1的答案。当收到IDi, SI对L1进行检查。如果找到查询的条目，将返回相同的答案。否则，SI随机选择进行计算



在L1中存储条目后，SI返回。



• H2-query.如果随机挑选并维护一个列表L2来存储散列oracle H2的答案。在接收到，SI对L2执行检查。如果找到查询的条目，将返回相同的答案。否则，SI随机选择进行计算



在L2中存储条目后，SI返回。

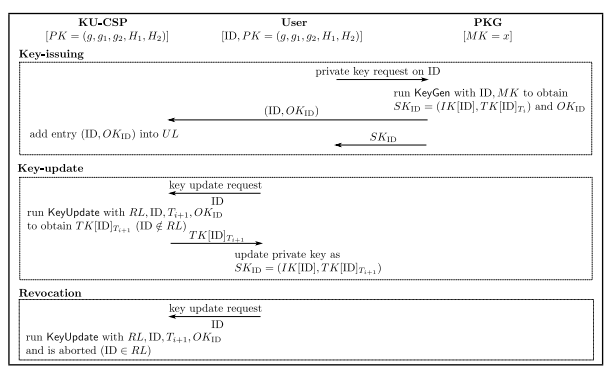


图4所示。密钥分发、密钥更新和撤销协议

• Private key query. 在接收到IDi时，SI以以下两种方式之一进行响应。

如果如果随机选择并试图模拟通过设置其中 。因此，其中， 。此外，SI设置OKIDi = x2并将其发送回SI。



如果如果随机选择和计算 其中和 。此外,S1设置。



添加条目输入进L后，, SI返回。



• Updated key query.在接收，SI检查是否存在一个条目在L中，如果没有，SI中止;否则，SI获取这样的条目并在以下两种情况下作出响应



SI选择，设置返回其中d， 。



如果检查是否j =η:如果是这样,中止。否则,设置随机之后设置返回其中, 。



挑战:AI将提交两条挑战消息M0和M1以及和 . SI检查标识是否,如果是安全游戏中止。否则,如果硬币翻转一个公平的二进制ν g和返回一个加密。密文是模拟 。我们注意到如果,如果我们让s = z,然后;;



**阶段2:阶段1重复。**

猜想:AI将提交一个猜想V’。如果SI输出,否则输出。



我们注意到,从的可能性提交挑战,安全完成游戏成功的概率。因此,我们有。因为猜测 = 1



当。如果µ= 0,那么看到一个加密的成功的游戏。因此,我们最后，我们在解决DBDH问题上具有SI的整体优势即



**模拟SII对抗II型对手**

**设置:**SII的执行与SI中的相同。

**阶段1**:SII初始化一个空表列表L和两个空集U和I。 允许U和i发出以下类型的查询。



*• H*1*-query.*SII的响应与SI中的相同。

*• H*2*-query.*SII的响应与SI中的相同。

*• Outsourcing key query.*SII接收到IDi后，随机选择，加入到L。



*• Private key query.*在收到IDi时，如果存在这样的条目L,检查i=κ,如果是中止。否则，SII设置其中，计算其中。设置U = ，SII返回。

