软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2021,32(11):3606−3627 [doi: 10.13328/j.cnki.jos.006034]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

应用区块链的多接收者多消息签密方案*

王利朋[1,2], 高健博[1], 李青山[1], 陈 钟[1]

[1](北京大学 信息科学技术学院,北京 100871)

[2](郑州师范学院 信息科学与技术学院,河南 郑州 450044)

通讯作者: 陈钟, E-mail: zhongchen@pku.edu.cn

摘 要: 信息通过公共链路进行传输时极易遭受窃听、篡改等形式的网络攻击,因此有必要保障信息在传输过程中的机密性和完整性,而签密技术能够有效地实现上述目的.基于椭圆曲线,提出一种多接收者多消息签密方案,能够有效地适配到广播系统中.采用多密钥分发中心管理系统主密钥信息,且能够周期地更新各自的秘密信息,以抵抗对应的 APT 攻击.不同更新周期注册的用户相互之间能够通信,不会影响系统的可用性.提出了一种基于区块链的周期更新策略,根据公有链中区块高度和时间戳触发密钥更新动作,基于区块链不可篡改特性确保方案的安全性,且该过程不需要执行交易动作,因此是免费的.基于 Computational Diffie-Hellman 问题和离散对数问题,在随机预言机模型下证明了签密方案的机密性和不可伪造性,该方案同时具有密钥托管安全性、前后向兼容性、不可否认性.性能分析表明,该签密方案具有较短的密文长度和较高的执行效率.在实验仿真部分,首先分析了密钥分发中心数量和门限值对签密算法性能的影响,在排除网络延迟等因素干扰下,引入多密钥分发中心后,性能损耗在 5%以内;其次,基于区块链实现周期更新时的时间误差百分比会随周期的增加而下降,当周期大于 550s 时,其值控制在 1%以内.这种误差使得攻击者很难预测更新的准确时间,增大了攻击的难度.

关键词: 密钥生成中心;区块链;签密;机密性;离散对数

中图法分类号: TP309

中文引用格式: 王利朋,高健博,李青山,陈钟.应用区块链的多接收者多消息签密方案.软件学报,2021,32(11):3606−3627. http://www.jos.org.cn/1000-9825/6034.htm

英文引用格式: Wang LP, Gao JB, Li QS, Chen Z. Blockchain-based multi-recipient multi-message signcryption scheme. Ruan Jian Xue Bao/Journal of Software, 2021,32(11):3606−3627 (in Chinese). http://www.jos.org.cn/1000-9825/6034.htm

Blockchain-based Multi-recipient Multi-message Signcryption Scheme

WANG Li-Peng[1,2], GAO Jian-Bo[1], LI Qing-Shan[1], CHEN Zhong[1]

[1](School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China)

[2](College of Information Science and Technology, Zhengzhou Normal University, Zhengzhou 450044, China)

Abstract: When data is transmitted through the network, it is vulnerable to network attacks such as eavesdropping and tampering. Therefore, data confidentiality and data integrity should be guaranteed which can be achieved with the signcryption schemes. Based on the elliptic curve, a multi-receiver multi-message signcryption scheme is proposed, which can be effectively adapted to many scenarios such as broadcast systems. Multiple key distribution centers are used to manage the system master key, and the secrets of each center can be updated periodically to resist the APT attacks. In addition, users registered in different periods can communicate with each other to improve the availability. A secret update strategy based on the public blockchain is proposed, and the update operation is triggered based

* 基金项目: 国家重点研发计划(2020YFB1005404); 河南省科技攻关计划(202102210359); 河南省高等学校重点科研项目(22A520048, 20B520040)

Foundation item: National Key Research and Development Program of China (2020YFB1005404); Science and Technology Program of Henan Province (202102210359); Henan Province Higher Education Key Research Project (22A520048, 20B520040)

收稿时间: 2019-11-29; 修改时间: 2020-01-20, 2020-02-23, 2020-03-14; 采用时间: 2020-03-21

on the block height and the block timestamp. Blockchain, with its non-tampering feature, can guarantee security of the proposed scheme. In addition, the new scheme does not need to send transactions and is therefore free. Based on the computational Diffie-Hellman problem and the discrete logarithm problem, confidentiality and unforgeability of the proposed scheme are analyzed on the random oracle model. The proposed scheme also has the following security attributes: key escrow security, forward and backward compatibility, and non-repudiation. Performance analysis shows that the proposed scheme has a shorter ciphertext length and higher efficiency. In the simulation part, influence of the number of key distribution centers and the threshold on the system performance is analyzed. Without considering the network delay and other disturbing factors, the performance loss is less than 5% for the proposed scheme compared with those with a single key distribution center. The time errors incurred by the update strategy based on blockchain decrease with the increasing periods. When the period is set more than 550s, the time error percentage is less than 1%. The time errors make it more difficult for the attackers to predict the update time and launch the attacks.

Key words: key generation center; blockchain; signcryption; confidentiality; discrete logarithm

1 引 言

1.1 研究动机

信息时代中,数据安全问题变得愈加重要.为了实现数据的机密性和完整性,在数据传输过程中,发送方首先要对消息内容进行签名,再执行加密流程,然后将密文信息发送至接收方.接收方收到密文信息后,首先执行解密流程,然后执行签名校验过程,以确保数据在传输过程中未被篡改.这种加(解)密技术和数字签名技术相分离的操作,存在执行效率低的问题,因此在 1997 年,文献[1]首次提出了签密概念,能够在一个逻辑步骤里同时实现签名和加(解)密步骤,能够有效提升算法的执行效率.

当前,大部分签密方案需要密钥分发中心(key generation center,简称 KGC)协助生成系统主密钥信息,并基于此生成用户的公私钥信息.然而在实际场景中,KGC 容易成为被攻击的对象,攻击者尝试发起对 KGC 的持续性攻击,一旦攻击成功,即可获取系统主密钥信息,进而威胁签密系统的安全性.为了解决上述问题,可以采用多个KGC通过门限秘密分享协议协同管理系统主密钥信息,即使攻击者成功获得了低于一定数量的KGC上存储的子秘密信息,仍无法计算得到系统主密钥.然而在实际中,攻击者如果攻破足够数量的密钥服务器,同样能够获知系统主密钥信息,因此对于长期运行的签密系统,仅仅靠门限秘密分享协议仍无法保障系统的安全性[2].因此,如果能够研究一种动态更新的多 KGC 签密系统,使得各 KGC 按照一定周期更新其存储的子秘密信息,是一个重要的研究方向.基于秘密共享技术实现签密场景中系统主密钥管理以及各 KGC 子秘密同步更新方案,需要重点解决以下问题.

(1) KGC密钥服务器会按照一定周期T动态更新自己的子秘密信息,用户可能会在不同周期与KGC交互生成自己的公私钥信息,如何使得这些用户之间同样能够正常执行签密流程,是首先要解决的问题.

(2) KGC密钥服务器之间需要依赖一种策略同步更新各自的子秘密信息,实际中,第三方可以攻击该更新策略,进而影响系统可用性.如何保证上述周期更新策略的安全性,是另一个需要解决的问题.

(3) KGC 密钥服务器需要根据约定周期执行更新过程,由于通信延迟的存在,如何保证各 KGC 更新子秘密信息的及时性,是需要重点解决的问题.

本文提出了一种多密钥分发中心多消息多接收者签密方案,并基于区块链技术实现密钥分发中心子秘密同步更新策略,具体贡献包括以下内容.

(1) 基于椭圆曲线提出一种多消息多接收者签密方案,采用多KGC通过门限秘密分享协议实现系统主密钥管理,然后周期更新各 KGC 子秘密信息,以抵抗针对其的持续性攻击.接收者收到密文信息后,会首先执行身份校验过程,如果校验失败,无需执行后续解密流程,进而减少接收者的计算负担.在随机预言机模型下,基于离散对数困难问题和 CDH 困难问题,证明了方案的机密性和不可伪造性.证明了方案的前后向兼容性,即不同周期内注册的用户相互之间仍然能够相互通信.最后证明了本方案的密钥托管安全性和不可否认性.

(2) 提出一种基于区块链的多密钥分发中心子秘密周期更新策略,根据公有链中块高度和块时间戳来触发密钥更新操作.新方案由区块不可篡改特性保证了周期更新过程的可信性.另外,该方案不需要执行交易过程,因此是免费的.在更新过程中,各密钥分发中心无需执行节点间通信过程,减少了带宽资源的消耗.

(3) 性能分析表明:本文提出的签密方案在满足同样安全属性要求的前提下,与现有方案相比具有较短的密文长度和较高的执行效率.签密方案仿真实验表明:在排除网络延迟等因素干扰下,引入多密钥分发中心后,性能损耗在 5%以内.基于区块链的密钥更新方案仿真实验表明:周期更新操作会有较小的时间误差,在周期为 550s 后,误差百分比下降到 1%以内;同时,随着周期增加,其对应的误差百分比仍会逐渐下降.最后,这种随机误差会增加执行密钥更新过程的不确定性,使攻击者无法准确预知子秘密更新时间,增大攻击者攻击 KGC 的难度.

1.2 相关工作

随着广播通信技术的发展,发送者需要向多个用户发送信息,并希望只有授权用户才能接收消息,而且希望消息在不可信的通信信道中传输时,第三方不能获知消息内容.近年来,研究人员提出了多接收者签密模型[3],用来解决上述问题.根据依赖的问题难度,传统的签密方案主要分为基于离散对数的签密方案、基于椭圆曲线离散对数的签密方案、基于双线性对的签密方案.一般来讲,这 3 种类型的签密方案,其对应的攻击难度依次上升.当前出现了具有抗量子攻击的签密体制,分别是基于哈希的密码体制、基于编码的密码体制、基于格的密码体制以及基于多变量的密码体制.近年来,研究人员基于这 4 种抗量子攻击的密码体制分别提出了相应的签密方案,极大地提升了签密算法的安全性.

当前出现了基于身份的多接收者签密方案,用户可以利用其身份信息作为公钥参与计算,简化了密钥管理问题.文献[4]提出了一种基于身份的多接收者签密方案,该方案只需要执行一次双线性对操作,即可实现签密功能.最后,基于随机预言机模型给出了机密性和不可篡改性证明.该方案的安全性是基于 Bilinear Diffie-Hellman (BDH)问题以及 Computational Diffie-Hellman(CDH)问题.文献[5]提出了一种基于双线性对的多接收者签密方案,该方案基于 Gap Diffie-Hellman 问题,并具有较高的计算效率和较低的通信延迟.文献[6–8]分别提出了相应的基于身份的多接收者签密机制,并给出了相应的安全证明.

目前出现了对接收者的身份信息进行校验的签密算法,在这些方案中,只有授权接收者才能正确执行解签密操作.文献[9]针对现有签密方案中接收者身份信息泄露以及签密不公平的问题,提出了一种具有解签密公平性的多接收者签密方案.该方案在解签密时没有校验密文合法性,而且没有实现发送者身份匿名的可控性,仍有较大的改进空间.文献[10]提出了一种满足密钥托管安全性的签密方案,新方案基于椭圆曲线离散对数问题(elliptic curve discrete logarithm problem,简称 ECDLP)和 CDH 问题,并对该方案的保密性和接收者匿名性进行了证明.文献[11]将接收者所需的身份信息与密文信息揉合一起,并基于密文信息实现公开可验证性.

当前已有的签密方案中,需要 KGC 生成系统主密钥信息,然后用户与之进行交互生成自身对应的公私钥信息.由前面论述可知,可以采用多个 KGC 通过门限秘密分享协议协同管理系统主密钥信息.即使攻击者攻破一定数量的 KGC,仍无法求解出系统主密钥信息.当前已有的秘密共享协议主要有 Shamir 秘密共享技术以及基于中国孙子定理的秘密共享技术,这些秘密共享技术被广泛应用于分散风险和容忍入侵等场景中.

相关研究人员将 Shamir 秘密共享技术应用于数据安全领域中.Tzer-Shyong 等人将椭圆曲线加密所需较短的密钥特征与(t,n)门限方法集成,提出了一种校验数据完整性的群签名方案[12].文献[13]提出了一种适用于云存储环境中关键字加密搜索算法,该方案基于 Shamir 算法实现密钥服务器的主密钥管理,能够有效抵御关键字猜测攻击.文献[14]提出了一种数据完整性校验方案,能够有效抵御 t 个成员的合谋攻击.

近些年来,出现了基于中国孙子定理的秘密分享方案.Asmuth 和 Bloom 提出了经典的 Asmuth-Bloom 门限秘密共享方案[15],与 Shamir 秘密共享技术相比,计算量较小.文献[16]提出了一种将 ElGamal 机制与 Asmuth-Bloom 门限秘密共享相结合的方案,能够防止秘密份额在传播过程中被篡改.文献[17]的方案能够有效地控制计算过程中的数据长度,具有良好的匿名性和防伪造性,然而必须依赖可信中心进行密钥分发.

除了上述两种秘密共享技术,当前出现了其他一些秘密共享方案:文献[18]基于双线性映射和秘密共享思想提出了一种基于身份的门限签名方案,采用基于身份的 t-out-of-n 秘密共享算法提升了算法的执行效率;文献[19]提出了一种离散对数难度的门限方案,能够有效抵抗针对秘密共享技术的攻击手段;文献[20,21]提出了基于 ECDSA 的门限签密技术,s 个参与者重构密钥,但却需要 $2s$+1 个参与者才能签名;Goldfeder 等人[22]提出利用秘密分享技术实现比特币密钥的多方控制功能,利用门限密码学技术实现密钥的可信管理.

2 预备知识

2.1 困难性问题

- 离散对数(discrete logarithm,简称 DL)问题

设定循环群 G_p 的阶为 p,且 p 是大素数,P 为 G_p 的生成元,$a \in Z_p^*$,给定元素组$\langle P,aP\rangle$,求解 a.给定算法λ,能够在概率多项式时间内解决 DL 问题的概率 $Adv^{DL}(\lambda)=\Pr\lceil \lambda(P,aP)=a\rceil$是可忽略的.

- Computational Diffie-Hellman(CDH)问题

设定循环群 G_p 的阶为 p,且 p 是大素数,P 为 G_p 的生成元,$a,b \in Z_p^*$,给定元素组$\langle P,aP,bP\rangle$,求解 abP.给定算法λ,能够在概率多项式时间内解决 CDH 问题的概率 $Adv^{CDH}(\lambda)=\Pr\lceil \lambda(P,aP,bP)=abP\rceil$是可忽略的.

2.2 系统模型

多接收者多消息签密方案由下述步骤组成.

(1) 系统初始化(Setup):输入参数 k,密钥生成中心协作生成系统公开参数 $Params$ 和主密钥 s,定义为 $Setup(k)\rightarrow(Params,s)$.

(2) 用户密钥生成(KeyGen):输入用户身份信息 ID、系统公开参数 $Params$ 以及系统主密钥 s,用户和密钥生成中心协作,生成用户私钥γ和用户公钥κ,定义为 $KeyGen(ID,Params,s)\rightarrow(\gamma,\kappa)$.

(3) 签密(SignCrypt):对于ρ个接收者,发送者 ID_S 发送含有ρ条消息的明文集合 M.输入系统公开参数 $Params$、明文集合 $M=\{m_1,m_1,\ldots,m_\rho\}$、发送者身份 ID_S 以及接收者身份集合 $ID_R=\{ID_{R_1},ID_{R_2},...,ID_{R_\rho}\}$,输出相应的密文集合 $C=\{c_{R_1},c_{R_2},...,c_{R_\rho}\}$, 定义为 $SignCrypt(Params,M,ID_S,ID_R)\rightarrow C$.

(4) 解签密(UnSignCrypt):接收者 $ID_{R_i}(i\in[1,\rho])$ 根据系统公开参数 $Params$、密文 $c_{R_i}(c_{R_i}\in C,i\in[1,\rho])$、接收者私钥 γ_{R_i},输出明文信息 $m'_{R_i}(i\in[1,\rho])$, 定义为 $UnSignCrypt(Params,c_{R_i},ID_{R_i},\gamma_{R_i})\rightarrow m'_{R_i}$.

(5) 签名校验(VerifySign):接收者 ID_{R_i} 根据输入参数 $Params$ 以及接收者公钥 κ_{R_i},对解签密后的消息 m'_{R_i} 进行校验:如果校验通过,输出 True;否则,输出 False:定义为

$$VerifySign(Params,m'_{R_i},ID_{R_i},\kappa_{R_i})\rightarrow(\text{True}\,|\,\text{False}).$$

2.3 安全模型

按照文献[23]定义的安全模型,本节将介绍选择密文攻击下的保密性和选择消息攻击下的不可伪造性的安全属性定义以及对应的游戏交互模型.设定接收者共有ρ个,标记为 $ID_R=\{ID_{R_1},ID_{R_2},...,ID_{R_\rho}\}$, 明文消息集合为 $M=\{m_1,m_1,\ldots,m_\rho\}$.不失一般性,下面主要描述第 i 个接收者 ID_{R_i} 处理消息 m_i 的步骤.设定选择密文攻击保密性的攻击敌手为λ^1,选择消息攻击不可伪造性的攻击敌手为λ^2,此外还需要考虑其他两种类型的攻击,分别如下.

- 第 1 种类型的攻击敌手并没有掌握系统主密钥信息,但敌手具有能够替换用户公钥的能力,这种攻击方式的敌手标记为λ_{I}.敌手λ_{I} 在对保密性进行攻击时重新标记为 λ_{I}^1,对不可伪造性进行攻击时重新标记为 λ_{I}^2.
- 第 2 种类型的攻击敌手能够获取系统主密钥信息,但不具有替换公钥的能力,这种攻击方式的敌手标记为λ_{II}.敌手λ_{II} 在对保密性进行攻击时重新标记为 λ_{II}^1,对不可伪造性进行攻击时重新标记为 λ_{II}^2.

定义 1(攻击敌手为 λ_{I}^1 时方案的保密性). 若不存在敌手 λ_{I}^1 能够通过多项式时间的计算以不可忽略的优势

$Adv^c(k)$赢得以下的游戏,则称方案具有多接收者多消息签密机制下选择密文攻击的保密性.

1. 输入参数 k,挑战者$\mathfrak{I}$执行 Setup 步骤,生成主密钥 s 和系统公开参数 $Params$,挑战者$\mathfrak{I}$将 $Params$ 发送给敌手 $\mathfrak{A}_{\mathrm{I}}^1$,同时$\mathfrak{I}$保存主秘密 s.
2. 敌手 $\mathfrak{A}_{\mathrm{I}}^1$ 向挑战者$\mathfrak{I}$执行如下询问.
   - 用户密钥生成询问:敌手 $\mathfrak{A}_{\mathrm{I}}^1$ 选择一个身份 ID,并发送至挑战者$\mathfrak{I}$,$\mathfrak{I}$执行 $KeyGen(ID,Params,s)\rightarrow(\gamma,\kappa)$,并返回$(\gamma,\kappa)$给敌手$\mathfrak{A}_1$.在后续证明过程中,该步骤可细分为用户公钥生成询问和用户私钥生成询问,分别返回用户 ID 对应的公钥信息κ和私钥信息γ.
   - 签密询问:发送者的身份为 ID_S,敌手 $\mathfrak{A}_{\mathrm{I}}^1$ 将明文消息 M 和接收者身份集合 $ID_R=\{ID_{R_1},ID_{R_2},...,ID_{R_\rho}\}$ 发送至挑战者$\mathfrak{I}$.挑战者$\mathfrak{I}$执行 $SignCrypt(Params,M,ID_S,ID_R)\rightarrow C$,并将密文信息 C 发送至敌手 $\mathfrak{A}_{\mathrm{I}}^1$.
   - 解签密询问:挑战者$\mathfrak{I}$收到发送者身份信息 ID_S、接收者身份信息 ID_{R_i} 以及密文 c_{R_i}($c_{R_i}\in C$, $i\in[1,\rho]$),挑战者$\mathfrak{I}$执行对接收者 ID_{R_i} 的密钥生成询问,并得到其对应的私钥信息 γ_{R_i} 和公钥 κ_{R_i},然后执行 $UnSignCrypt(Params,c_{R_i},ID_{R_i},\gamma_{R_i})\rightarrow m'_{R_i}$,$VerifySign(Params,m'_{R_i},ID_{R_i},\kappa_{R_i})\rightarrow(\mathrm{True}\,|\,\mathrm{False})$,如果校验结果为 True,则返回 m'_{R_i} 给敌手 $\mathfrak{A}_{\mathrm{I}}^1$;否则,返回$\perp$给敌手 $\mathfrak{A}_{\mathrm{I}}^1$.
   - 公钥替换询问: $\mathfrak{A}_{\mathrm{I}}^1$ 可以在任何时间点替换用户 ID 的公钥κ.
3. 在挑战阶段,敌手 $\mathfrak{A}_{\mathrm{I}}^1$ 选择两条消息 m_0 和 m_1,以及发送者身份信息 ID_S、接收者身份信息 ID_R,其中,两条消息长度相同,并将上述信息发送至挑战者$\mathfrak{I}$,挑战者$\mathfrak{I}$随机选择 $i\in\{0,1\}$,并执行 $SignCrypt(Params,m_i,ID_S,ID_R)\rightarrow C$,并将 C 返回给敌手 $\mathfrak{A}_{\mathrm{I}}^1$.
4. 在猜测阶段,敌手 $\mathfrak{A}_{\mathrm{I}}^1$ 可以执行多次步骤 2 中的询问操作,但是不能对 ID_R 执行私钥生成询问操作,也不能够对 C 中的密文执行解签密询问.
5. 敌手 $\mathfrak{A}_{\mathrm{I}}^1$ 输出对随机数 i 的猜测值 i',如果 $i=i'$,则 $\mathfrak{A}_{\mathrm{I}}^1$ 赢得本次游戏,并设定敌手 $\mathfrak{A}_{\mathrm{I}}^1$ 赢得上述游戏的优势为 $Adv^c(k)=\left|P[i=i']-\frac{1}{2}\right|$.

定义 2(攻击敌手为 $\mathfrak{A}_{\mathrm{I}}^2$ 下方案的不可伪造性). 若不存在敌手 $\mathfrak{A}_{\mathrm{I}}^2$ 能够通过多项式时间的计算以不可忽略优势 $Adv^u(k)$赢得以下的游戏,则称方案具有多接收者多消息签密机制下选择消息攻击下的不可伪造性:

输入参数 k,挑战者$\mathfrak{I}$执行初始化步骤,并发送公共参数给 $\mathfrak{A}_{\mathrm{I}}^2$.询问阶段,攻击敌手 $\mathfrak{A}_{\mathrm{I}}^2$ 选定身份 ID,其余步骤与定义 1 中的第 2 步相同.在伪造阶段,敌手 $\mathfrak{A}_{\mathrm{I}}^2$ 伪造密文 C',然后执行询问步骤,但是不能对接收者询问私钥信息,也不能对密文执行签密和解签密询问操作.如果解密出的消息进行校验的结果为 True,则敌手 $\mathfrak{A}_{\mathrm{I}}^2$ 赢得本次游戏.

定义 3(攻击敌手为 $\mathfrak{A}_{\mathrm{II}}^1$ 下方案的保密性). 若不存在敌手 $\mathfrak{A}_{\mathrm{II}}^1$ 能够通过多项式时间的计算以不可忽略优势 $Adv^c(k)$赢得以下的游戏,则称方案具有多接收者多消息签密机制下选择密文攻击的保密性.

输入参数 k,挑战者$\mathfrak{I}$执行初始化步骤,并将主密钥 s 和系统公开参数 $Params$ 发送至敌手 $\mathfrak{A}_{\mathrm{II}}^1$.在询问阶段,攻击敌手 $\mathfrak{A}_{\mathrm{II}}^1$ 选定身份 ID,敌手 $\mathfrak{A}_{\mathrm{II}}^1$ 可执行定义 1 中除公钥替换外的其他询问操作.在挑战阶段和猜测阶段,敌手 $\mathfrak{A}_{\mathrm{II}}^1$ 和挑战者$\mathfrak{I}$执行的步骤同定义 1 相同.最后,敌手 $\mathfrak{A}_{\mathrm{II}}^1$ 输出对 j 的猜测值 j',如果 $j=j'$,则敌手 $\mathfrak{A}_{\mathrm{II}}^1$ 赢得本次游戏.

定义 4(攻击方式为 $\mathfrak{A}_{\mathrm{II}}^2$ 下方案的不可伪造性). 若不存在敌手 $\mathfrak{A}_{\mathrm{II}}^2$ 能够通过多项式时间的计算以不可忽略优势 $Adv^u(k)$赢得以下的游戏,则称该方案具有多接收者多消息签密机制下适应性选择消息攻击下的不可伪造性.

在初始化阶段,挑战者$\mathfrak{I}$执行的步骤同定义 3 的初始化过程相同.在询问阶段,敌手 $\mathfrak{A}_{\mathrm{II}}^2$ 选定身份 ID,执行的步骤同定义 3 中的第 2 步相同.在伪造阶段,给定伪造密文 C',执行步骤同定义 2 中的伪造阶段相同.如果解密出的消息进行校验结果的为 True,则敌手 $\mathfrak{A}_{\mathrm{II}}^2$ 赢得本次游戏.

3 方案步骤

3.1 系统架构

本文方案的系统架构图如图 1 所示,方案中共有 3 种角色,分别是 KGC、发送方(sender)和接收方(receiver).首先,KGC 执行初始化操作,生成系统主密钥等信息,然后负责为发送方和接收方生成公私钥信息,但 KGC 并不负责托管公私钥信息.发送方接收到明文信息后,对其执行签密操作,并将密文信息发送至接收方.接收方收到密文信息后,执行解密操作得到明文信息,然后利用发送方和接收方的公钥等信息对解密后的信息进行校验.KGC 由一组节点构成,能够周期更新各自的子秘密信息,以抵抗对应的 APT 攻击.

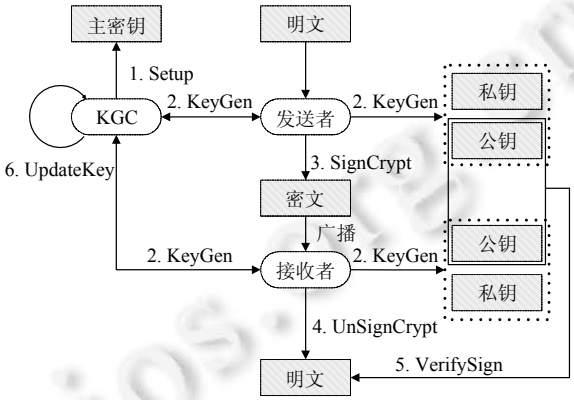

Fig.1 System framework

图 1 系统架构

为了更好地理解本文提出的方案,下面表格定义了本文算法所采用的符号,见表 1.

Table 1 Notations of the proposed scheme

表 1 本文方案的符号表示

符号	含义	符号	含义
KGC	密钥分发中心	P	G 的生成元
KS_i	第 i 个密钥服务器	n	密钥服务器总个数
ID	用户身份信息	t	合成系统主密钥所需最少密钥服务器个数
k	系统安全参数	m	明文信息
γ	用户私钥信息	L_m	明文 m 的长度
κ	用户公钥信息	L_I	用户 ID 的长度
p	大素数	ρ	用户的个数
G	加法循环群	s	系统主私钥
$Params$	系统公开参数	P_{pub}	系统主公钥
s_i	KS_i 的私钥信息	k_i	KS_i 的公钥信息
M	待签密消息集合	ID_R	接收者的身份集合

3.2 系统实现

3.2.1 初始化(Setup)

KGC 负责执行本步骤,在本方案中,KGC 由多个密钥服务器构成.设定密钥服务器集合为$\{KS_1,KS_2,\ldots,KS_n\}$,共有 n 个密钥服务器,消息发送者只需要收到 t 个服务器的正确应答消息,即可计算得到系统主密钥信息.设定循环群 G 的阶数为 p,生成元为 P,其中,p 为大素数.

定义抗碰撞密码学单向哈希函数为

$$H_1:\{0,1\}^{L_I}\times G\to Z_p^*,H_2:\{0,1\}^{L_I}\times\{0,1\}^{L_I}\times\{0,1\}^{L_m+|Z_p^*|}\times G\to Z_p^*,H_3:\{0,1\}^{L_I}\times G\to\{0,1\}^{L_m+|Z_p^*|},$$

其中,L_l为用户 ID 的长度,L_m为明文 m 的长度,$|Z_p^*|$为 Z_p^*中整数的长度.定义索引函数 $F_{index}^{\rho}(ID)\to Z_p^*$,对于$\rho$个用户,$ID\in\{ID_1,ID_1,\ldots,ID_\rho\}$.该函数是将用户 ID 均匀映射到$\{1,2,\ldots,\rho\}$,以用于消息发送者和接收者从消息集合中定位自己的位置坐标.定义⊕为$\{0,1\}^*$的异或运算,对于$\forall M,N\in\{0,1\}^*$,满足 $M\oplus N\oplus M=N$.

对于 $i=\{1,2,\ldots,n\}$个服务器 KS_i,输入系统参数 k,各个 KS_i随机生成一个 $t-1$ 次的多项式 $g_i(x)=a_{i,0}+a_{i,1}x+\ldots+a_{i,t-1}x^{t-1}$,其中,$a_{i,j}\in Z_p^*(0\leqslant j\leqslant t-1)$.当 $x=0$ 时,可以得到 $g_i(0)=a_{i,0}$,其中,$a_{i,0}$为 KS_i的子秘密.KS_i计算$\{a_{i,\xi}P|0\leqslant\xi\leqslant t-1\}$,并将其发送至其他 $n-1$ 个 $KS_j(j\neq i)$服务器.然后,KS_i计算$\{g_i(j)|1\leqslant j\leqslant n\}$,并将每个 $g_i(j)$通过秘密渠道依次发送至对应的$KS_j(j\neq i)$服务器.收到上述$g_i(j)$消息后,KS_j对其进行校验,校验公式为 $g_i(j)P=\sum_{\xi=0}^{t-1}(j^{\xi}(a_{i,\xi}P))$.如果校验失败,则拒绝接收该项数据.$KS_j$对接收到的 $n-1$ 项 $g_i(j)$全部校验成功后,加上在本地计算得到的 $g_i(i)$消息,此时,KS_j得到元素个数为 n 的$\{g_i(j)|1\leqslant i\leqslant n\}$.

KS_j的私钥为 $s_j=\sum_{\xi=1}^{n}g_\xi(j)$,其对应的公钥为 $k_j=s_jP$.系统主私钥为 $s=\sum_{\xi=1}^{n}a_{\xi,0}$,系统主公钥为 $P_{pub}=\sum_{\xi=1}^{n}a_{\xi,0}P$,并公开参数 $Params=\{G,p,k_j,P,P_{pub},H_1,H_2,H_3,F_{index}^{\rho},\oplus,\|\}$.初始化流程如图 2 所示.

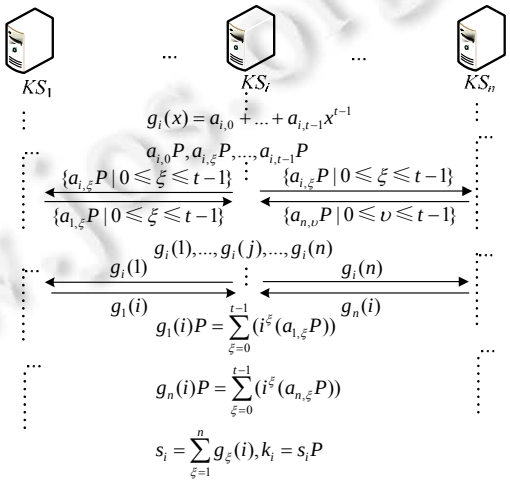

Fig.2 System setup

图 2 系统初始化

3.2.2 用户密钥生成(KeyGen)

用户 ID 随机选择 $u_{id}\in Z_p^*$,并计算得到 $U_{id}=u_{id}P$,然后将 U_{id}公布出去;然后,用户将自己的 ID 信息发送至 n 个密钥服务器.当KS_i收到用户的 ID 信息后,首先要对用户身份信息按照一定准入原则进行判定,才允许其加入,这里不再对准入原则进行赘述.然后,KS_i计算得到 $h_1^{id}=H_1(ID,U_{id})$,并得到 $w_i=s_ih_1^{id}$,并将 w_i通过秘密通道发送给用户 ID.

用户 ID 收到 w_i后,对其进行校验,校验公式为 $w_iP=k_ih_1^{id}$.校验通过后,用户接收 w_i.当用户收到 t 份 w_i后,构成集合 $W=\{w_{i_1},w_{i_2},\ldots,w_{i_t}\}$,其中,索引集合为 $I=\{i_1,i_2,\ldots,i_t\}$.

用户进而计算得到 $\gamma=\sum_{\xi=i_1}^{i_t}w_\xi\delta_\xi+u_{id}$,其中,$\delta_\xi=\prod_{\substack{i_1\leqslant j\leqslant i_t\\ j\neq\xi,\,j\in I}}\frac{j}{j-\xi}$.用户私钥信息为$\gamma$,用户公钥信息为$\kappa=U_{id}$.

3.2.3 签密(SignCrypt)

设定接收者共有ρ个,待签密消息集合为 $M=\{m_1,m_1,\ldots,m_\rho\}$,接收者身份集合为 $ID_R=\{ID_{R_1},ID_{R_2},\ldots,ID_{R_\rho}\}$,发送者的身份为 ID_S.首先,发送者 ID_S随机选择 $\alpha_s\in Z_p^*$,并计算得到 $A_s=\alpha_sP$,α_s用于保证对于相同的消息每次生成

的密文信息均不相同.依次对每一个接收者 $ID_{R_i}(1\leqslant i\leqslant \rho)$ 执行如下步骤.

① 发送者 ID_S 计算得到索引 $J_{R_i}=F_{index}^{\rho}(ID_{R_i})$,此时,$1\leqslant J_{R_i}\leqslant \rho$;

② 计算 $h_1^{R_i}=H_1(ID_{R_i},U_{R_i}),X_{R_i}=\alpha_s(P_{pub}h_1^{R_i}+U_{R_i}),K_{R_i}=H_3(ID_{R_i},X_{R_i})$ 和 $\eta_{R_i}=\gamma_s+\alpha_s h_1^{R_i}$;

③ 计算 $c_{R_i}=(m_{R_i}\|\eta_{R_i})\oplus K_{R_i}$,并将其存于集合 C 中位置为 J_{R_i} 处,即 $C[J_{R_i}]\leftarrow c_{R_i}$;

④ 计算 $d_{R_i}=H_2(ID_s,ID_{R_i},c_{R_i},A_s),t_{R_i}=H_2(ID_s,ID_{R_i},c_{R_i},X_{R_i})$;

⑤ 计算 $V_{R_i}=d_{R_i}\gamma_s+\alpha_s t_{R_i}$,并将其存储于集合 V 中位置为 J_{R_i} 处,即 $V[J_{R_i}]\leftarrow V_{R_i}$.

3.2.4 解签密(UnSignCrypt)

接收者收到密文信息 $\phi=\{A_s,C,V\}$ 后,首先计算得到 $J_{R_i}=F_{index}^{\rho}(ID_{R_i})$,并从集合 C 中获得 $c'_{J_{R_i}}$,从集合 V 中获得 $V'_{J_{R_i}}$,其中,$1\leqslant J_{R_i}\leqslant \rho$.

- 校验接收者身份

接收者 $ID_{J_{R_i}}(1\leqslant i\leqslant \rho)$ 首先要校验接收者的身份信息,其步骤如下.

① 计算 $d'_{J_{R_i}}=H_2(ID_s,ID_{J_{R_i}},c'_{J_{R_i}},A_s),h_1^s=H_1(ID_s,U_s),X'_{R_i}=A_s\gamma_{R_i}$ 以及 $t'_{J_{R_i}}=H_2(ID_s,ID_{J_{R_i}},c'_{J_{R_i}},X'_{R_i})$;

② 校验 $V'_{J_{R_i}}P-t'_{J_{R_i}}A_s=d'_{J_{R_i}}(P_{pub}h_1^s+U_s)$ 是否成立:如果不成立,则拒绝解密该信息.

- 解密密文

① 计算 $K'_{J_{R_i}}=H_3(ID_{J_{R_i}},X'_{R_i})$;

② 计算得到 $m'_{J_{R_i}}\|\eta'_{J_{R_i}}=c'_{J_{R_i}}\oplus K'_{J_{R_i}}$,进而得到明文 $m'_{J_{R_i}}$ 和参数 $\eta'_{J_{R_i}}$.

3.2.5 校验消息(VerifySign)

① 计算 $h_1^{J_{R_i}}=H_1(ID_{J_{R_i}},U_{J_{R_i}})$;

② 校验 $\eta'_{J_{R_i}}P=A_s h_1^{J_{R_i}}+U_s+P_{pub}h_1^s$ 是否成立:如果失败,则拒绝接收该消息.

3.2.6 KGC 密钥更新(UpdateKey)

对于每一个 $KS_i(i\in[1,n])$,首先重新选择一个 $t-1$ 阶的多项式 $f_i(x)=b_{i,1}x+b_{i,2}x^2+\ldots+b_{i,t-1}x^{t-1}$,其中,$b_{i,j}\in Z_p^*(1\leqslant j\leqslant t-1)$.$KS_i$ 计算 $\{b_{i,j}P|1\leqslant j\leqslant t-1\}$,并将其公布出去.然后,$KS_i$ 计算 $\{f_i(j)|1\leqslant j\leqslant n\}$,并将 $\{f_i(j)|1\leqslant j\leqslant n,j\neq i\}$ 依次发送至对应的 KS_j.KS_j 对收到的 $f_i(j)$ 进行校验,其校验公式是 $f_i(j)P=\sum_{\xi=1}^{t-1}(j^{\xi}(b_{i,\xi}P))$. 如果校验成功,则接收 $f_i(j)$.KS_j 对接收到的 n 项 $f_i(j)$ 全部校验成功后,重新计算其对应的子秘密 $s'_j=s_j+\sum_{\xi=1}^{n}f_\xi(j)$.

4 正确性分析

定理 1. KGC 生成用户 i 对应的私钥信息 $\gamma=\sum_{\xi=i_1}^{i_t}w_\xi\delta_\xi+u_{id}=sh_1^{id}+u_{id}$ 成立.

证明:设 $\mathbb{G}(x)=\sum_{\xi=1}^{n}g_\xi(x)$,由于 $g_i(x)=a_{i,0}+a_{i,1}x+\ldots+a_{i,t-1}x^{t-1}$,$s_j=\sum_{\xi=1}^{n}g_\xi(j)$,可知:

$$\begin{cases}g_1(x)=a_{1,0}+a_{1,1}x+\ldots a_{1,t-1}x^{t-1}\\ g_2(x)=a_{2,0}+a_{2,1}x+\ldots a_{2,t-1}x^{t-1}\\ \ldots\\ g_n(x)=a_{n,0}+a_{n,1}x+\ldots a_{n,t-1}x^{t-1}\end{cases}\Rightarrow\begin{cases}s_1=g_1(1)+g_2(1)+\ldots+g_n(1)\\ s_2=g_1(2)+g_2(2)+\ldots+g_n(2)\\ \ldots\\ s_n=g_1(n)+g_2(n)+\ldots+g_n(n)\end{cases}\Rightarrow\begin{cases}s_1=\mathbb{G}(1)\\ s_2=\mathbb{G}(2)\\ \ldots\\ s_n=\mathbb{G}(n)\end{cases}.$$

由于 $\mathbb{G}(x)$ 最大为 $t-1$ 次多项式,故最多只需要 t 对 $\{(i,\mathbb{G}(i))|1\leqslant i\leqslant n\}$ 即可求解 $\mathbb{G}(x)$.根据拉格朗日插值定理,可以得到:

$$\sum_{\xi=i_1}^{i_t}\mathbb{G}(\xi)\prod_{\substack{i_1\leqslant j\leqslant i_t\\ j\neq\xi,\,j\in I}}\frac{j}{j-\xi}=\sum_{\xi=i_1}^{i_t}s_\xi\prod_{\substack{i_1\leqslant j\leqslant i_t\\ j\neq\xi,\,j\in I}}\frac{j}{j-\xi}=a_{1,0}+a_{2,0}+\ldots+a_{n,0}=\sum_{\xi=1}^{n}a_{\xi,0}=s.$$

进而得到:

$$\gamma=\sum_{\xi=i_1}^{i_t}w_\xi\delta_\xi+u_{id}=\sum_{\xi=i_1}^{i_t}s_\xi h_1^{id}\prod_{\substack{i_1\leqslant j\leqslant i_t\\ j\neq\xi,\,j\in I}}\frac{j}{j-\xi}+u_{id}=\left(\sum_{\xi=i_1}^{i_t}s_\xi\prod_{\substack{i_1\leqslant j\leqslant i_t\\ j\neq\xi,\,j\in I}}\frac{j}{j-\xi}\right)h_1^{id}+u_{id}=sh_1^{id}+u_{id}.$$ □

定理 2. 检验密文接收用户 $ID_{J_{R_i}}(1\leqslant i\leqslant\rho)$ 身份合法的公式 $V'_{J_{R_i}}P-t'_{J_{R_i}}A_s=d'_{J_{R_i}}(P_{pub}h_1^s+U_s)$ 成立.

证明: $V'_{J_{R_i}}P-t'_{J_{R_i}}A_s=d'_{J_{R_i}}\gamma_sP+\alpha_st'_{J_{R_i}}P-t'_{J_{R_i}}A_s=d'_{J_{R_i}}\gamma_sP+t'_{J_{R_i}}A_s-t'_{J_{R_i}}A_s=d'_{J_{R_i}}\gamma_sP=d'_{J_{R_i}}(P_{pub}h_1^s+U_s).$ □

定理 3. 密文接收用户 $ID_{J_{R_i}}(1\leqslant i\leqslant\rho)$ 解密消息后,检验消息完整性公式 $\eta'_{J_{R_i}}P=A_sh_1^{J_{R_i}}+U_s+P_{pub}h_1^s$ 成立.

证明: $\eta'_{J_{R_i}}P=\gamma_sP+\alpha_{ID_S}h_1^{J_{R_i}}P=sPh_1^s+u_sP+A_sh_1^{J_{R_i}}=P_{pub}h_1^s+U_s+A_sh_1^{J_{R_i}}$. □

定理 4. 执行 KGC 密钥更新操作之后,系统主密钥并没有改变.

证明:执行密钥更新操作之前 $s=\sum_{i=1}^{n}g_i(0)$, 当 KS_i 执行密钥更新操作后,设定 $f_i'(x)=g_i(x)+f_i(x)$, 系统主密钥为 $s'=\sum_{i=1}^{n}f_i'(0)=\sum_{i=1}^{n}(g_i(0)+f_i(0))=\sum_{i=1}^{n}g_i(0)=s$, 故更新 KGC 密钥信息后,系统主密钥并没有改变. □

5 安全性分析

5.1 机密性

定理 5. 本方案具有攻击方式为 $\mathcal{A}_1$ 的选择密文攻击下的保密性.即在随机预言机模型下,敌手 $\mathcal{A}_1^1$ 以不可忽略的优势 ε 赢得定义 1 中的游戏.如果在游戏中,敌手 $\mathcal{A}_1^1$ 最多进行 ς 次签密查询和 ξ 次私钥生成查询,则挑战者 $\mathfrak{I}$ 能以不可忽略的优势 $Adv_{\mathcal{A}_1}^{C}(k)>\left(1-\frac{\xi}{2^k}\right)\frac{\varepsilon}{\mathrm{e}(\varsigma+1)}$ 解决 CDH 问题(其中,e 为自然对数的底数).

证明:令算法 Γ 的输入为 CDH 问题的挑战实例 $\langle G,aG,bG\rangle$,其中, $a,b\in Z_p^*$,而且其值未知,算法 Γ 的目的就是计算出 abG 的数值.算法 Γ 以敌手 $\mathcal{A}_1^1$ 为子程序来求解上述问题,并充当定义 1 中的挑战者 $\mathfrak{I}$.首先,挑战者 $\mathfrak{I}$ 执行 Setup 步骤,令 $P_{pub}=aG$,进而得到公共参数 $Params=\{G,p,q,P,P_{pub},H_1,H_2,H_3,F_{index}^{\rho},\oplus,\|\}$,该主密钥信息不被敌手 $\mathcal{A}_1^1$ 获知,然后将 $Params$ 发送至敌手 $\mathcal{A}_1^1$.挑战者 $\mathfrak{I}$ 维护列表 $L_1,L_2,L_3,L_{sk},L_{pk}$,用于记录相关的询问结果.其中,$L_1$ 跟踪预言机 H_1,L_2 用于跟踪预言机 H_2,L_3 用于跟踪预言机 H_3,L_{sk} 用于跟踪私钥生成询问结果,L_{pk} 用于跟踪公钥生成询问结果.上述列表初始化为空.

- 询问阶段

H_2 询问:挑战者 $\mathfrak{I}$ 收到敌手 $\mathcal{A}_1^1$ 关于 $H_2(ID_i,ID_j,c_j,A_i(X_i))$ 的询问信息时,若存在 $\langle ID_i,ID_j,c_j,A_i(X_i),h_2'\rangle\in L_2$,则返回 h_2 给敌手 $\mathcal{A}_1^1$;否则,挑战者 $\mathfrak{I}$ 选取满足 $\langle ID_i,ID_j,c_j,A_i(X_i),h_2'\rangle\notin L_2$ 的随机数 h_2 作为询问结果返回给敌手 $\mathcal{A}_1^1$,并添加 $\langle ID_i,ID_j,c_j,A_i(X_i),h_2'\rangle$ 到 L_2 中.

H_3 询问:挑战者 $\mathfrak{I}$ 收到敌手 $\mathcal{A}_1^1$ 关于 $H_3(ID_i,X_i)$ 的询问信息时,若存在 $\langle ID_i,X_i,h_3'\rangle\in L_3$,则返回相应的 h_3' 给敌手 $\mathcal{A}_1^1$;否则,$\mathfrak{I}$ 选取满足条件 $\langle ID_i,X_i,h_3'\rangle\notin L_3$ 的随机数 h_3',并添加 $\langle ID_i,X_i,h_3'\rangle$ 到 L_3 中.

公钥生成询问:挑战者 $\mathfrak{I}$ 收到敌手 $\mathcal{A}_1^1$ 询问用户 ID_i 对应的公钥信息时,$\mathfrak{I}$ 会查询列表 L_{pk}:如果列表 L_{pk} 存在元素 $\langle ID_i,U_i',v_i\rangle$,其中,$v_i\in\{0,1\}$,则返回公钥信息 U_i' 给敌手 $\mathcal{A}_1^1$;否则,$\mathfrak{I}$ 随机选择 $v_i\in\{0,1\}$,且设定 $\Pr[c_i=1]=\delta$,其中, $\delta=\frac{1}{\varsigma+1}$,然后,$\mathfrak{I}$ 根据 v_i 的数值执行如下判断.

(1) 若 v_i=0,挑战者 $\mathfrak{I}$ 选取随机数 $h_1,\gamma_i' \in Z_p^*$,计算得到 $U_i' = \gamma_i' P - P_{pub}h_1$,挑战者 $\mathfrak{I}$ 添加 $\langle ID_i,U_i',v_i\rangle$ 到 L_{pk} 中,并将公钥信息 U_i' 发送给敌手 λ_1^1,同时添加 $\langle ID_i,\gamma_i',v_i\rangle$ 到 L_{sk} 中,添加 $\langle ID_i,U_i',h_1\rangle$ 到 L_1 中;

(2) 若 v_i=1,选取 $b \in Z_p^*$ 以及 $h_1 \in Z_p^*$,令 $U_i' = bP$,挑战者 $\mathfrak{I}$ 添加 $\langle ID_i,U_i',v_i\rangle$ 到 L_{pk} 中,添加 $\langle ID_i,U_i',h_1\rangle$ 到 L_1 中,并返回公钥信息 $\kappa' = U_i'$ 给敌手 λ_1^1.

H_1 询问:挑战者 $\mathfrak{I}$ 收到敌手 λ_1^1 询问 H_1 中信息时,若存在 $\langle ID_i,U_i',h_1\rangle \in L_1$,则返回相应的 h_1 给敌手 λ_1^1;否则,$\mathfrak{I}$ 对 ID_i 进行公钥生成询问后,返回 L_1 中的 h_1 给敌手 λ_1^1.

公钥替换询问:敌手 λ_1^1 可以在任何时间点随机生成用户 ID 对应的公钥 κ',并替换其合法公钥 κ.

私钥生成询问:当挑战者 $\mathfrak{I}$ 收到敌手 λ_1^1 询问 ID_i 的私钥信息时,若存在 $\langle ID_i,\gamma_i'\rangle \in L_{sk}$,则返回 γ_i' 给敌手 λ_1^1;否则,对 ID_i 执行公钥生成询问,获得相应的公钥信息 $\langle ID_i,U_i',v_i\rangle$.若 v_i=0,则此时已经成功生成对应的私钥信息,从 L_{sk} 中获取 $\langle ID_i,\gamma_i',v_i\rangle$,并返回私钥信息 γ_i' 给敌手 λ_1^1;若 v_i=1,挑战者 $\mathfrak{I}$ 终止游戏并退出.

签密询问:当挑战者 $\mathfrak{I}$ 收到敌手 λ_1^1 关于 $\langle ID_s,ID_R=\{ID_{R_1},ID_{R_2},...,ID_{R_\rho}\},M=\{m_1,m_2,...m_\rho\}\rangle$ 的签密询问时,挑战者 $\mathfrak{I}$ 首先从 L_{pk} 中查询 ID_s 所对应$\langle ID_s,U_s,v_s\rangle$信息,并执行如下操作:若 v_s=1,则 $\mathfrak{I}$ 终止模拟,并退出游戏;若 v_s=0,则 $\mathfrak{I}$ 对 ID_s 执行私钥生成询问,获取 γ_s 信息,对 $ID_{R_i}(1\leqslant i\leqslant \rho)$ 执行公钥生成询问,并获取 $\langle ID_{R_i},U_{R_i}\rangle$ 信息,然后运行 SignCrypt 算法,获取密文信息 $\langle ID_{R_i},C=\{c_{R_1},c_{R_2},...,c_{R_\rho}\}\rangle$,并将密文发送至敌手 λ_1^1.

解签密询问:当挑战者 $\mathfrak{I}$ 收到敌手 λ_1^1 关于 ID_s, ID_{R_i} 和密文 c_{R_i} 的解签密询问时,挑战者 $\mathfrak{I}$ 首先对 ID_s 执行公钥生成询问,获得$\langle ID_s,U_s,v_s\rangle$信息,并根据 v_s 的数值进行如下的计算.

(1) 如果 $\langle ID_s,U_s',v_s\rangle \in L_{pk}$ 且 v_s=0,则 $\mathfrak{I}$ 对密文信息执行 UnSignCrypt 算法,并返回结果给敌手 λ_1^1.

(2) 如果 $\langle ID_s,U_s',v_s\rangle \in L_{pk}$ 且 v_s=1,对 ID_{R_i} 执行私钥生成询问,获取 γ_{R_i} 信息,对 ID_s 执行公钥生成询问,并获取 $\langle ID_s,U_s',v_s\rangle$ 信息,查询 H_1 获得 $\langle ID_{R_i},U_{R_i}',h_1^{R_i}\rangle \in L_1,\langle ID_s,U_s',h_1^s\rangle \in L_1$,然后计算得到 $X_{R_i}' = A_s'\gamma_{R_i},K_{J_{R_i}}' = H_3(ID_{R_i},X_{R_i}')$,计算得到 $m'\|\eta_{R_i}' = c_{R_i}' \oplus K_{R_i}'$.若执行校验等式 $\eta_{R_i}'P = A_s'h_1^{R_i} + U_s' + P_{pub}h_1^s$ 成立,则返回明文信息 m'给敌手 λ_1^1;否则密文信息无效,游戏退出.

(3) 如果 $\langle ID_s,U_s',v_s\rangle \notin L_{pk}$,则说明公钥被替换,此时以 ID_{R_i} 和 ID_s 为索引,搜索 H_1,L_{pk} 和 L_{sk},分别得到 $\langle ID_s,\boldsymbol{U}_s',v_s\rangle,\langle ID_{R_i},U_{R_i}',h_1^{R_i}\rangle \in L_1,\langle ID_s,U_s',h_1^s\rangle \in L_1$ 和 γ_{R_i} 信息,计算得到 $X_{R_i}' = A_s'\gamma_{R_i},K_{R_i}' = H_3(ID_{R_i},X_{R_i}')$,计算得到 $m'\|\eta_{R_i}' = c_{R_i}' \oplus K_{R_i}'$.若执行校验等式 $\eta_{R_i}'P = A_s'h_1^{R_i} + \boldsymbol{U}_s' + P_{pub}h_1^s$ 成立,则返回明文信息 m'给敌手 λ_1^1;否则密文信息无效,游戏退出.

- 挑战阶段

敌手 λ_1^1 选取一对等长的明文信息(m_0,m_1)以及两个身份 (ID_s,ID_{R_i}),敌手 λ_1^1 将挑战信息发送至 $\mathfrak{I}$.$\mathfrak{I}$ 收到敌手发送的明文信息和身份信息后,首先,$\mathfrak{I}$ 对 ID_s 执行公钥生成询问,挑战者 $\mathfrak{I}$ 可获得 L_{pk} 中该用户对应身份信息 $\langle ID_s,U_s',v_s\rangle$.然后挑战者 $\mathfrak{I}$ 执行如下操作:如果 v_s=0,则 $\mathfrak{I}$ 终止游戏,退出模拟过程;若 v_s=1,挑战者随机选择 $\kappa\in\{0,1\}$,然后随机选择 $\alpha_s \in Z_p^*$,并计算得到 A_s=$\alpha_s P$.对于接收者 $ID_{R_i}(1\leqslant i\leqslant \rho)$ 以及发送者 ID_s,计算 $h_1^{R_i} = H_1(ID_{R_i},U_{R_i})$,选择 $b \in Z_p^*$,计算得到 $U_{R_i} = bP$,使其满足 $\gamma_{R_i}P = P_{pub}h_1^{R_i} + U_{R_i}$,然后计算 $X = \alpha_s(P_{pub}h_1^{R_i} + U_{R_i}),K_{R_i} = H_3(ID_{R_i},X_{R_i}),\eta_{R_i} = \gamma_s + \alpha_s h_1^{R_i},c_{R_i} = (m_\kappa\|\eta_{R_i}) \oplus K_{R_i}$,并将密文 $\{c_{R_i},U_{R_i}\}$ 发送敌手 λ_1^1.

- 猜测阶段

敌手 λ_1^1 收到 $\{c_{R_i},U_{R_i}\}$ 后,猜测 $\kappa'\in\{0,1\}$:如果 κ'=κ,则 $\mathfrak{I}$ 可输出 CDH 的解为 $abG = (\gamma_{R_i} - u_{R_i})U_{R_i}h_1^{-1}$;否则,$\mathfrak{I}$ 没有解决 CDH 问题.挑战者 $\mathfrak{I}$ 模拟了真实场景下攻击模式,若挑战者 $\mathfrak{I}$ 在整个游戏过程中并未终止,而且敌手 λ_1^1 以不可忽略的优势 ε 攻破系统的机密性,则挑战者 $\mathfrak{I}$ 即可输出 CDH 问题的有效解.

令事件 P 表示敌手 λ_1^1 未对挑战者身份 ID_s 进行私钥生成询问,即 $\Pr(P) = 1 - \frac{\xi}{2^k}$;事件 P'表示签密阶段未结

束,即 $\Pr(P')=(1-\delta)^{\varsigma}$;事件 P''表示挑战阶段未终止,即 $\Pr(P'')=\delta$.则游戏未终止的概率至少为 $\Pr(P\wedge P'\wedge P'')=\left(1-\frac{\xi}{2^k}\right)(1-\delta)^{\varsigma}\delta$.由于 $\delta=\frac{1}{\varsigma+1}$,当$\varsigma$足够大时,$(1-\delta)^{\varsigma}\to\frac{1}{\mathrm{e}}$,故 $\Pr(P\wedge P'\wedge P'')>\left(1-\frac{\xi}{2^k}\right)\frac{1}{\mathrm{e}(\varsigma+1)}$.

综上所述,若挑战者$\mathfrak{I}$在整个游戏过程中没有终止,且敌手 $\hat{\lambda}_{\mathrm{I}}^{1}$ 以不可忽略的优势ε在多项式时间内赢得相关游戏,则$\mathfrak{I}$能以优势 $Adv_{\hat{\lambda}_1}^{c}(k)>\left(1-\frac{\xi}{2^k}\right)\frac{\varepsilon}{\mathrm{e}(\varsigma+1)}$ 解决 CDH 问题. □

定理 6. 本方案具有攻击方式为$\hat{\lambda}_2$ 下的选择密文攻击的保密性,即在随机预言机模型下,敌手 $\hat{\lambda}_2^{1}$ 以不可忽略的优势ε赢得定义 2 中的游戏.如果在游戏中,敌手 $\hat{\lambda}_2^{1}$ 最多进行ς次签密查询和ξ次私钥生成查询,则挑战者$\mathfrak{I}$能以不可忽略的优势 $Adv_{\hat{\lambda}_2}^{c}(k)>\left(1-\frac{\xi}{2^k}\right)\frac{\varepsilon}{\mathrm{e}(\varsigma+1)}$ 解决 CDH 问题.

证明:证明过程同定理 5 类似,这里不再赘述. □

5.2 不可伪造性

定理 7. 本方案具有攻击方式为$\hat{\lambda}_{\mathrm{I}}$ 下的选择消息攻击下的不可伪造性,即在随机预言机模型下,敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$ 以不可忽略的优势ε赢得定义 3 中的游戏.如果在游戏中,敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$ 最多进行ς次签名查询,则挑战者$\mathfrak{I}$能以不可忽略的优势 $Adv_{\hat{\lambda}_1}^{u}(k)>\frac{\varepsilon}{\mathrm{e}(\varsigma+1)}$ 解决 DL 问题(其中,e 为自然对数的底数).

证明:令算法Γ的输入为 DL 问题的挑战实例$\langle G,aG\rangle$,其中,$a\in Z_p^*$,而且其值未知,算法Γ的目的就是计算出 a 的数值.算法Γ以敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$ 为子程序来计算上述问题,并充当定义 3 中挑战者$\mathfrak{I}$.首先,挑战者$\mathfrak{I}$执行 Setup 步骤,令 $P_{pub}=aG$,进而得到公共参数 $Params=\{G,p,q,P,P_{pub},H_1,H_2,H_3,F_{index}^{\rho},\oplus,\|\}$;然后将 $Params$ 发送至敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$.挑战者$\mathfrak{I}$维护列表 $L_1,L_2,L_3,L_{sk},L_{pk}$,以跟踪对预言机 H_1,H_2,H_3 以及私钥生成、公钥生成的询问结果,上述列表初始化为空.

- 询问阶段

敌手执行定理 1 中对预言机 H_2,H_3 以及公钥替换的询问.

公钥生成询问:挑战者$\mathfrak{I}$收到敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$ 询问用户 ID_i 对应的公钥信息时,$\mathfrak{I}$会查询列表 L_{pk}:如果列表 L_{pk} 存在元素 $\langle ID_i,U_i',\nu_i\rangle$,其中,$\nu_i\in\{0,1\}$,则返回公钥信息 U_i' 给敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$;否则,$\mathfrak{I}$随机选择 $\nu_i\in\{0,1\}$,且设定 $\Pr[c_i=1]=\delta$,其中,$\delta=\frac{1}{\varsigma+1}$.若 $\nu_i=0$,挑战者$\mathfrak{I}$选取随机数 $h_1,\gamma_i'\in Z_p^*$,计算得到 $U_i'=\gamma_i'P-P_{pub}h_1$.挑战者$\mathfrak{I}$添加 $\langle ID_i,U_i',\nu_i\rangle$ 到 L_{pk} 中,并将公钥信息 U_i' 发送给敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$,同时添加 $\langle ID_i,\gamma_i'\rangle$ 到 L_{sk} 中,添加 $\langle ID_i,U_i',h_1\rangle$ 到 L_1 中.若 $\nu_i=1$,选取 $\gamma_i',b\in Z_p^*$ 以及 $h_1\in Z_p^*$,使得 $\langle *,\gamma_i'\rangle\notin L_{sk}$.令 $U_i'=bP$,挑战者$\mathfrak{I}$添加 $\langle ID_i,U_i',\nu_i\rangle$ 到 L_{pk} 中,添加 $\langle ID_i,U_i',h_1\rangle$ 到 L_1 中,添加 $\langle ID_i,\gamma_i'\rangle$ 到 L_{sk} 中,并返回公钥信息 $\kappa'=U_i'$ 给敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$.

敌手执行定理 1 中对预言机 H_1 的询问.

私钥生成询问:当挑战者$\mathfrak{I}$收到敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$ 询问 ID_i 的私钥信息时,若存在 $\langle ID_i,\gamma_i'\rangle\in L_{sk}$,则返回 γ_i' 给敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$;否则对 ID_i 执行公钥生成询问,并从 L_{sk} 返回私钥信息 γ_i' 给敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$.

签名询问:当挑战者$\mathfrak{I}$收到敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$ 关于 $\langle ID_s,ID_R=\{ID_{R_1},ID_{R_2},...,ID_{R_\rho}\},M=\{m_1,m_2,...m_\rho\}\rangle$ 的签名询问时,挑战者$\mathfrak{I}$首先从 L_{pk} 中查询 ID_s 所对应的$\langle ID_s,U_s,\nu_s\rangle$信息,并执行如下操作:若 $\nu_s=1$,则$\mathfrak{I}$终止模拟,并退出游戏;若 $\nu_s=0$,则$\mathfrak{I}$对 ID_s 执行私钥生成询问,获取γ_s 信息,对 $ID_{R_i}(1\leqslant i\leqslant\rho)$ 执行公钥生成询问,并获取 $\langle ID_{R_i},U_{R_i}\rangle$ 信息,然后运行 *SignCrypt* 算法,获取签名信息,并发送至敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$.

签名验证询问:当挑战者$\mathfrak{I}$收到敌手 $\hat{\lambda}_{\mathrm{I}}^{2}$ 关于 ID_s,ID_{R_i} 和 $\langle ID_{R_i},c_{R_i},m_{R_i}\rangle$ 的签名验证询问时,挑战者$\mathfrak{I}$首先对 ID_s 执行公钥生成询问,获得$\langle ID_s,U_s,\nu_s\rangle$信息,并根据 ν_s 的数值进行如下的计算.

(1) 如果$\langle ID_s,U_s,v_s\rangle\in L_{pk}$且$v_s$=0,则挑战者$\mathfrak{I}$执行 UnSignCrypt 算法,并返回结果给敌手λ_1^2.

(2) 如果$\langle ID_s,U_s,v_s\rangle\in L_{pk}$且$v_s$=1,则挑战者$\mathfrak{I}$需要执行如下步骤:首先对$ID_{R_i}$执行私钥生成询问,获取$\gamma_{R_i}$信息,对$ID_s$执行公钥生成询问,并获取$\langle ID_s,U_s,v_s\rangle$信息,查询$H_1$获得$\langle ID_{R_i},U_{R_i},h_1^{R_i}\rangle\in L_1,\langle ID_s,U_s,h_1^s\rangle\in L_1$,然后计算得到$X'_{R_i}=A'_s\gamma_{R_i},K'_{J_{R_i}}=H_3(ID_{R_i},X'_{R_i})$,计算得到$m'\|\eta'_{R_i}=c'_{R_i}\oplus K'_{R_i}$.如果执行校验等式$\eta'_{R_i}P=A'_sh_1^{R_i}+U_s+P_{pub}h_1^s$成立,则返回信息$m'$给敌手$\lambda_1^2$;否则,挑战者$\mathfrak{I}$退出游戏.

(3) 如果$\langle ID_s,U_s,v_s\rangle\notin L_{pk}$,则说明公钥被替换,此时以$ID_{R_i}$和$ID_s$为索引,搜索$H_1$,$L_{pk}$和$L_{sk}$,分别得到$\langle ID_s,U'_s,v_s\rangle$,$\langle ID_{R_i},U'_{R_i},h_1^{R_i}\rangle\in L_1$,$\langle ID_s,U'_s,h_1^s\rangle\in L_1$和$\gamma_{R_i}$信息,计算得到$X'_{R_i}=A'_{ID_s}\gamma_{R_i},K'_{J_{R_i}}=H_3(ID_{R_i},X'_{R_i})$,计算得到$m'\|\eta'_{R_i}=c'_{R_i}\oplus K'_{R_i}$.若执行校验等式$\eta'_{R_i}P=A'_sh_1^{R_i}+U'_s+P_{pub}h_1^s$成立,则并返回明文信息$m'$给敌手$\lambda_1^2$;否则密文信息无效,游戏退出.

- 伪造阶段

经过有限次上述询问后,敌手λ_1^2伪造ID_s关于消息$M=\{m_1,m_1,\ldots,m_\rho\}$和接收者$ID_R=\{ID_{R_1},ID_{R_2},...,ID_{R_\rho}\}$的伪造签名.随机选择$\alpha_s\in Z_p^*$,并计算得到$A_s=\alpha_sP$.对于接收者$ID_{R_i}(1\leqslant i\leqslant\rho)$以及发送者$ID_s$,计算$h_1^{R_i}=H_1(ID_{R_i},U_{R_i})$,随机选择$\alpha_s\in Z_p^*$,计算得到$A_s=\alpha_sP$,然后计算:

$$X=\alpha_s(P_{pub}h_1^{R_i}+U_{R_i}),K_{R_i}=H_3(ID_{R_i},X_{R_i}),\eta_{R_i}=\gamma_s+\alpha_sh_1^{R_i},c_{R_i}=(m_\kappa\|\eta_{R_i})\oplus K_{R_i}.$$

若敌手λ_1^2伪造成功,则挑战者$\mathfrak{I}$以ID_s为索引查询L_{pk}获得$\langle ID_s,U'_s,v_s\rangle$,计算$h_1^s=H_1(ID_s,U'_s)$.若$v_s$=1,则挑战者$\mathfrak{I}$输出$a=(S^*h_1^s)^{-1}(\alpha_s-u_sS^*)$作为 DL 问题的解,其中,$S^*=\alpha_s\gamma_s^{-1}$;若$v_s$=0,挑战者$\mathfrak{I}$终止模拟,未解答 DL 问题.

令事件P表示签名询问过程中挑战者$\mathfrak{I}$没有终止游戏,即$\Pr(P)=(1-\delta)^\varsigma$,则挑战者$\mathfrak{I}$询问阶段不终止的概率为$(1-\delta)^\varsigma$,伪造阶段不终止的概率为$\delta$,则游戏未终止的概率至少为$(1-\delta)^\varsigma\delta$.由于$\delta=\frac{1}{\varsigma+1}$,当$\varsigma$足够大的时候,$(1-\delta)^\varsigma\to\frac{1}{\mathrm{e}}$,故游戏不终止的概率至少为$\frac{1}{\mathrm{e}(\varsigma+1)}$.

综上所述,若$\mathfrak{I}$在整个游戏过程中没有终止,且敌手λ_1^2以不可忽略的优势ε在多项式时间内赢得相关游戏,则$\mathfrak{I}$能以优势$Adv_{\lambda_1}^u(k)>\frac{\varepsilon}{\mathrm{e}(\varsigma+1)}$解决 DL 问题. □

定理 8. 本方案具有攻击方式为λ_2下的选择消息攻击下的不可伪造性,即在随机预言机模型下,敌手λ_2^2以不可忽略的优势ε赢得定义 4 中的游戏.如果在游戏中,敌手λ_2^2最多进行ς次签名查询,则挑战者$\mathfrak{I}$能以不可忽略的优势$Adv_{\lambda_2}^u(k)>\frac{\varepsilon}{\mathrm{e}(\varsigma+1)}$解决 DL 问题(其中,e 为自然对数的底数).

证明:证明过程同定理 7 类似,这里不再赘述. □

5.3 密钥托管安全性

在本方案中,KGC 负责系统主密钥托管,用户将身份信息 ID 发送至 KGC 中各个KS_i,KS_i基于 ID 和子秘密s_i计算得到w_i并发送至用户.每隔一定周期,KS_i会重新选择一个t–1 阶多项式$f_i(x)$.设定敌手连续两个周期ε和ε+1 攻破了t个服务器并成功从中搜集了t份s_i,即$\{s_1^\varepsilon,s_2^\varepsilon,...,s_\upsilon^\varepsilon,s_{\upsilon+1}^{\varepsilon+1},...,s_t^{\varepsilon+1}\}$,其中,$\upsilon<t$.$KS_i$在$\varepsilon$+1 时刻更新其子秘密$s_i^{\varepsilon+1}=s_i^\varepsilon+\sum_{\xi=1}^n f_\xi(i)$.对于周期$\varepsilon$的系统主密钥$s^\varepsilon$和周期$\varepsilon$+1 的系统主密钥$s^{\varepsilon+1}$,根据拉格朗日插值可知:

$$\sum_{\xi=1}^t s_\xi^\varepsilon\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi,\,j\in I}}\frac{j}{j-\xi}=s^\varepsilon,\sum_{\xi=1}^t s_\xi^{\varepsilon+1}\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi,\,j\in I}}\frac{j}{j-\xi}=s^{\varepsilon+1}.$$

由定理 4 可知,执行子秘密更新后,系统主密钥不变.则可以得到:

$$s=s^{\varepsilon}=s^{\varepsilon+1}=\sum_{\xi=1}^{t}s_{\xi}^{\varepsilon}\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi,\,j\in I}}\frac{j}{j-\xi}=\sum_{\xi=1}^{t}s_{\xi}^{\varepsilon+1}\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi,\,j\in I}}\frac{j}{j-\xi}.$$

敌手若要从子秘密集合 $\{s_1^{\varepsilon},s_2^{\varepsilon},...,s_{\upsilon}^{\varepsilon},s_{\upsilon+1}^{\varepsilon+1},...,s_t^{\varepsilon+1}\}$ 中获得系统主密钥 s,则可知:

$$\begin{aligned}\sum_{\xi=1}^{\upsilon}s_{\xi}^{\varepsilon}\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi}}\frac{j}{j-\xi}+\sum_{\xi=\upsilon+1}^{t}s_{\xi}^{\varepsilon+1}\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi}}\frac{j}{j-\xi}&=\sum_{\xi=1}^{\upsilon}s_{\xi}^{\varepsilon}\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi}}\frac{j}{j-\xi}+\sum_{\xi=\upsilon+1}^{t}(s_{\xi}^{\varepsilon}+\sum_{i=1}^{n}f_i(\xi))\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi}}\frac{j}{j-\xi}\\&=\sum_{\xi=1}^{t}s_{\xi}^{\varepsilon}\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi}}\frac{j}{j-\xi}+\sum_{\xi=\upsilon+1}^{t}\sum_{i=1}^{n}f_i(\xi)\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi}}\frac{j}{j-\xi}\\&=s+\sum_{\xi=\upsilon+1}^{t}\sum_{i=1}^{n}f_i(\xi)\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi}}\frac{j}{j-\xi}.\end{aligned}$$

由于 $\sum_{\xi=1}^{\upsilon}s_{\xi}^{\varepsilon}\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi}}\frac{j}{j-\xi}$, $\sum_{\xi=\upsilon+1}^{t}s_{\xi}^{\varepsilon+1}\prod_{\substack{1\leqslant j\leqslant t\\ j\neq\xi}}\frac{j}{j-\xi}$ 已知,因此,敌手若想获得系统主密钥 s,则其必须知道 $t-\upsilon$个元素 $\left\{\sum_{i=1}^{n}f_i(\upsilon+1),\sum_{i=1}^{n}f_i(\upsilon+2),...,\sum_{i=1}^{n}f_i(t)\right\}$ 的数值.由于 $t-\upsilon>0$,数组非空,因此敌手并不能通过在两个时间周期内搜集 t 份 s_i 来获得系统主密钥信息.

5.4 前后向兼容性

这里的前后向兼容性代表在 KGC 不同密钥更新周期注册的用户,相互之间能够通信.为了简化起见,这里设定两个用户在相邻两个周期生成自己的公私钥信息.下面证明这两个用户之间仍然能够进行通信.设定参与签密的用户 a 在周期ε生成自己的公私钥,用户 b 在周期ε+1 跟 KGC 协作生成自己的公私钥,其中,用户 a 的私钥为 $\gamma_a=\sum_{\xi=i_1}^{i_t}w_{\xi}\delta_{\xi}+u_a$,公钥为$\kappa_a=U_a$;用户 b 的私钥为 $\gamma_b=\sum_{\xi=i_1}^{i_t}w_{\xi}\delta_{\xi}+u_b$,公钥为$\kappa_b=U_b$.其中,$w_i=s_ih_{\mathrm{I}}^i$,$i=a,b$.

由于不同周期各个 KS_i 对应的 s_i 会动态更新,因此,不同周期的 w_i 并不相同.

根据定理 1 可知,$\gamma_i=\sum_{\xi=1}^{t}w_{\xi}\delta_{\xi}+u_i=sh_{\mathrm{I}}^i+u_i$,$U_i=u_iP$.用户生成的公私钥信息与周期相关因子 w_i 无关,因此,用户的公私钥不受周期更新的影响.

从本方案具有的前后向兼容性可知,在不同周期注册的用户相互之间不受 KGC 更新子秘密的影响,相互之间能够通信,不会影响系统的可用性.

5.5 不可否认性

由定理 7 和定理 8 可知,本文方案对于攻击方式λ_{I}和攻击方式λ_{II}具有不可伪造性,恶意第三方无法伪造密文信息.当密文信息发送至接收方时,接收方利用自己的私钥和发送方的身份信息对密文信息进行校验,发送方不能对自己合成的密文信息进行否认,因此,本文方案具有不可否认性.

6 安全属性及性能分析

在本部分,将重点考察签密算法的安全属性、计算复杂度以及对应的密文长度.

6.1 安全属性分析

当前,相关研究人员提出的多接收者签密算法,主要是基于离散对数、椭圆曲线、双线性映射等进行实现,在达到同等安全强度下,基于离散对数的实现算法,所需的密钥长度较长.近几年来,基于离散对数实现的多接收者签密方案较少,本部分不对其进行考察.本部分将从保密性、不可伪造性、密钥托管安全性、不可否认性等 4 个维度,对相关签密算法进行对比分析.

在对比密文长度时,为了方便起见,定义如下符号:L_m 代表明文长度,$|G_p|$代表阶数为 p 的循环群 G 中元素长度,$|Z_p^*|$为 Z_p^* 中整数的长度,$|ID|$代表参与方身份 ID 的长度,n 为接收者的数量,m 为发送者伪装身份个数.

安全属性对比结果见表 2,其中,×代表不满足该安全属性,√则代表满足.在对比密文长度时,对于双线性对操作:$e:G_1\times G_1\to G_2$.为了方便对比,这里设定两个群的阶数均为$|G_p|$.

Table 2 Comparison of security attributes

表 2 安全属性对比

签密方案	理论基础	密文长度	不可伪造性	保密性	不可否认性	密钥托管安全强度	通信模式
文献[4]	双线性对	$L_m+(n+3)\|G_p\|$	×	×	×	×	单消息
文献[5]	双线性对	$L_m+(n+2)\|G_p\|$	√	×	√	√	单消息
文献[6]	双线性对	$L_m+3\|G_p\|+n\|ID\|$	√	√	√	√	单消息
文献[24]	双线性对	$L_m+(n+1)\|Z_p^*\|+2n\|G_p\|+n\|ID\|$	√	√	√	√	单消息
文献[25]	双线性对	$m\|Z_p^*\|+(m+2n+2)\|G_p\|+L_m+m\|ID\|$	√	√	√	√	多消息
文献[26]	双线性对	$L_m+(n+2)\|G_p\|$	×	√	×	√	多消息
文献[23]	椭圆曲线	$\|Z_p^*\|+(n+1)\|G_p\|+nL_m$	√	√	√	√	多消息
本文	椭圆曲线	$nL_m+2n\|Z_p^*\|+\|G_p\|$	√	√	√	√(强)	多消息

对于通信模式,单消息模式代表多接收者签密方案一次只能发送单个消息,而多消息模式一次能给不同接收者发送不同的消息.本文提出的算法是一种多接收者多消息通信模式签密方案,能够适配到广播通信等应用场景中.

本文提出的多接收者多消息签密方案,采用多个 KGC 协同管理系统主密钥信息,且 KGC 能够动态更新自己的子秘密信息,进而实现密钥托管安全性.与表 2 中其他几种签密方案相比,本文方案能够有效抵御针对 KGC 的持续性攻击.

文献[23]与本文均是基于椭圆曲线实现的多消息签密方案,均具有不可伪造性和保密性,文献[23]密文长度也要比本文提出的方案要短,但本文方案能够有效抵御针对 KGC 的持续性攻击.此外,本文方案中,接收者在收到消息后,会首先对接收者身份信息进行校验,如果校验不通过,则可中止后续的解签密流程,减少了不必要的计算资源浪费.

文献[25]是基于双线性对实现的多消息签密方案,满足了不可伪造性、保密性、不可否认性等安全属性要求;此外,还具有发送者和接收者的身份匿名性.该方案适合于消息发送方和接收方互不信任的广播通信场景中,然而该方案为了达到身份匿名要求,产生的密文长度最长,效率也最低.

6.2 计算复杂度分析

本文提出的签密算法,基于椭圆曲线进行实现.为了便于对比计算复杂度,表 3 定义了相关符号,并根据文献[27]给出了其他操作换算成模乘操作对应的计算量,以方便不同算法之间进行对比.

Table 3 Symbolic representation of the proposed scheme

表 3 本文方案的符号表示

符号	定义	描述
T_m	模乘运算	×
T_e	指数运算	$T_e\approx 240T_m$
T_b	双线性对	$T_b\approx 87T_m$
T_{pm}	椭圆曲线点乘	$T_{pm}\approx 29T_m$

需要说明的是,由于模加法、模减法、哈希以及椭圆曲线加法计算开销较低,这里不对其进行考察.这里将各个方案计算量统一化为模乘运算计算量,具体实验数据见表 4.

Table 4 Comparison of computational complexity

表 4 计算复杂度对比

方案名称	签密		解签密	
	计算复杂度	统一化	计算复杂度	统一化
文献[7]	$nT_b+nT_e+T_{pm}$	$(327n+29)T_m$	$3T_b+T_e$	$501T_m$
文献[9]	$T_e+(3+m+n)T_{pm}$	$(29m+29n+327)T_m$	$4T_b+(4+m)T_{pm}$	$(29m+464)T_m$
文献[24]	$2nT_e+(n+1)T_{pm}$	$(509n+29)T_m$	$2T_b+3T_e+T_{pm}$	$923T_m$
文献[25]	$(n+2)T_b+(m+n+3)T_{pm}$	$(116n+29m+261)T_m$	$6T_b+(m+2)T_{pm}$	$(29m+580)T_m$
文献[23]	$(3n+1)T_{pm}$	$(87n+29)T_m$	$5T_{pm}$	$145T_m$
文献[28]	$T_e+(1+2n)T_{pm}$	$(58n+269)T_m$	T_e+2T_b	$414T_m$
文献[29]	$T_b+(2+n+m)T_e$	$(240n+240m+567)T_m$	$(n+3)T_b$	$(87n+261)T_m$
本文	$(2n+1)T_{pm}$	$(58n+29)T_m$	$7T_{pm}$	$203T_m$

从表 4 中可知,文献[23]和本文均是基于椭圆曲线实现的多接收者签密算法,无论是签密还是解签密过程,比其他几种基于双线性对实现的多接收者签密方案[7,9,24,25,28,29]的计算复杂度普遍要低.虽然在解签密过程中,文献[23]要比本文的方法执行效率较高,但在执行签密过程时,本文计算复杂度要低.此外,本文提出的方案在执行解签密操作时,接收者会首先利用自己的私钥信息进行身份校验,如果校验不通过,后续的解签密操作无需执行,可以减少接收者的计算负担,增加系统的执行效率.

当发送者伪装身份个数 m 较小而接收者的数量 n 较大时,文献[9]中的签密操作执行效率比本文方案更高,但是此时发送者的匿名性将会减弱,攻击者猜中其身份的概率将会增加,进而影响其安全性.此外,在执行解签密操作时,本文提出的方案,其性能要优于文献[9].

7 基于区块链的 KGC 密钥更新策略

本文提出的签密方案中,KGC 由一组节点组成,各节点以时间周期 τ同步更新子密钥信息.为了实现同步的目的,一种直接的方式是指定某一节点为时间服务器,在周期到来时,向其他节点广播同步消息,其他节点收到该消息后,执行密钥更新操作.然而,这种方式将会加重通信负担.此外,一旦时间服务器被攻破,将会影响整个系统的安全性.另外一种方法是设定 KGC 各个节点时刻一致,然后约定更新周期,各节点根据本地时间更新子密钥信息.然而,这种依赖本地时间的更新策略,会由于出现网络攻击、机器故障等问题导致各节点不能同步更新.本文提出了一种基于区块链的 KGC 密钥周期更新策略,根据公有链中块高度和块时间戳来触发密钥更新操作.新方案由区块不可篡改特性保证了周期更新过程的安全性.另外,该方案不需要执行交易过程,因此是免费的,最后,不需要 KGC 各节点之间执行通信过程,减少了带宽资源的消耗.

以以太坊为例进行说明,矿工在挖到一个区块时,会将自己本地时间记录到区块中作为区块时间戳,因此在这种场景中,矿工是可以根据自己的意愿随意调整挖出块的时间戳.但是实际中,矿工需要将挖出的区块进行全网公布,其他矿工需对该区块进行校验,该区块对应的时间戳必须大于其祖先块对应的时间戳,时间戳误差必须处于区块链全网节点容忍范围内,否则,上述区块会被网络拒绝.因此,主链区块时间戳在一定程度上能够反映当时真实时间.因此,基于以太坊实现同步更新操作具有一定的可行性.需要说明的是,在应用到 KGC 密钥更新过程中,这种时间误差会增加密钥更新过程的不确定性,使攻击者不能知道密钥更新的准确时间,加大了攻击者攻击的难度.

攻击者若要攻击该周期更新方案,首先必须要知道准确的更新周期,根据周期计算出对应区块编号,然后必须掌握 51%的算力,篡改上述区块的时间戳,而这在实践中是非常困难的.由于以太坊出块周期为 15s,因此在实际应用中,周期应尽量设置为 15s 的倍数.由于 KGC 密钥更新过程不需要过于频繁,15s 倍数的周期更新方案在本场景中是可行的.需要注意的是,以太坊区块时间戳是 Unix 时间戳,定义为从格林威治时间 1970 年 1 月 1 日 0 时 0 分 0 秒起至现在的总秒数,Unix 时间戳可以换算成“年/月/日/时/分/秒”的表述形式,后面不做说明的话,时刻均以 Unix 时间戳表述.

算法 1. 密钥更新策略 updateStrategy.

输入:开始执行同步的时刻 *start_timestamp*,更新周期 *period*.

输出:执行结果(TRUE/FALSE).

```
1.  blockHeight=eth.getblockNumber(·);
2.  current_timestamp=eth.getTimestamp(blockHeight);
3.  if (current_timestamp≥start_timestamp)
4.     return FALSE;
5.  while (is_exit(·)){
6.     sleep(1);
7.     blockHeight=eth.getblockNumber(·);
8.     current_timestamp=eth.getTimestamp(blockHeight);
9.     if (start_timestamp≥current_timestamp)
10.       start_timestamp=current_timestamp;
11.       break;
12. }
13. do {
14.    blockHeight=eth.getblockNumber(·);
15.    current_timestamp=eth.getTimestamp(blockHeight);
16.    if (current_timestamp−start_timestamp>min(period,30))
17.       sleep(min(period,30)/2);
18.    else
19.       sleep(1);
20.    if (current_timestamp≥start_timestamp+period){
21.       UpdateKey(·);
22.    }
23. } while (is_exit(·));
24. return TRUE;
```

UPDATE STRATEGY 算法执行流程描述.

1. 获取当前最新区块对应的时间戳,判断是否要早于开始执行同步的时刻:如果否,则退出程序.
2. 等待一定时间,直到当前最新区块对应的时间戳等于或晚于开始执行同步的时刻.
3. 等待 *period* 秒,然后执行密钥更新操作 *UpdateKey*(·).为了减少访问区块链的次数,这里设定:如果距离更新的时间大于 min(*period*,30),则执行睡眠的时间为 min(*period*,30)/2.

8 仿真实验

8.1 签密方案性能分析

本文椭圆曲线参数设置参考 Secp256k1.Secp256k1 是经优化的基于有限域$\mathbb{F}_p$的椭圆曲线,与其他曲线相比,其性能能够提升 30%左右[30].由于其密钥长度较短,故占用较小的存储和带宽资源.实验环境的具体配置如下.

- CPU: Intel(R) Core(TM) i5-7500 3.40GHz;
- RAM: 8.00GB;
- 磁盘: 128GB SSD+1TB SATA;
- OS: Windows 7 64.

实验采用的编程语言为 JAVA 1.7,IDE 环境采用 MyEclipse 1.5,采用的密码学开源库为 JPBC 2.0.0.

由于本文提出的多 KGC 方案只会影响签密方案中初始化和用户密钥生成两步执行效率,因此下面考察上述两步执行时间与 KGC 密钥服务器数量 N 以及门限值 t 之间的关系.设定发送者和接收者的数量均为 1,图 3 中给出了当密钥服务器 N 为 50 时,初始化以及用户密钥生成两步执行时间与门限值 t 之间的关系;图 4 中给出了当门限值 t 为 30 时,初始化以及用户密钥生成两步执行时间与服务器数 N 之间的关系.需要说明的是,后续实验均排除了网络延迟等因素的干扰,以更好地从计算复杂度角度考察算法的性能.

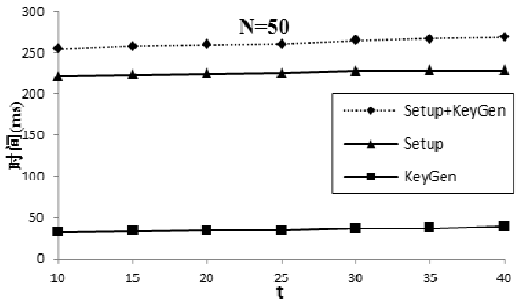

Fig.3 Execution time for the Setup phase and the keyGen phase over the threshold t

图 3 初始化以及用户密钥生成阶段门限值 t 对应的执行时间

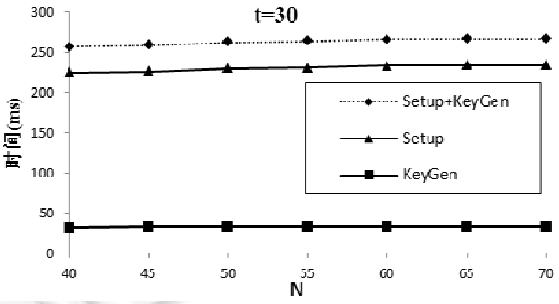

Fig.4 Execution time for the Setup phase and the keyGen phase over N

图 4 初始化以及用户密钥生成阶段 N 对应的执行时间

从图 3 和图 4 可知,随着 N 或 t 的增加,初始化和用户密钥生成时间之和也随着增加,但增加率不到 10%.当 N 为 50 时,t 从 10 递增到 40,初始化和用户密钥生成时间之和增加了不到 20ms.当 t 为 30 时,N 从 40 递增到 70,执行时间之和增加了约 10ms.t 主要影响多项式 $g_i(x)$的次数,以及用户在合成密钥时需要收集的 w_i 的数量;而 N 主要影响密钥分发中心在合成子秘密 s_j 时收集的 $g_i(j)$的数量.在排除网络延迟等因素的干扰下,N 和 t 的增加对初始化和用户密钥生成的执行时间影响不大.

下面考察方案总体执行时间与接收者数量之间的关系,以及引入多密钥分发中心后,对系统性能的影响.此时统计了初始化、用户密钥生成、签密和解签密这 4 个步骤的执行时间,解签密过程包括了算法的解签密(UnSignCrypt)步骤以及校验消息(VerifySign)步骤,而且由于解签密过程可以并发执行,所以解签密步骤只考察一个接收者执行的时间.设定密钥分发中心数量 N 为 3,门限值 t 为 2,消息发送者数量为 1,实验重复 50 次,并统计其平均值,实验结果如图 5 所示.为了考察 KGC 密钥更新以及其他步骤的执行效率,图 6 给出了算法各个步骤占总体时间百分比,此时统计了初始化、用户密钥生成、签密、解签密和密钥更新这 5 个步骤的执行时间占总体执行时间的百分比,此时,N,t 以及消息发送者数量同图 5 配置参数相同.

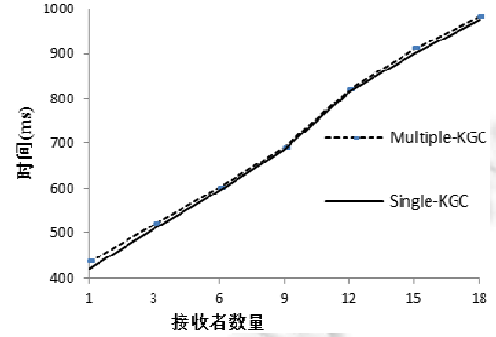

Fig.5 Execution time over the number of recipients

图 5 接收者数量对应的执行时间

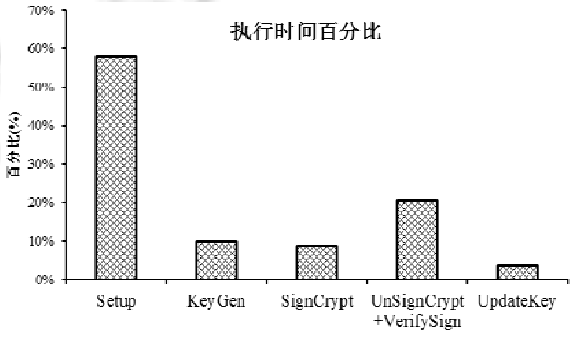

Fig.6 Percentage of execution time over each phase

图 6 各个阶段执行时间百分比

从图 5 可知,随着接收者数量的增加,方案总体执行时间会依次增加.这是由于接收者数量主要会影响用户密钥生成以及签密过程的执行时间.在排除网络延迟等因素干扰后,对于指定数量的接收者,引入多密钥分发中心后,相比较传统的单密钥分发中心,性能损耗控制在 5%以内,但却极大地增强了系统安全性.当接收者数量继续增加时,由于初始化和用户密钥生成操作只需要在系统初始化和用户加入时候执行一次,解签密步骤又是并发执行的,签密过程的执行时间占总体执行时间的份额会逐渐增加.因此,当接收者数量较大时,系统性能瓶颈之一是签密步骤,这也给后续的优化指出了一种可行的方向.

从图 6 可知,初始化步骤占用总体执行时间的百分比最多,接近 60%,但是初始化步骤只需要在系统启动时执行一次.签密步骤占用时间百分比大约为 10%,而解签密步骤占用的时间接近 20%.表 4 中给出的签密步骤时间复杂度为$(58n+29)T_m$,而解签密步骤为$203T_m$,签密步骤要比解签密步骤计算复杂度低 1 倍左右,这与仿真结果接近.对于给定的配置参数,密钥更新阶段占用的时间不到 5%,具有较高的执行效率.在实际场景中,密钥更新步骤可以由 KGC 异步执行,在系统空闲时,由 KGC 在后台异步更新密钥,以降低对系统可用性的影响.

8.2 基于区块链的密钥更新方案分析

在考察基于区块链的时间同步方案时,首先需要考察同步更新的及时性,也就是说,如果设定周期 T 为 30s,该同步方案能否在周期到来时及时触发同步操作.

区块链采用以太坊,采用的库为 Etherscan Ethereum Developer APIs,该库通过向 Etherscan 服务器端发送 GET/POST 请求来获得以太坊公链信息,而且服务器端 1s 最多接收 5 次请求.

以太坊节点分为全节点和轻节点,这两种节点均包含区块头信息.由于区块头包含了算法所要用到的区块高度以及时间戳信息,因此原则上这两种节点均能满足算法要求.在将本文算法应用在实际场景中时,可以有以下 4 种部署架构:第 1 种是 KGC 节点全部部署成全节点;第 2 种是 KGC 节点全部部署成轻节点;第 3 种是选择一些节点部署为全节点,KGC 节点访问这些全节点以获取区块信息;第 4 种是选择一些节点部署为轻节点,KGC 节点访问这些轻节点以获取区块信息.一般来讲,引入区块链后,这 4 种架构所耗费的额外存储资源和计算开销依次降低,但其对应的安全性也随之下降.因此在选择部署架构时,需要根据实际应用场景进行选择.

实验仿真部分选择了第 3 种架构进行部署,这里的 Etherscan 服务器相当于以太坊中的全节点.在某些计算资源紧缺的业务场景中,可以将某一台服务器部署为以太坊全节点,其他 KGC 服务器与其进行交互,组成一个安全域.当需要增加 KGC 时,尤其是需要异地部署多个 KGC 服务器时,可以通过部署多个安全域进行实现,这种部署架构具有良好的可扩展性.此外,这种部署方案可以只让全节点服务器对外网暴露,其他 KGC 服务器与该全节点服务器形成一个安全内网,实现内外网分离,以提升系统的安全性.最后,在这种部署方案中,以太坊的计算和存储开销只由全节点承担,可以减少其他 KGC 节点的资源消耗.虽然该全节点服务器有可能会成为安全瓶颈,但是在实际应用时,全节点服务器可以采用内外网分离等措施进行安全加固.此外,用户还可以根据不同应用场景下不同安全等级要求,从 4 种部署架构中选择一种部署方案.

由于以太坊的出块周期为 15s,所以实验设定的周期均为 15s 的倍数.给定周期 T,重复采样 25 次,相关实验结果如图 7 所示.对周期 T 对应的 25 次采样时间求解平均值 $\bar{t}$,并计算误差百分比$|T-\bar{t}|/T$,实验结果如图 8 所示.

从图 7 可知,当周期 T 为 15s 时,触发更新动作的时间在 1s~60s 范围内波动;当指定的周期 T 增加时,尤其是大于 50s 后,其触发时间均在$[T-50,T+50]$内波动.这说明在采用本文提出的周期更新方案时,需要能够容忍 50s 的时间误差.从图 8 可知,当周期 T 为 150s,误差百分比为 1.5%;随着设定周期 T 的增加,其误差百分比呈下降趋势;当 T 为 1 550s 时,误差低于 1%.根据上述分析可以推知:随着周期 T 的增加,其误差百分比还会继续下降.

需要说明的是,当 T=550s 时,其误差百分比不到 0.1%,明显低于两侧.之所以出现该现象,是由于当对周期 T=550s 连续采样 25 次并求解平均值 $\bar{t}$ 时,在这 25 次连续采样时间间隔中,可能由于以太坊公链网络通信状况较为良好等原因,导致以太坊产块较为及时,使得实验得到的误差百分比较 T=350s 和 T=750s 这两个时刻低.尽管如此,从理论上分析,当 T=350s 和 T=750s 时,允许的时间误差百分比最大应该为 50/350×100%=14%和

50/750×100%=6%,其误差百分比仍在理论范围内.

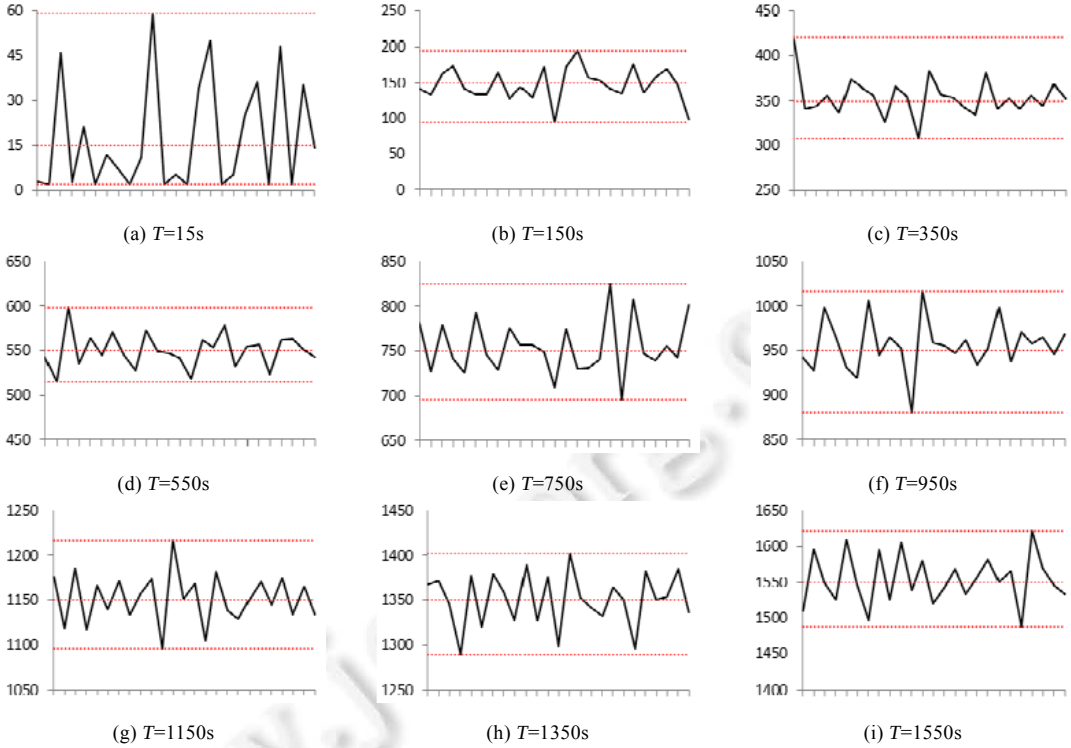

Fig.7 Result of updating the master key periodly based on blockchain

图 7 基于区块链的主密钥周期更新结果

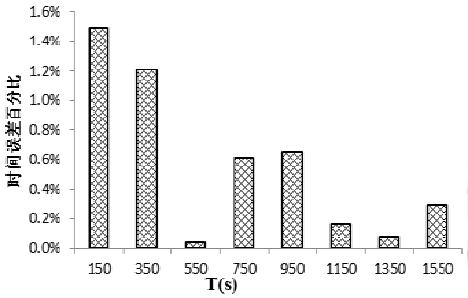

Fig.8 Percentage of time errors for updating the master key periodly

图 8 主密钥周期更新时间误差百分比

在实际中,虽然基于区块链的同步方案会有 50s 左右的时间误差,然而在应用到本文提出的签密方案中,这种随机误差会增加密钥更新过程的不确定性,使得攻击者很难预测密钥更新的准确时间,增大攻击者攻击的难度.此外,误差百分比随周期增加而下降,当周期大于 1 550s 后,误差百分比低于 1%.因此,上述更新策略能够有效地适配到新提出的签密方案中.

在将基于区块链的同步方案引入到签密方案时,还需要考察该同步方案对系统性能的影响,例如访问区块链带来的网络延迟、获取区块链数据后进行数据分析而带来的计算开销等.从算法 1 可知,引起访问区块链的操作主要有两种:第 1 种是 *eth.getblockNumber*(·),第 2 种是 *eth.getTimestamp*(·).第 1 种操作是访问区块链以获得最新的区块号,第 2 种操作是根据区块号以获得该区块对应的时间戳.获取到区块链数据后,数据为 JSON 格式,

需要在本地执行数据解析操作,与访问区块链操作相比较,解析数据耗费时间远小于前者耗费的时间,故后面不再对其进行分析.

下面评估前面所述的两种对区块链操作所耗费的时间,同样采用 Etherscan Ethereum Developer APIs,实验重复 50 次,其他配置参数与图 7 相同.*eth.getblockNumber*(·)操作耗时约为 350ms,而 *eth.getTimestamp*(·)耗时约为 740ms.从算法 1 可知,步骤 13~步骤 23 是同步方案中耗时最多的操作,其中,*sleep*(·)函数将会使程序进入到休眠状态,不会占用较多的计算资源.当执行密钥更新操作 *UpdateKey*(·)时,需要执行一次 *eth.getblockNumber*(·)和 *eth.getTimestamp*(·)操作,加起来约为 1s 左右.需要说明的是,这两种操作耗费的时间主要是等待区块链返回数据的时间,本地节点并不会耗费过多的计算资源.由前面论述可知,同步方案具有最大 50s 左右的时间误差,*eth.getblockNumber*(·)和 *eth.getTimestamp*(·)所耗费的时间远远小于前者,在实际场景中可以忽略不计.

在本文的仿真部分,KGC 并不是以太坊节点.但在实际应用时,可以将 KGC 搭建为以太坊节点,以充分使用计算资源.某一区块被矿工挖出后,会被矿工打上时间戳并发送出去.以太坊节点在接收该区块时,需要校验和存储该区块相关信息.从区块被挖出到指定的以太坊节点校验并成功存储该区块,在排除时间误差等干扰因素后,这一过程的时间延迟等于接收该区块的以太坊节点的本地时间与该区块的时间戳差值,本文将对该时间延迟进行考察.

由于在实际场景中,上述区块时间戳以及接收区块的以太坊节点本地时间均会存在一定的误差,因此实验重复 300 次,并求解时间延迟的平均值,以减少上述时间误差.基于 Parity-Ethereum 2.7.2 搭建以太坊轻节点,其他配置如下.

- Ubuntu:18.04;
- CPU:i5-7200U 2.50GHz;
- 网卡:1000Mb/s.

基于上述配置,仿真得到的时间延迟约为 18s,接近以太坊设定的出块周期 15s.

9 结束语

签密算法能够同时实现对消息安全性和认证性的校验,基于椭圆曲线提出一种多接收者多消息签密方案,能够有效适配到广播通信等应用场景中.采用多密钥分发中心管理系统主密钥信息,并能够周期更新各自的子秘密信息,以抵抗对应的 APT 攻击.新方案具有机密性、不可伪造性、密钥托管安全性、前后向兼容性、不可否认性.提出了一种基于区块链的周期更新策略,根据公有链中区块高度和块时间戳来触发密钥更新操作.签密方案具有较短的密文长度和较高的运行效率,基于区块链实现周期更新操作时,当周期大于 550s 时,时间误差百分比控制在 1%以内,并能够随周期的增加而下降.对于本方案,当接收者数量较多时,系统性能的瓶颈之一是签密步骤,后续将会对此进行优化.

References:

[1] Zheng YL. Digital signcryption or how to achieve *cost*(*signature* & *encryption*)≪*cost*(*signature*)+*cost*(*encryption*). In: Proc. of the Int'l Cryptology Conf. 1997. 165−179.

[2] Shamir A. How to share a secret. Communications of the ACM, 1979,22(11):612−613.

[3] Qiu J, Fan K, Zhang K, Pan Q, Li H, Yang YT. An efficient multi-message and multi-receiver signcryption scheme for heterogeneous smart mobile IoT. IEEE Access, 2019,7(1):180205−180217.

[4] Yu Y, Yang B, Huang XY, Zhang MW. Efficient identity-based signcryption scheme for multiple receivers. In: Proc. of the Int'l Conf. on Autonomic and Trusted Computing. 2007. 13−21.

[5] Li FG, Hu YP, Liu SG. Efficient and provably secure multi-recipient signcryption from bilinear pairings. Wuhan University Journal of Natural Sciences, 2007,12(1):17−20.

[6] Selvi SSD, Vivek SS, Srinivasan R, Rangan CP. An efficient identity-based signcryption scheme for multiple receivers. In: Proc. of the Int'l Workshop on Security. 2009. 71−88.

[7] Wu L. An ID-based multi-receiver signcryption scheme in MANET. Journal of Theoretical and Applied Information Technology, 2012,46(1):120−124.

[8] Peng C, Chen J, Obaidat MS, Pandi V, He DB. Efficient and provably secure multi-receiver signcryption scheme for multicast communication in edge computing. IEEE Internet of Things Journal, 2020,7(7):6056−6068.

[9] Pang LJ, Cui JJ, Li HX, Pei QQ, Jiang ZT, Wang YM. A new multi-receiver ID-based anonymous signcryption. Chinese Journal of Computers, 2011,34(11):2104−2113 (in Chinese with English abstract).

[10] Pang L, Wei M, Li H. Efficient and anonymous certificateless multi-message and multi-receiver signcryption scheme based on ECC. IEEE Access, 2019,7(1):24511−24526.

[11] Li HX, Chen XB, Ju LF, Pang LJ, Wang YM. Improved multi-receiver signcryption scheme. Journal of Computer Research and Development, 2013,50(7):1418−1425 (in Chinese with English abstract).

[12] Chen TS, Hsiao TC, Chen TL. An efficient threshold group signature scheme. In: Proc. of the IEEE Region 10 Conf. 2004. 13−16.

[13] Zhang Y, Xu CX, Ni JB, Li HW, Shen XM. Blockchain-assisted public-key encryption with keyword search against keyword guessing attacks for cloud storage. IEEE Trans. on Cloud Computing, 2019. [doi:10.1109/TCC.2019.2923222]

[14] Xie D, Li JJ, Shen ZH. A new threshold signature scheme based on elliptic curve cryptosystem. Journal of Hangzhou Normal University, 2013,12(1):57−60 (in Chinese with English abstract).

[15] Asmuth C, Bloom J. A modular approach to key safeguarding. IEEE Trans. on Information Theory, 1983,29(2):208−210.

[16] Cheng Y, Liu HP. The Asmuth-bloom verifiable threshold sharing scheme. Natural Sciences Journal of Harbin Normal University, 2011,27(3):35−38 (in Chinese with English abstract).

[17] Dang JL, Yu HF. Group signature scheme using Chinese remainder theorem. Computer Engineering, 2015,41(2):113−116 (in Chinese with English abstract).

[18] Liu HW, Xie WX, Yu JP, Zhang P. Efficiency identity-based threshold group signature scheme. Journal on Communications, 2009, 30(5):122−127 (in Chinese with English abstract).

[19] Yan J, Y XR, Zhang WJ. Research on group signature with threshold value based on elliptic curve. Journal of Southeast University (Natural Science Edition), 2008,38(1):43−46 (in Chinese with English abstract).

[20] Gennaro R, Jarecki S, Krawczyk H, Tai R. Robust threshold DSS signatures. In: Proc. of the Advances in Cryptology. 1996. 354−371.

[21] Gennaro R, Jarecki S, Krawczyk H, Tai R. Secure distributed key generation for discrete-log based cryptosystems. In: Proc. of the Int'l Conf. on Theory and Application of Cryptographic Techniques. 1999. 295−310.

[22] Steven G, Rosario G, Harry K, Bonneau J, Joshua A, Edward W, Arvind N. Securing Bitcoin Wallets via a New DSA/ECDSA Threshold Signature Scheme. 2015. http://stevengoldfeder.com/papers/threshold_sigs.pdf

[23] Zhou YW, Yang Bo, Zhang WZ. Multi-receiver and multi-message of certificateless signcryption scheme. Chinese Journal of Computers, 2017,40(7):1714−1724 (in Chinese with English abstract).

[24] Miao SQ, Zhang FT, Zhang L. Cryptanalysis of a certificateless multi-receiver signcryption scheme. In: Proc. of the Int'l Conf. on Multimedia Information Networking & Security. 2010. 593−597.

[25] Zhou YW, Yang Bo, Wang QL. Anonymous hybrid signcryption scheme with multi-receiver (multi-message) based on identity. Ruan Jian Xue Bao/Journal of Software, 2018,29(2):442−455 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5250.htm [doi: 10.13328/j.cnki.jos.005250]

[26] Jing Q, Jun B, Chuan SX, Hou SM. Secure and efficient multi-message and multi-receiver ID-based signcryption for rekeying in ad hoc networks. Journal of Chongqing University, 2013,12(2):91−96.

[27] Pang LJ, Man K, Wei MM, Li HX. Anonymous certificateless multi-receiver signcryption scheme without secure channel. IEEE Access, 2019,7(1):84091−84106.

[28] Gen LF, Xiong H, Yun NX. A new multi-receiver ID-based signcryption scheme for group communications. In: Proc. of the Int'l Conf. on Communications. 2009. 296−300.

[29] Bo Z, Liang XQ. An ID-based anonymous signcryption scheme for multiple receivers secure in the standard model. Int'l Journal of Advanced Science Technology, 2010,6059(20):15−27.

[30] Corp. C. SEC 2: Recommended elliptic curve domain parameters. Certicom Corporation, 2010. 1−33.

附中文参考文献:

[9] 庞辽军,崔静静,李慧贤,裴庆祺,姜正涛,王育民.新的基于身份的多接收者匿名签密方案.计算机学报,2011,34(11):2104−2113.

[11] 李慧贤,陈绪宝,巨龙飞,庞辽军,王育民.改进的多接收者签密方案.计算机研究与发展,2013,50(7):1418−1425.

[14] 谢冬,李佳佳,沈忠华.一种新的基于椭圆曲线的门限群签名方案.杭州师范大学学报(自然科学版),2013,12(1):57−60.

[16] 程宇,刘焕平.可验证的 Asmuth-Bloom 门限秘密共享方案.哈尔滨师范大学自然科学学报,2011,27(3):35−38.

[17] 党佳莉,俞惠芳.使用中国剩余定理的群签名方案.计算机工程,2015,41(2):113−116.

[18] 刘宏伟,谢维信,喻建平,张鹏.基于身份密码体制的高效门限群签名方案.通信学报,2009,30(5):122−127.

[19] 闫杰,尹旭日,张武军.基于椭圆曲线的带门限值的群签名研究.东南大学学报(自然科学版),2008,38(1):43−46.

[23] 周彦伟,杨波,张文政.无证书多接收者多消息签密机制.计算机学报,2017,40(7):1714−1724.

[25] 周彦伟,杨波,王青龙.基于身份的多接收者(多消息)匿名混合签密机制.软件学报,2018,29(2):442−455. http://www.jos.org.cn/1000-9825/5250.htm [doi: 10.13328/j.cnki.jos.005250]

王利朋(1987－),男,博士生,主要研究领域为密码学,区块链.

李青山(1977－),男,博士,主要研究领域为区块链.

高健博(1994－),男,博士,主要研究领域为区块链.

陈钟(1963－),男,博士,教授,博士生导师,CCF 会士,主要研究领域为区块链.