第 69 卷 第 1 期
2023 年 2 月

武汉大学学报(理学版)
J. Wuhan Univ. (Nat. Sci. Ed.)

Vol. 69 No. 1
Feb. 2023,51~59

DOI: 10.14188/j.1671-8836.2021.0308

多答案保护秘密共享协议

肖 健[1],杨 敏[1†],孟庆树[2]

1. 空天信息安全与可信计算教育部重点实验室 武汉大学 国家网络安全学院,湖北 武汉 430072;
2. 武汉天喻信息产业股份有限公司,湖北 武汉 430223

收稿日期:2021-11-08 † 通信联系人 E-mail:yangm@whu.edu.cn
基金项目:国家自然科学基金面上项目(62172308)
作者简介:肖 健,男,硕士生,现从事密码学、区块链方向的研究。E-mail:1339675740@qq.com

摘 要:针对目前基于口令解决区块链私钥丢失问题的方案中面临的口令遗忘和泄露的问题,提出一种基于密保问题的多答案保护秘密共享方案(multi-answer protected secret sharing,MAPSS)。该方案允许用户将一个秘密共享给多个服务器,并令所有服务器存储多个密保问题,之后用户仅需向部分(阈值)服务器提供部分(阈值)密保问题的答案就能重构该秘密。该方案不仅可以用于区块链私钥找回,还支持抗遗忘和泄露的找回策略;该方案无需公钥基础设施且高效。在随机预言机模型下证明了该方案的安全性基于 Threshold Parallel One-More Diffie-Hellman (TP-OMDH)假设,实现了该方案的系统原型,验证了该方案的实用性。

关 键 词:区块链私钥;密保问题;秘密共享;多答案保护秘密共享方案
中图分类号:TP309.7 **文献标志码**:A **文章编号**:1671-8836(2023)01-0051-09

Multi-Answer Protected Secret Sharing Protocol

XIAO Jian[1], **YANG Min**[1†], **MENG Qingshu**[2]

1. Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, Hubei, China;
2. Wuhan Tianyu Information Industry Co.,Ltd, Wuhan 430223, Hubei, China

Abstract: Aiming at solving the problem of password forgetting and leakage in the current password-based solution to the problem of blockchain private key loss, this paper proposed a Multi-Answer Protected Secret Sharing Scheme (MAPSS) based on secret problems. This scheme allows users to share a secret with multiple servers, and all servers store multiple secret questions. The scheme can not only be used to recover the private key of the blockchain, but also support the retrieval strategy against forgetting and leakage, and the scheme does not require public key infrastructure and is efficient. After that, the user only needs to provide a threshold number of answers to a threshold number of servers to reconstruct the secret. Finally, this paper not only proved that the security of the scheme is based on the Threshold Parallel One-More Diffie-Hellman (TP-OMDH) assumption under the random oracle model, but also implemented the system prototype of the scheme, which proved the practicality of the scheme.

Key words: blockchain private key; secret problems; secret sharing; multi-answer protected secret sharing scheme

0 引 言

区块链因具有去中心化、集体维护、公开透明、不可篡改、准匿名性等突出特点而被广泛应用于版权保护、溯源、取证、去中心化金融等领域。在区块链中,私钥是用户对区块链进行操作的唯一凭证。用户一般需要将私钥(或对应助记词)记在纸上保存或者借助脑钱包、各种软硬件钱包保存。这些保存方式的缺点是用户的私钥一旦被遗忘或随着钱包丢失就难以找回[1]。

引用格式:肖健,杨敏,孟庆树. 多答案保护秘密共享协议[J]. 武汉大学学报(理学版),2023,69(1):51-59. DOI:10.14188/j.1671-8836.2021.0308.
XIAO Jian, YANG Min, MENG Qingshu. Multi-Answer Protected Secret Sharing Protocol [J]. *J Wuhan Univ* (*Nat Sci Ed*), 2023, 69(1): 51-59. DOI: 10.14188/j.1671-8836.2021.0308(Ch).

而区块链去中心化的特点意味着传统的基于第三方的密钥恢复机制不适用于区块链私钥恢复。为了解决区块链私钥丢失难以找回的问题,学者们提出了许多基于口令保护区块链账户的方法[2~13]。但是口令作为低熵信息是一种较弱的认证方式,而用户通常会在不同系统中重复使用口令,这会导致外部泄露(数据库泄露)和口令猜测(离线字典攻击)等风险大大提升。即使某些用户出于安全的考虑在不同系统(尤其是涉及到财产或隐私相关的系统)中使用不同的口令,也会造成口令过多而遗忘的情况。

为了解决上述问题,一些解决私钥丢失问题的方法被提出。托管钱包[1]是一种使用第三方平台提供密钥托管服务的方案,用户仅需通过口令即可实现对密钥的操作。但是该方案存在单点失效、后门攻击等问题,且一旦托管服务器被攻破,大量密钥失窃将会造成严重的损失。口令保护钱包[2]通过用户口令对区块链私钥进行加密,提高了区块链可用性。但是这种方案不能抵抗口令的穷举攻击。Breuer等[3]提出的DZT(distributed zero-tester)允许用户提供口令让服务器产生可公开验证的身份证明,但该方案不能直接恢复出用户的私钥,仅可被用于实现区块链的通证转移,即智能合约可以利用公开参数和上述身份证明进行验证,并以此确定用户身份,将通证转移到备用账户。

PPSS(password-protected secret sharing)最早由Bagherzandi等[4]提出,其本质是一种在线门限方案。该方案具有如下性质:1) 安全性:任意t个以下服务器合谋不能得到任何关于秘密或口令的任何信息,也无法通过在线攻击猜出用户口令;2) 正确性:只有当用户输入正确的口令,才能重构正确的秘密;3) 鲁棒性:只要用户向至少t个诚实服务器发送正确的信息,总能从服务器得到正确的秘密。该方案是一种基于公钥基础设施(public key infrastructure, PKI)的方案,允许用户将一个秘密共享给n个服务器且用户仅凭口令就能够从$t(t<n)$个服务器中重构该秘密,用户在重构秘密之前需要对服务器的公钥证书进行验证。如果用户错误地使用了攻击者的公钥,那么攻击者可以使用私钥和离线字典攻击便破解口令。文献[5~7]提出了一些无需进行证书验证的方案,但需要较大的通讯和计算开销。Jarecki等[8]提出了ROPPSS(round-optimal PPSS)方案,该方案无需对服务器的公钥证书进行认证,用户和服务器只需单轮交互,服务器之间无需通讯。文献[9,10]提出了效率更优的方案,其中文献[10]提出的TOPPSS(threshold oblivious PPSS)计算开销较小,使用的关键技术是门限茫然伪随机数生成器(threshold oblivious pseudo-random function, T-OPRF),其安全性是基于随机预言机模型(random oracle model, ROM)的Threshold One-More Diffie-Hellman(T-OMDH)假设。目前的PPSS方案都仅支持单个口令,这样的设置缺乏灵活性且存在口令遗忘和泄露的风险。

Mackenzie等[11]提出的PAKE(password authentication key exchange)是一种与PPSS方案紧密相关的方案。该方案允许用户通过口令从一组服务器中重构私钥,并且在最新的方案[12,13]中,允许用户输入存在误差(不相同但相近)的口令重构私钥。虽然这种手段可以一定程度降低用户口令遗忘的风险,但是它会泄露口令的部分熵,降低了系统的安全性,并且仍然存在口令泄露的风险。

针对上述方案中口令遗忘带来的私钥无法恢复以及口令泄露导致的安全性问题,本文提出了一种新的密码学原语,多答案保护秘密共享(multi-answer protected secret sharing, MAPSS)方案。该方案通过多个密保问题的答案替代单一口令进行区块链私钥恢复。密保问题(又称秘密问题)是一种允许用户在服务器中预设多个问题及其答案,之后用户仅需提供部分正确答案即可通过服务器认证的辅助认证方式[14]。相关学者研究工作[14~16]表明,密保问题作为一种提示性回忆任务相较于口令具有更好的可记忆性。本文方案采用了PPSS方案的在线秘密共享技术,降低了密保答案泄露的风险且具有如下特点。

a) 鲁棒性:相较于单口令方案,即便用户遗忘某些答案也能进行秘密找回。

b) 安全性:相较于单口令方案,即便某些答案泄露,恶意攻击者也不能得到用户的秘密。

c) 灵活性:相较于单口令方案,用户可以更加灵活地设置恢复秘密的条件。

d) 高效性:无需PKI且是轮次最优的(用户和服务器只需进行单轮通讯,服务器之间无须通信)。

我们在随机预言机模型下证明了MAPSS的安全性是基于Threshold Parallel One-More Diffie-Hellman (TP-OMDH)假设的。

1 预备知识

1.1 Shamir Secret-Sharing

Shamir在文献[17]中提出了一种门限秘密共享

方案,它以门限值t、秘密s以及参与者数量n作为输入,通过选择一个随机$t-1$次多项式,将s分割为n个份额$\{s_i\}_{i\in[1,n]}$。任意t个不同份额$\{s_i\}_{i\in I}$(其中I为这t个份额的下标集合)可以重构出秘密s。上述随机多项式如下所示

$$q(x)=s+a_1x+a_2x^2+\cdots+a_{t-1}x^{t-1} \tag{1}$$

其中,$q(0)=s$为秘密值,$a_1,a_2,\cdots,a_{t-1}$为随机数;第i个份额$s_i=q(i)$。任意t个份额$\{s_i\}_{i\in I}$,可以通过下式计算得出秘密值

$$s=\sum_{i\in I}s_i\lambda_i \tag{2}$$

其中,λ_i为拉格朗日系数,计算式如下

$$\lambda_i=\prod_{j\in I,j\neq i}\frac{j}{j-i} \tag{3}$$

1.2　OPRF/T-OPRF

茫然伪随机数生成器(oblivious pseudo-random function,OPRF)是一种两方安全计算协议[12],由一个随机数生成器f和C/S协议构成。在该协议中,服务器拥有一个秘密k,用户在不知道k的情况下通过向服务器输入待度量值x来获得$f_k(x)$。在这一过程中用户不会得到k的任何信息,服务器也无法得到x和$f_k(x)$的任何信息。通过OPRF,用户可以将一个低熵口令pw输入给服务器获得一个伪随机密钥$f_k(\mathrm{pw})$。

OPRF的实现如图1所示,过程如下:首先,用户发送$a=H_1(\mathrm{pw})^r$给服务器,其中r为随机数,H_1:$\{0,1\}^*\to G$,H_2:$\{0,1\}^*\to\mathbb{Z}_m$为哈希函数。然后服务器计算$b=a^k$并发送给用户。最后,用户用随机数r盲化$H_1(\mathrm{pw})^k=b^{1/r}$,即可得到伪随机值$f_k(\mathrm{pw})=H_2\left(\mathrm{pw},H_1(\mathrm{pw})^k\right)$。

但是OPRF无法抵抗恶意服务器的穷举攻击,因为服务器可以通过遍历pw′,并用密钥k来直接计算不同的$f_k(\mathrm{pw}')$,从上层协议(如PPSS)中观测pw′是否猜测成功。为了应对这种攻击,Jarecki等在文献[10]中提出T-OPRF,使用门限的方法,让多个服务器共同管理密钥k。在使用了(t,n)门限的T-OPRF中,用户需要与t个服务器进行通信获得密钥。在该方案中任意不超过t个服务器合谋不能得到任何关于秘密或口令的任何信息,也无法通过在线攻击猜出用户口令。

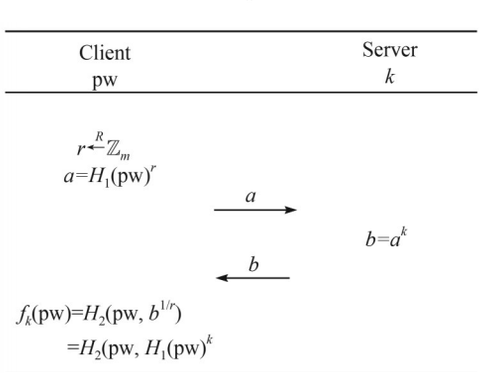

图1　茫然伪随机数生成器[12]

Fig. 1　OPRF[12]

T-OPRF的具体过程如图2所示。其中素数m是循环群G的阶;H_1:$\{0,1\}^*\to G$,H_2:$\{0,1\}^*\to\mathbb{Z}_m$为两个哈希函数;t,n为服务器阈值。初始化过程中,需要使用(t,n),Shamir Secret-Sharing将k分割为$\{k_i\}_{i\in[1,n]}$,每个服务器拥有私钥k_i。恢复密钥的过程中,用户需要与t个服务器组S进行通讯(S_i为对应服务器的序号)。该协议的安全性建立在ROM下的T-OMDH假设。

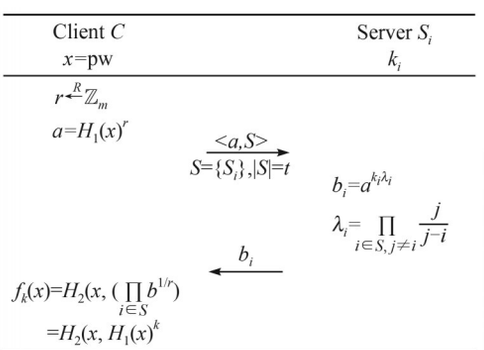

图2　门限茫然伪随机数生成器[3]

Fig. 2　T-OPRF[3]

值得注意的是,文献[10]指出,T-OPRF是目前实现PPSS的核心技术,当其作为工具应用到PPSS协议时,初始化过程一般不由服务器执行(比如使用DKG(distributed key generation)算法),而是由用户通过安全信道(例如:TLS流量)为各个服务器分发密钥份额k_i。这样的设置更符合PPSS作为密钥找回策略的实际应用场景。

2　MAPSS

MAPSS是一种使用密保问题的答案替代口令的秘密共享协议,该方案允许用户将一个秘密共享给n_S个服务器,并令所有服务器存储n_Q个密保问题,之后用户仅需向t_S($t_S<n_S$)个服务器提供t_Q($t_Q<n_Q$)个答案就能重构秘密(用户密钥)。同时,任意不超过t_S个服务器合谋不能得到任何关于秘密或答案的任何信息。任意不超过t_Q个答案泄

露不能得到最终的秘密,也不能得到其他答案的信息。该方案是鲁棒、安全、灵活和高效的,它不仅解决了区块链私钥丢失的问题,还能降低口令遗忘和泄露的风险。

2.1 方案构造

本文将T-OPRF与门限结合使得MAPSS能够一次性对一组用户输入进行度量,并最后取得一个固定的秘密。MAPSS方案由如下3个算法构成:

• Setup($1^l,t_S,n_S,t_Q,n_Q$)。初始化算法,该算法由用户执行。通过执行该算法建立系统参数。

1)用户首先选择安全参数l,群G为阶为素数m的循环群,g为生成元;

2)用户选择3个哈希函数满足$H_1:\{0,1\}^*\rightarrow G$,$H_2:\{0,1\}^*\rightarrow\{0,1\}^l$,$H_3:\{0,1\}^*\rightarrow\{0,1\}^l$;

3)用户指定服务器数量n_S,服务器数量阈值$t_S(t_S<n_S)$,密保问题数量n_Q,密保问题数量阈值$t_Q(t_Q<n_Q)$;

4)算法输出系统公共参数$PP=(G,m,g,t_S,n_S,t_Q,n_Q,l,H_1,H_2,H_3)$。

• KeyGen($k,q,Q=\{Q_i\}_{i\in1,\cdots,n_Q},X=\{X_i\}_{i\in1,\cdots,n_Q}$)。密钥生成算法,该算法由用户执行。通过执行该算法用户可以获得用户密钥rw。同时该算法还会产生用于密钥重构的信息,包括各个服务器密钥$k_i(i=1,\cdots,n_S)$以及可公开信息PI,这些信息将会保存在服务器中,用户无需保管。

1)用户密钥生成:用户选择随机数$q\leftarrow_R\mathbb{Z}_m$,计算得到用户密钥$rw=H_2(g,g^q)$;

2)服务器密钥生成:用户选择随机数$k\leftarrow_R\mathbb{Z}_m$,通过$(t_S,n_S)$Shamir Secret-Sharing将$k$分割为$k_i(i=1,\cdots,n_S)$作为各个服务器的密钥;

3)可公开信息生成:用户选择包含n_Q个密保问题的集合$Q=\{Q_i\}_{i\in1,\cdots,n_Q}$以及对应的答案集合$X=\{X_i\}_{i\in1,\cdots,n_Q}$。通过$(t_Q,n_Q)$Shamir Secret-Sharing将$q$分割为$q_i(i=1,\cdots,n_Q)$,然后计算一个大小为$n_Q$的集合$R=\{R_i=H_1(X_i)^k g^{q_i}\}_{i=1,\cdots,n_Q}$。最后,计算用户密钥的哈希值$C=H_3(rw)$。令可公开信息为$PI=\{Q,C,R\}$;

4)用户通过安全信道向第$i(i=1,\cdots,n_S)$个服务器发送并存储消息$Msg_i=(k_i,PI=\{Q,C,R\})$。

• RecKey($\{k_i\}_{i\in S,|S|=t_S},\{X_i'\}_{i\in I,|I|=t_Q},PI$)。密钥重构算法,该算法由服务器和用户一起执行。通过执行该算法可以重构用户密钥rw。一旦用户丢失密钥rw,用户只需要通过密保问题集合Q回忆任意t_Q个的答案,并与任意t_S个服务器组通讯即可恢复出密钥rw′。用户以及服务器的计算和通信过程如图3所示。具体过程如下:

1)用户首先根据可公开信息PI中的密保问题集合Q回忆任意t_Q个问题的答案集合$\{X_i'\}_{i\in I,|I|=t_Q}$($I$为对应的问题序号);

2)用户选择随机数$r\leftarrow_R\mathbb{Z}_m$并计算$a_i=H_1(X_i')^{\lambda_{I,i}}$和$a=\left(\prod_{i\in I}a_i\right)^r$,其中$\lambda_{I,i}$为拉格朗日系数,$\lambda_{I,i}=\prod_{j\in I,j\neq i}\frac{j}{j-i}$;

3)用户发送消息$Msg=(a,I,S)$任意t_S个服务器组(S为对应的服务器序号);

4)各服务器接收到消息后,验证$a\in G$是否成立。若成立则计算$b_i=a^{k_i\lambda_{S,i}}$和$M=\prod_{i\in I}R_i^{\lambda_{I,i}}$,其中$\lambda_{S,i}$为拉格朗日系数$\lambda_{S,i}=\prod_{j\in S,j\neq i}\frac{j}{j-i}$;否则算法输出错误;

5)各服务器发送消息$Msg=(b_i,M,C)$给用户;

6)用户收集到服务器组S的t_S个消息后,验证各个消息中的(M,C)是否相同。若相同则计算得到密钥$rw'=H_2(g,M/(\prod_{i\in S}b_i)^{1/r})$;否则算法输出错误;

7)用户验证$H_3(rw')=C$是否成立。若成立说明成功恢复密钥rw,算法输出正确;否则算法输出错误。

上述算法的正确性证明如下

$$M=\prod_{i\in I}R_i^{\lambda_{I,i}}=\prod_{i\in I}H_1(X_i)^{k\lambda_{I,i}}g^{q_i\lambda_{I,i}}=g^{\sum_{i\in I}q_i\lambda_{I,i}}\prod_{i\in I}H_1(X_i)^{k\lambda_{I,i}}=g^q\prod_{i\in I}H_1(X_i)^{k\lambda_{I,i}}\tag{4}$$

$$\left(\prod_{i\in S}b_i\right)^{1/r}=\left(\prod_{i\in S}a^{k_i\lambda_{S,i}}\right)^{1/r}=\left(a^{\sum_{i\in S}k_i\lambda_{S,i}}\right)^{1/r}=a^{\frac{k}{r}}=\prod_{i\in I}(a_i^{\lambda_{I,i}})^{\frac{k}{r}}=\prod_{i\in I}H_1(X_i')^{k\lambda_{I,i}}\tag{5}$$

由(4)和(5)式可知

$$rw=H_2\left(g,M\Big/\left(\prod_{i\in S}b_i\right)^{1/r}\right)=H_2\left(g,g^q\prod_{i\in I}H_1(X_i)^{k\lambda_{I,i}}\Big/\prod_{i\in I}H_1(X_i')^{k\lambda_{I,i}}\right)\tag{6}$$

若对于所有$i\in I$有$X_i'=X_i$,则

$$rw=H_2(g,g^q)\tag{7}$$

注1 MAPSS算法的关键在于,初始化过程中定义了一个公开集合$R=\{R_i=H_1(X_i)^k g^{q_i}\}_{i=1,\cdots,n_Q}$,其

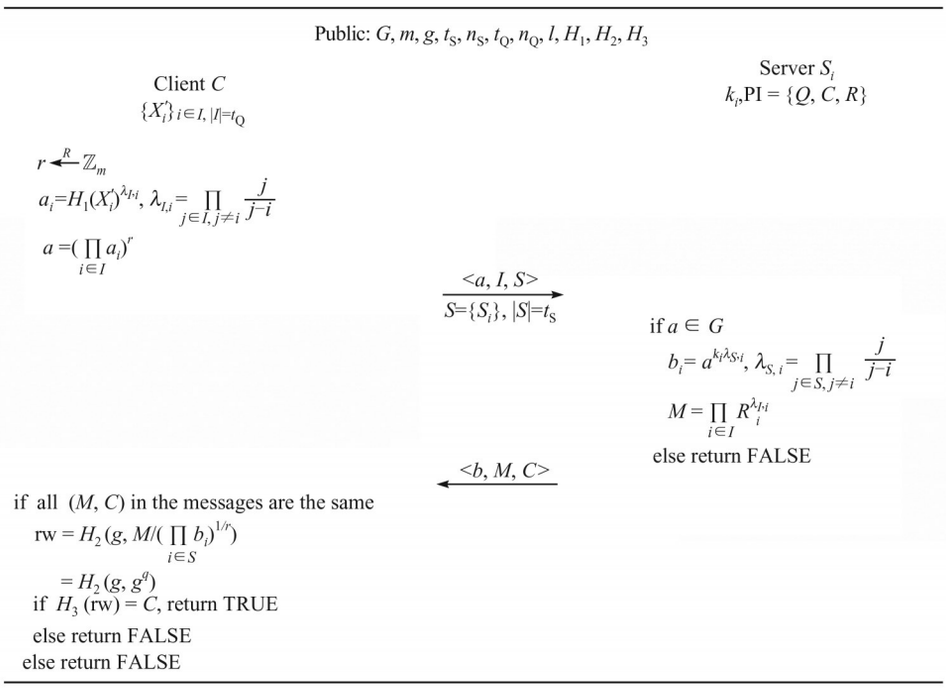

图 3　密钥重构算法

Fig. 3　Reconstructed key algorithm

中 g^{q_i} 为用户密钥的份额，$H_1(X_i)^k$ 为伪随机密钥。在重构密钥过程中，用户如果回忆起答案 X_i，那么可以通过执行 T-OPRF 算法来重新获得 $H_1(X_i)^k$，从而从 R 中提取对应的用户密钥份额 $g^{q_i}=R_i/H_1(X_i)^k$。理论上，只要用户能够回忆起 t_Q 个答案，执行 t_Q 轮 T-OPRF，就能获得 t_Q 个用户密钥份额，即可重构用户密钥 rw。而 MAPSS 是轮次最优的，这意味着 MAPSS 事实上在仅仅 1 轮 T-OPRF 就完成了密钥重构。

2.2　在区块链中应用

在区块链私钥找回的实际应用中，密钥 rw 有两种使用方法：1）用来作为注册区块链账户的私钥，2）将该密钥与区块链账户的私钥绑定。假设某个区块链账户对应的公私钥对为（pk，sk），用户可以简单计算 rec_string = sk⊕rw 作为找回私钥的字符串并通过智能合约发起交易上链存储。不论用户使用上述的何种方式，只要用户通过口令再次得到密钥，即可找回对应的区块链账户私钥。

在实际应用中，MAPSS 方案通过密保问题的设置赋予了用户更自由的策略选择。例如当用户将密保问题设置为“口令是什么？”，那么该方案可以从某种程度上视为单口令的 PPSS 方案。而当用户设置多个密保问题时，该方案在保证高效的同时，选择合适的参数(2.3 节)会提供更好的安全性和鲁棒性。

相较于 PPSS 方案，MAPSS 方案可以提供更好的安全性。在 PPSS 方案中，口令是用户从服务器中重构私钥的唯一凭证。因此用户口令一旦泄露，攻击者就能通过该口令向服务器发起恢复申请，从而盗取用户的区块链私钥。而在 MAPSS 方案中，用户不但可以使用多个密保问题的答案作为重构私钥的凭证，还可以指定一个阈值，只有提供不少于阈值个数密保问题的答案才能从服务器中重构出私钥。这意味着，只要用户不泄露等于或大于阈值个数密保问题的答案，攻击者就无法从服务器中重构出私钥。

同样地，MAPSS 方案相较于 PPSS 方案提供了更好的鲁棒性。由于口令的可记忆性较差，因此在 PPSS 方案中用户一旦将口令遗忘，就将无法从服务器中恢复出私钥。而在 MAPSS 中，密保问题是一种记忆性更好的信息，而且用户可以通过设定阈值来允许一定程度的遗忘，只要用户仍能提供不少于阈值个数密保问题数答案，就能再次获得区块链私钥。

2.3　安全性分析

本节对 MAPSS 的方案的安全性进行证明。将 MAPSS 方案的安全性定义为 TP-OMDH 假设，并证明该假设可以归约到文献[10]提出的 T-OMDH 假设。

令 $[n]$ 表示为集合 $\{1,\cdots,n\}$，I_w 为集合 $[n]$ 的大小为 w 的子集集合，如：$I_w=\{I\subseteq[n], \text{s.t.}\,|I|=w\}$。

令 V_w 表示汉明权重为 w 的 n 维的 0-1 向量的集合(即共有 w 个维度为 1),如:$V_w=\{\boldsymbol{v}=(v_1,\cdots,v_n),v_i\in\{0,1\},\text{s.t.}\ v_i=1\ \text{iff}\ i\in I_w\}$。对于任意一个 n 维向量 $\boldsymbol{q}=(q_1,\cdots,q_n)$,定义 $C_w(\boldsymbol{q})$ 为满足如下条件的最大整数 m:$\boldsymbol{v}_1,\cdots,\boldsymbol{v}_m\in V_w$ 且 $\boldsymbol{v}_1+\cdots+\boldsymbol{v}_m\leqslant\boldsymbol{q}$(注意 $\boldsymbol{v}_i$ 和 $\boldsymbol{v}_j$ 可能相同$(i\neq j)$)。

事实上,在下文中 V_w 代表一次性访问 n 个服务器中的 w 个的所有可能策略的集,I_w 是策略所对应的服务器序号的集合。例如当 $n=3,w=2$ 时 $\boldsymbol{v}=[1,0,1]\in V_w,I=[1,3]\in I_w$。在向量 $\boldsymbol{q}=(q_1,\cdots,q_n)$ 中,q_i 表示想要访问第 i 个服务器的次数。$C_w(\boldsymbol{q})$ 表示为了尽可能满足向量 $\boldsymbol{q}=(q_1,\cdots,q_n)$ 中对各个服务器的访问次数要求,需要执行多少次 V_w 中策略。例如:$\boldsymbol{q}=[2,3,3]$,那么 $C_2(\boldsymbol{q})=4$,因为 $C_2(\boldsymbol{q})=[1,0,1]+[1,1,0]+[0,1,1]+[0,1,1]$。

在 T-OMDH 假设中,群 G 为阶为素数 m 的循环群,生成元为 g,秘密 k 通过 $t-1$ 次多项式 $p(\cdot)$ 秘密共享给 n 个服务器,每个服务器拥有份额 $k_i=p(i)(1\leqslant i\leqslant n)$,并且需要对于参数为 $(i,a)\in[n]\cdot G$,计算 $a^{p(i)}$ 作为回应。在 T-OMDH 假设中,$\text{TOMDH}_p(\cdot,\cdot)$ 为服务器相对应的预言机,它以 (i,a) 为输入时,输出结果 $b_i=\text{TOMDH}_p(i,a)=a^{k_i}$,$q_i$ 表示访问 $\text{TOMDH}_p(i,\cdot)$ 的次数。

在 TP-OMDH 假设中,每个服务器拥有 $k_i=p(i)(1\leqslant i\leqslant n)$,且要对参数为 $\left(i,\{a_j\}_{j\in I}\right)\in[n]\cdot G^{t_Q}$,计算 $\left(\prod_{j\in I}a_j\right)^{p(i)}$ 作为回应,其中 $I\in I_{t_Q}$。在 TP-OMDH 假设中,$\text{TPOMDH}_p(\cdot,\cdot)$ 为服务器相对应的预言机,它以 $\left(i,\{a_j\}_{j\in I}\right)$ 为输入时,输出结果 $\text{TPOMDH}_p\left(i,\{a_j\}_{j\in I}\right)=(\prod_{j\in I}a_j)^{k_i}$。$q'_i$ 表示对于 $\text{TPOMDH}_p(i,\cdot)$ 的所有询问中,不同的 a_j 出现的数量。例如:对于第 1 个服务器有且只有两个询问

$$\text{TPOMDH}_p\left(1,\{a_1,a_2,a_3\}\right)=(\prod_{j=1}^{3}a_j)^{k_1}$$

$$\text{TPOMDH}_p\left(1,\{a_1,a_3,a_5\}\right)=(a_1a_3a_5)^{k_1}$$

由于出现了 4 个不同的 a_j,那么可知 $q'_1=4$。

T-OMDH/TP-OMDH 通用场景:在通用场景中,T-OMDH/TP-OMDH 假设意味着有 $t'(t'<t_S)$ 个拥有 $k_i=p(i)$ 的份额服务器合谋,因此敌手应知道这些份额。

定义 1 T-OMDH 称 (t',t_S,n,N,Q)T-OMDH 假设在阶为素数 m 的循环群 G 中成立,如果对于任意集合 $B\subseteq[n],|B|=t'<t_S$ 以及任意多项式时间,敌手 A 赢得下述游戏的概率是不可忽略的:A 收到一组挑战 $R=\{g_1,\cdots,g_N\}$,其中 $g_i\leftarrow_R G(1\leqslant i\leqslant N)$。A 可以访问预言机 $\text{TOMDH}_p(\cdot,\cdot)$,其中 $p(\cdot)$ 为 $\mathbb{Z}_m$ 上的 t_S-1 次多项式,并且对于任意 $j\in B$,A 知道份额 $p(j)=k_i$。如果 A 能给出 $Q+1$ 个 $(g_j)^k$ 且 $C_{t_S-t'}(q_1,\cdots,q_n)\leqslant Q$,其中 $k=p(0),g_j\in R$,对于 $i\in B$ 令 $q_i=0$,则称 A 赢得了游戏。

定义 2 TP-OMDH 称 (t',t_S,n,t_Q,N,Q)TP-OMDH 假设在阶为素数 m 的循环群 G 中成立,如果对于任意集合 $B\subseteq[n],|B|=t'<t_S$ 以及任意多项式时间敌手 A 赢得下述游戏的概率是不可忽略的:A 收到一组挑战 $R=\{g_1,\cdots,g_N\}$,其中 $g_i\leftarrow_R G(1\leqslant i\leqslant N)$。A 可以访问预言机 $\text{TPOMDH}_p(\cdot,\cdot)$,其中 $p(\cdot)$ 为 $\mathbb{Z}_m$ 上的 t_S-1 次多项式,并且对于任意 $j\in B$,A 知道份额 $p(j)=k_i$。如果 A 能给出 $Q+1$ 个 $(g_j)^k$ 且 $C_{t_S-t'}(q'_1,\cdots,q'_n)\leqslant Q$,其中 $k=p(0),g_j\in R$,对于 $i\in B$ 令 $q'_i=0$,则称 A 赢得了游戏。

定理 1 对于任意 $t'<t_S,1\leqslant t_Q\leqslant N$,$(t',t_S,n,t_Q,N,Q)$TP-OMDH 和 (t',t_S,n,N,Q)T-OMDH 是等价的。

证 如果敌手 A 能够破坏 (t',t_S,n,t_Q,N,Q)TP-OMDH 假设,并且询问次数满足 $C_{t_S-t'}(q'_1,\cdots,q'_n)\leqslant Q$。那么另外一个敌手 R 可以通过如下方式来破坏 (t',t_S,n,N,Q)T-OMDH:

对于一组发给 R 的挑战 $C=\{g_1,\cdots,g_N\}$ 并限制 R 对于预言机 $\text{TOMDH}_p(\cdot,\cdot)$ 查询次数满足 $C_{t_S-t'}(q_1,\cdots,q_n)\leqslant Q$,同样地,若 $i\notin B$ 则 q_i 表示对于 $\text{TOMDH}_p(i,\cdot)$ 的查询次数,若 $i\in B$ 则 $q_i=0$。敌手 R 转发挑战 C 给敌手 A。

对于 A 对 $\text{TPOMDH}_p(\cdot,\cdot)$ 的任意 $\left(i,\{a_j\}_{j\in I}\right)$ 查询,其中 $I\in I_{t_Q}$,R 需要记录一个表项 $R_{\text{Asked}}=(i,a_*,b_{i,*})$,且对于每一个 (i,a_j),其中 $j\in I$,做出如下操作:1) 若 R_{Asked} 已经记录 (i,a_j),直接从记录表中获得 $b_{i,j}$。2) 若 $i\in B$,则直接使用份额计算 $b_{i,j}=(a_j)^{k_i}$。3) 否则向 $\text{TOMDH}_p(\cdot,\cdot)$ 进行一次询问,获得 $b_{i,j}=(a_j)^{k_i}$,随后将该 $(i,a_j,b_{i,j})$ 记录在表中。完

成所有查询之后,计算$\prod_{j\in I}b_{i,j}=\left(\prod_{j\in I}a_j\right)^{k_i}$,并将结果发送给A,作为A对TPOMDH$_p(\cdot,\cdot)$的$\left(i,\{a_j\}_{j\in I}\right)$查询的回复。在这个过程中,对于每一个$i\notin B$的查询$(i,a_j)$,R仅对TOMDH$_p(\cdot,\cdot)$询问一次,故$q'_i=q_i$,所以当A的查询满足$C_{t_S-t'}(q'_1,\cdots,q'_n)\leqslant Q$时,R的查询也满足$C_{t_S-t'}(q_1,\cdots,q_n)\leqslant Q$。

如果敌手A赢得了游戏,输出至少$Q+1$个$(g_j)^k$,那么R可以复制A输出,并以此来破坏(t',t_S,n,N,Q)T-OMDH。

3　性能分析与实验

本部分首先从理论上分析各个算法的复杂度,然后给出实现方案中硬件的配置和选取的参数并记录各个算法的实际运行时间。

1)各个算法的理论复杂度。虽然在本文之前PPSS的相关研究工作仅支持单口令,即不能一次性处理多个用户输入,但文献[10]提出的TOPPSS方案是可能通过多轮处理多个用户输入的,因此我们将多轮TOPPSS与MAPSS的性能对比(包括支持的输入大小、KeyGen算法复杂度、RecKey算法的复杂度、消息数量以及通信轮次),理论分析如表1所示,其中polynomial evaluation为多项式求值,exp为幂运算。从表1中可知,MAPSS协议在密钥重构阶段相比于多轮TOPPSS,在通信轮次、消息数量和用户计算量有明显的优势。

2)实验硬件配置和参数选取。在实际应用中,用户可以使用如电脑、智能设备或者第三方设备作为服务器。我们使用的设备配置如下:客户端配置为Win10 PC AMD R5-3550H CPU @ 2.1 GHz processor,16 GB DDR4-RAM;10台服务器的配置均为Win10 PC AMD R7-3700X CPU @ 3.59 GHz processor,16 GB DDR4-RAM。算法实现采用Python Crypto and Python 3.8。选择安全参数$l=512$,群G的阶m为1 024位素数。

表1　多轮TOPPSS与MAPSS理论性能对比分析

Table 1　Theoretical performance comparison analysis of Multi-Round TOPPSS and MAPSS

参数	Multi-Round TOPPSS[10]	MAPSS
输入大小	t_Q	t_Q
KeyGen复杂度	n_S(polynomial evaluation)+t_Q(exp)	n_S+n_Q(polynomial evaluation)+$2n_Q+1$(exp)
RecKey复杂度	Client:$2t_Q$(exp.),Server:t_Q(exp)	Client:t_Q+2(exp.),Server:t_Q+1(exp)
RecKey消息数量	$6t_St_Q$	$6t_S$
RecKey通信轮次	t_Q	1

文献[4~6]通过大量的数据调研和统计,给出了许多密保问题(包括常用密保问题、用户自定义问题以及口令等)在各种情况下的实际猜测概率和遗忘概率。本文的实现方案从上述研究成果中综合计算并选择了一些记忆性较高且猜测概率相对较小的密保问题,如表2所示。用户可以根据实际的需要从中选择任意数量的问题n_Q以及服务器数量n_S,然后指定其他参数包括阈值t_Q,t_S(这些参数不会影响方案的理论安全性,见2.3节,其中服务器和密保问题的最大数量受到实际部署时的约束)。

3)各算法实际运行时间。图4给出了实现方案中服务器数量和阈值在2~10时以及密保问题数量和阈值在范围2~10时,MAPSS各个算法的实际花费时间。从图4(a)和4(c)中可以得出,KeyGen算法的时间随着服务器数量n_S和密保问题数量n_Q的增长而增长,即呈现一次线性相关。这意味着虽然提高服务器问题数量和阈值会增加攻击者攻击服务

表2　可供选择的密保问题

Table 2　Recommendation Secret Questions　　%

可供选择的密保问题		被攻击者猜中的平均概率	被用户正确记忆的平均概率
名字	第一个朋友的名字 第一个老师的名字 第一个学校的名字	2.32	64.63
爱好	最喜欢的歌曲 最喜欢的书籍 最喜欢的食物	3.93	78.00
地名	出生地 高中学校	4.60	80.10
数字	第一个电话号码	2.06	28.90
用户自选秘密	口令	1.00	19.23

器的难度,但是用户进行密钥生成的时间也会随之增加。从图 4(b)得知服务器数量阈值 t_S 不会影响算法的运行时间。从图 4(d)可以得知,RecKey 算法在服务端和客户端的运行时间均与密保问题数量阈值 t_Q 的线性相关。这意味着虽然提高密保问题数量和阈值会增加攻击者猜测密保答案的难度,但是用户重构私钥所花费的时间也会随之增加。上述结论与表 1 的分析结果一致。

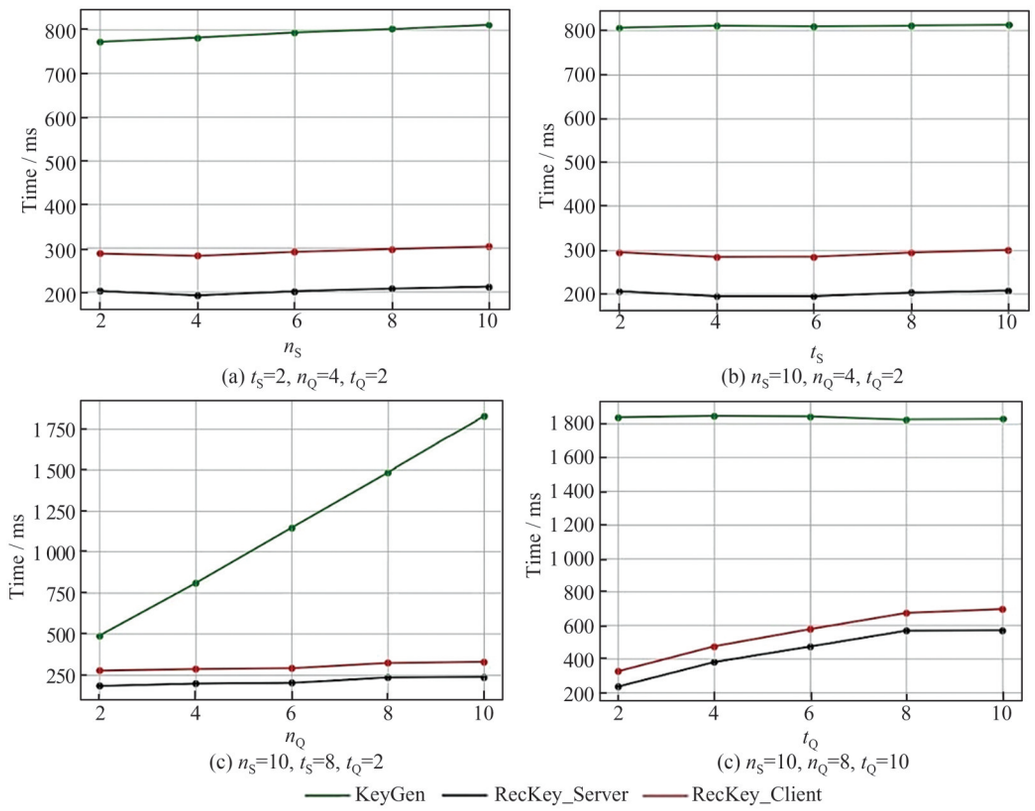

图 4 MAPSS 实际运行时间

Fig. 4 Performance evaluation of MAPSS

4 结 语

本文提出了一种新的密码学原语 MAPSS 协议,它允许用户使用 n_Q 个密保问题的任意 t_Q(t_Q<n_Q)答案来进行秘密重构。该协议具有鲁棒、安全、灵活和高效的特点,它不仅解决了区块链私钥丢失的问题,而且相较于一般仅支持单口令的 PPSS,多个密保问题的使用还能降低口令遗忘和泄露的风险。最后,我们不仅在安全模型下分析了原语的安全性,还通过实现该方案证明了该方案的实用性。

参考文献:

[1] 韩璇,袁勇,王飞跃.区块链安全问题:研究现状与展望[J].自动化学报,2019,**45**(1):206-225. DOI: 10.16383/j.aas.c180710.
HAN X, YUAN Y, WANG F Y. Security problems on blockchain: The state of the art and future trends [J]. *Acta Automatica Sinica*, 2019, **45**(1): 206-225. DOI: 10.16383/j.aas.c180710(Ch).

[2] GENNARO R, GOLDFEDER S, NARAYANAN A. Threshold-optimal DSA/ECDSA signatures and an application to bitcoin wallet security [J]. *International Conference on Applied Cryptography and Network Security*, 2016, **9696**: 156-174. DOI: 10.1007/978-3-319-39555-5_9

[3] BREUER F, GOYAL V, MALAVOLTA G. Cryptocurrencies with security policies and two-factor authentication [EB/OL]. [2021-03-29]. *https://eprint.iacr.org*/2021/416.

[4] BAGHERZANDI A, JARECKI S, SAXENA N, *et al*. Password-protected secret sharing [C]// *Proceedings of the 18th ACM Conference on Computer and Communications Security*. New York: ACM, 2011: 433-444. DOI: 10.1145/2046707.2046758.

[5] CAMENISCH J, LEHMANN A, LYSYANSKAYA A, *et al*. Memento: How to reconstruct your secrets from a single password in a hostile environment [C]// *Advances in Cryptology—CRYPTO* 2014. Heidelberg: Springer, 2014: 256-275. DOI: 10.1007/978-3-662-44381-1_15.

[6] YI X, HAO F, CHEN L Q, *et al*. Practical threshold password-authenticated secret sharing protocol [C]// Computer Security — *ESORICS* 2015. Cham: Springer International Publishing, 2015, **9326**: 347-365. DOI: 10.1007/978-3-319-24174-6_18

[7] YI X, TARI Z, HAO F, *et al*. Efficient threshold password-authenticated secret sharing protocols for cloud computing [J]. *Journal of Parallel and Distributed Computing*, 2019, **128**: 57-70. DOI: 10.1016/j.jpdc.2019.01.013.

[8] JARECKI S, KIAYIAS A, KRAWCZYK H. Round-optimal password-protected secret sharing and T-PAKE in the password-only model [C]// *Advances in Cryptology — ASIACRYPT* 2014. Heidelberg: Springer, 2014: 233-253. DOI: 10.1007/978-3-662-45608-8_13.

[9] JARECKI S, KIAYIAS A, KRAWCZYK H, *et al*. Highly-efficient and composable password-protected secret sharing (or: How to protect your bitcoin wallet online) [C]// 2016 *IEEE European Symposium on Security & Privacy*. New York: IEEE, 2016: 276-291. DOI: 10.1109/EuroSP.2016.30.

[10] JARECKI S, KIAYIAS A, KRAWCZYK H, *et al*. TOPPSS: Cost-minimal password-protected secret sharing based on threshold OPRF [J]. *International Conference on Applied Cryptography and Network Security*, 2017, **10355**: 39-58. DOI: 10.1007/978-3-319-61204-1_3.

[11] MACKENZIE P, SHRIMPTON T, JAKOBSSON M. Threshold password-authenticated key exchange [C]// *Advances in Cryptology — CRYPTO* 2002. Heidelberg: Springer, 2002: 385-400. DOI: 10.1007/3-540-45708-9_25.

[12] DUPONT P A, HESSE J, POINTCHEVAL D, *et al*. Fuzzy password-authenticated key exchange [C]// *Advances in Cryptology — EUROCRYPT* 2018. Cham: Springer International Publishing, 2018: 393-424. DOI: 10.1007/978-3-319-78372-7_13.

[13] ERWIG A, HESSE J, ORLT M, *et al*. Fuzzy asymmetric password-authenticated key exchange [C]// *Advances in Cryptology* — ASIACRYPT 2020. Cham: Springer International Publishing, 2020: 761-784. DOI: 10.1007/978-3-030-64834-3_26.

[14] BONNEAU J, BURSZTEIN E, CARON I, *et al*. Secrets, lies, and account recovery: Lessons from the use of personal knowledge questions at Google [C]// *Proceedings of the 24th International Conference on World Wide Web*. New York: ACM, 2015: 141-150. DOI: 10.1145/2736277.2741691.

[15] SCHECHTER S, BRUSH A, EGELMAN S. It's no secret: Measuring the security and reliability of authentication via "secret" questions [C]// *Proceedings of the 30th IEEE Symposium on Security and Privacy*. New York: IEEE, 2009: 375-390. DOI: 10.1109/SP.2009.11.

[16] POND R, PODD J, BUNNELL J, *et al*. Word association computer passwords: The effect of formulation techniques on recall and guessing rates [J]. *Computers & Security*, 2000, **19**(7): 645-656. DOI: 10.1016/S0167-4048(00)07023-1.

[17] SHAMIR A. How to share a secret [J]. *Communications of the ACM*, 1979, **22**(11): 612-613. DOI: 10.1145/359168.359176.

□