一个高效率的双指数多重签名方案

王文超 刘晋璐 秦 静

(山东大学数学学院 济南 250100)

摘 要 本文提出了一个高效率的双指数多重签名 DEMSP(Double Exponential Pairing-Based Multi-Signature)方案,在 DEMSP 方案中,每个签名者进行 BLS(Boneh-Lynn-Shacham)签名,在聚合公钥的计算上,DEMSP 方案比 MSP(Pairing-Based Multi-Signature with Public-Key Aggregation)方案高效数倍,倍数与签名者人数呈正相关. MSP 方案实现了签名聚合与密钥聚合,使得验证者验证一个消息是由 n 个签名者签名时,只需一个签名与一个公钥,并且 MSP 方案可以在普通公钥模型下抵抗流氓密钥攻击,然而 MSP 方案中,每个签名者在签名时都需要获得其他签名者的公钥,这增加了通信开销,在验证阶段,验证者计算聚合公钥时需要进行额外的指数计算. DEMSP 方案通过引入可追责第三方实现高签名效率与高验证效率,并利用分叉引理将 DEMSP 方案规约到 co-CDH 问题. DEMSP 方案应用于在线支付,使得交易合法,商家与用户的纠纷得以有效处理,并且 DEMSP 方案被扩展至多权威机构的多重签名方案. 同时 DEMSP 方案应用于 ASM(Accountable-Subgroup Multi-signature)方案及 MSDL(Discrete-Logarithm based Multi-Signature)方案,使得计算聚合公钥的时间比原方案减少数倍.

关键词 双指数多重签名方案;普通公钥模型;流氓密钥攻击;可追责第三方;分叉引理

中图法分类号 TP309 **DOI 号** 10.11897/SP.J.1016.2023.01213

An Efficient Double Exponential Multi-Signature Scheme

WANG Wen-Chao LIU Jin-Lu QIN Jing

(*School of Mathematics, Shandong University, Jinan* 250100)

Abstract We propose an efficient double exponential pairing-based multi-signature (DEMSP) scheme. In the DEMSP scheme, each signer only needs to perform the Boneh-Lynn-Shacham (BLS) signature. In the calculation of aggregate public keys, the DEMSP scheme is several times more efficient than the pairing-based multi-signature scheme with public-key aggregation (MSP). The efficiency is linear with the number of signers. The MSP scheme implements signature aggregation and key aggregation, so that the verifier only needs one signature and one public key when verifying that a message is signed by n signers. The MSP scheme can resist the rogue public-key attack under the plain public key model. However, in the MSP scheme, each signer needs to obtain public keys of other signers when signing, which increases the communication overhead. In the verification phase, the verifier needs to perform additional exponential calculations when calculating the aggregated public key. The DEMSP scheme improves the signature and verification efficiency of the MSP scheme by introducing the accountable third party, and we use forking lemma to reduce the DEMSP scheme to the co-CDH problem. The DEMSP scheme is applied to online payment to make transactions legal and disputes between merchants and users resolved. The DEMSP scheme has also been extended to multi-signature schemes with multiple authorities. At the same time, we apply the

收稿日期:2022-05-26;在线发布日期:2023-01-19. 本课题得到国家自然科学基金(62072276,61772311)资助. **王文超**,博士研究生,主要研究方向为公钥密码学. E-mail: 18406594347@163.com. **刘晋璐**,博士研究生,主要研究方向为云计算安全、公钥可搜索加密. **秦 静**(通信作者),博士,教授,博士生导师,主要研究领域为密码学、云计算安全. E-mail: qinjing@sdu.edu.cn.

DEMSP scheme to the accountable-subgroup multi-signature (ASM) scheme and the discrete-logarithm based multi-signature (MSDL) scheme. The time of calculating the aggregate public key is reduced several times.

Keywords double exponential pairing-based multi-signature scheme; plain public key model; rogue key attack; accountable third party; forking lemma

1 引 言

多重签名是由 Itakura 和 Nakamura[1]在 1983 年提出的,是一组签名者对同一个消息进行签名,并使得验证者确信每个签名者都参与了签名的方案.一种平凡的实现多重签名的方式是将多个签名连接起来,然后由验证者逐一验证,然而这种平凡的方法使得签名大小与验证时间与签名者的数量呈线性关系.多重签名应保证签名大小与验证时间尽可能小,并与签名者人数独立,与此同时签名时间也应保持不变.在很多应用中,需要在不同的公钥下对同一消息的多个签名提供高效的验证,例如涉及多类型或一对多的通信应用程序:IP 多播、点对点文件共享、移动自组织网络等.具体地,需要多重签名维护的消息一般都是黑客、罪犯和间谍机构所要攻击的高价值的目标,如网络权威机构的密钥,包括时间、名称、证书、软件更新服务等.然而使得多重签名方案引起广泛关注的是其在比特币技术中的应用,例如,为了增加事务的安全性,比特币中经常有需要多个密钥来授权的比特币交易.

多重签名中一个严重的攻击是流氓密钥攻击,敌手可以在未经其他签名者同意下生成与之共同签署的多重签名.这样的攻击摧毁了早期的协议[2],直到 Micali、Ohta 和 Reyzin[3]提出了一个正式的模型和一个可证明安全的方案,他们的方案要求敌手给出他们所选择的公钥所对应的私钥的知识的证明 KOSK(Knowledge of Secret Key),从而阻止了流氓密钥攻击.然而,他们的解决方案依赖于一个昂贵且不切实际的交互密钥生成协议,导致每个签名者的公钥大小依赖于签名的人数,签名需要提前确定签名者,并且签名之后不能加入新的签名者.另一种实现 KOSK 假设的方式是由 Yilek 等人[4]提出的基于 PKI(Public Key Infrastructure)的密钥登记模型 KR(Key Registration model),即引入可信第三方,由可信第三方来给所登记过的公钥出示私钥知识的证明,称这样的证明为密钥拥有证明 POP(a Proof of Possession of the secret key),但是签名者向可信第三方提供公钥所对应的私钥的知识的证明使得公钥的合法性由第三方验证,而不是多重签名的验证者验证,这限制了多重签名的使用,因此 Bagherzandi 和 Jarecki[5]提出了密钥可验证模型 KV(Key Verification model),在这个模型中,密钥能够证明隐含在公钥中且不需要可信第三方参与,之后可以被接受者验证,然而这样做增加了验证时间.

截止目前,较为实用的模型是 Bellare 和 Neven 在文献[6]中提出的普通公钥模型,即签名者不需要给出他们所选择的公钥所对应的私钥的知识的证明,在这个模型中,潜在的签名者集合是动态的,这些签名者可以根据自己的意愿选择自己的公钥,并可以在任何时候注册密钥,然而他们的方案没有实现密钥聚合,在文献[7]中,Maxwell 等人介绍了一种聚合技术,它允许将所有涉及的签名者的公钥聚合为一个公钥,他们的 MuSig(simple schnorr Multi-Signature)方案基于 Schnorr 签名方案,并在离散对数困难问题下证明了其安全性,MuSig 方案的签名大小与验证时间同单个签名者一样简单,但是 MuSig 方案是一个交互式协议,由签署人之间的 2 轮通信组成,随后,Boneh 等人[8]应用这种聚合技术,并使用基于双线性对的密码学提出了一个非交互式的多重签名方案 MSP 方案,然而 MSP 方案的验证效率低于 MuSig 方案,MSP 方案基于 BLS 签名[9],其安全性基于 Gap-Diffie-Hellman 假设,非交互性质通过使用双线性映射实现,这是一种特殊的密码操作,可以由友好配对的椭圆曲线[10]构造,此外,在文献[8]中 Boneh 等人提出了可以支持公钥聚合与交互式群设置的 ASM 方案;改进了由 Maxwell 等人[7]提出的基于 Schnorr 签名的 MusSig 方案,并提出了 MSDL 方案,在普通公钥模型中,MSP 方案、ASM 方案、MSDL 方案被证明是安全的.

目前,已有许多作者对上述聚合技术构造的方案展开了广泛的研究,MuSig 方案是两轮多重签名方案,然而 Drijvers 等人[11]指出在基于离散对数困难问题上的双轮多重签名方案在并行会话时易遭受

次指数复杂度的攻击，为了阻止这样的攻击，文献[11]提出了 mBCJ 方案，但该方案不能输出正常的 Schnorr 签名，Boneh 等人在文献[8]通过加入承诺，提出安全的三轮多重签名方案，之后，很多作者致力于实现安全的两轮多重签名方案，Nick 等人[12]提出了 MuSig-DN 方案，但是该方案防止并行攻击需要复杂的零知识证明，文献[13]证明了 DWMS(two-round schnorr Multi-Signatures via Delinearized Witnesses)方案在代数群与随机语言机的组合模型下基于 OMDL(One-More Discrete Logarithm)问题和 2-缠绕和问题的安全性，与此同时，文献[14]分别证明了 MuSig2 方案在随机语言机模型下的安全性与代数群和随机语言机的组合模型下的安全性，基于的困难问题是 AOMDL(Algebraic One-More Discrete Logarithm)问题，此外，这种聚合技术构造的方案有广泛的应用场景[15-18].

假设有 n 个签名者$\{S_1,\cdots,S_n\}$共同签署一个信息 m(如图 1)，令$\{pk_1=g_2^{sk_1},\cdots,pk_n=g_2^{sk_n}\}$为每个签名者的公钥，$\{\sigma_1,\cdots,\sigma_n\}$为每个签名者的签名. σ 为聚合签名，apk 为聚合公钥(如图 2). 实现多重签名的一种平凡的方式是将每个签名连接起来，然后逐一验证每个签名者的签名. 这种方式有两方面的缺点：

(1) 多个签名与多个公钥占用了大量的内存.

(2) 逐一验证每个消息的签名需要计算 n 个双线性对，n 是签名者数量.

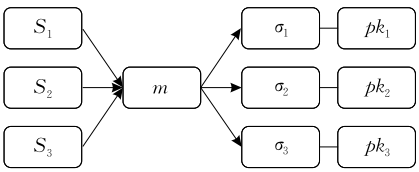

图 1　平凡的多重签名方案

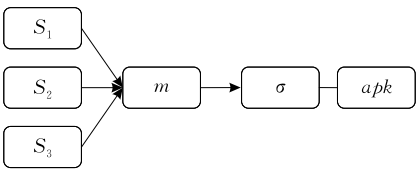

图 2　改进的多重签名方案

为实现高签名效率与高验证效率，我们构造一个高效率的 DEMSP 方案，其中 MSP 方案中的聚合技术在 DEMSP 方案中起到了关键作用，这里 MSP 方案中的聚合技术为

$$e(\sigma,g_2^{-1})\cdot e(H_0(m),apk)=1_{\mathbb{G}_t},$$

$$apk\leftarrow\prod_{i=1}^{n}pk_i^{a_i},$$

这里 $a_i\leftarrow H_1(pk_i,\{pk_1,\cdots,pk_n\})$，

$$\sigma\leftarrow\prod_{i=1}^{n}s_i,$$

这里 $s_i\leftarrow H_0(m)^{a_i\cdot sk_i}$，

Boneh 等人提出的 MSP 方案，一方面抵抗了流氓密钥攻击，另一方面实现了签名聚合与密钥聚合，使得签名大小与验证时间独立于签名者人数，但是每个签名者签名时需要获得其他签名者的公钥，增加了通信开销，此外，验证者计算聚合公钥 apk 需要计算 n 个哈希值 a_i 与指数 $pk_i^{a_i}$，使得验证时间变长，本文提出的 DEMSP 方案通过引入易被监督的可追责第三方去除了签名者与验证者的额外的哈希值与指数的运算，提升了 MSP 方案的签名效率与验证效率，对于签名效率，DEMSP 方案中的签名者不需要获得其他人的签名，减小了通信开销，对于验证效率，在 MSP 方案的验证过程中，验证者先计算聚合公钥，再进行验证，这里的聚合公钥需要计算额外的哈希值与指数，额外的计算提供了所有签名者的密钥拥有证明，而 DEMSP 方案将“添加哈希值与指数来提供密钥拥有证明”的聚合技术分为两步，首先利用这种聚合技术实现了可追责第三方对于所有签名者的密钥拥有证明，并且所有人容易验证可追责第三方是否进行了密钥拥有证明，然后，可追责第三方利用这种聚合技术生成一个签名，该签名可以证明多重签名确实是由可追责第三方参与生成，可追责第三方的可追责性体现在：

(1) 所有人可以验证可追责第三方是否对签名者的签名进行了密钥拥有证明.

(2) 多重签名只能由可追责第三方生成.

本文贡献如下：

(1) 构造了一个 DEMSP 方案，与 MSP 方案相比，DEMSP 方案在签名与验证时无需计算额外的哈希值与指数，DEMSP 方案降低了签名者的通信开销，签名者无需获得其他签名者的公钥就可以签名；减轻了验证者的计算负担，成倍提升了计算聚合公钥的效率.

(2) 将 DEMSP 的思想用到了在线支付中，确保了商家与用户的交易合法，处理了商家与用户可能发生的纠纷，保证了纠纷处理不当后的可追溯性，DEMSP 方案扩展至多权威机构多重签名方案，使得 DEMSP 方案的应用更加广泛.

(3) 为了拓广 DEMSP 方案的应用场景，将 DEMSP 方案推广至 ASM 方案与 MSDL 方案上[8]，使得计算聚合公钥与原方案相比少计算 $n-2$ 个哈希值与指数，n 是签名者人数.

2 预备知识

2.1 双线性对

令$\mathcal{G}$是一个双线性群生成器,以安全参数κ为输入,输出$(q,\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_t,e,g_1,g_2)$,这里$\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_t$是阶为素数$q$的群,$g_1,g_2$分别是$\mathbb{G}_1,\mathbb{G}_2$的生成元,$e$是一个有效的非退化的双线性映射$e:\mathbb{G}_1\times\mathbb{G}_2\rightarrow\mathbb{G}_t$,满足:

(1)双线性. 对于$\forall g_1\in\mathbb{G}_1$,$\forall g_2\in\mathbb{G}_2$,$\forall a,b\in\mathbb{Z}_q$,均有$e(g_1^a,g_2^b)=e(g_1,g_2)^{ab}$.

(2)非退化性. $\exists g_1\in\mathbb{G}_1$,$\exists g_2\in\mathbb{G}_2$使得$e(g_1,g_2)\neq 1_{\mathbb{G}_t}$.

(3)可计算性. 存在有效算法,对于$\forall g_1\in\mathbb{G}_1$,$\forall g_2\in\mathbb{G}_2$,均可计算$e(g_1,g_2)$.

2.2 计算困难问题

定义 1. 离散对数问题. 对于一个阶为素数q的群$\mathbb{G}=\langle g\rangle$,整数$x$满足$g^x=h$,其中$g,h\in\mathbb{G}$,并且它们不是单位元,称对$x$的求解为离散对数问题,定义敌手$\mathcal{A}$的优势$\mathrm{Adv}_{\mathbb{G}}^{\mathrm{dl}}$为

$$Pr[y=g^x:y\xleftarrow{\$}\mathbb{G},x\xleftarrow{\$}\mathcal{A}(y)].$$

这里的可能性取决于$\mathcal{A}$的随机选取,以及y的随机选择,如果敌手在至多时间τ内,攻破方案的优势为$\mathrm{Adv}_{\mathbb{G}}^{\mathrm{dl}}\geq\epsilon$,则称$\mathcal{A}(\tau,\epsilon)$攻破了离散对数问题,如果没有这样的敌手出现,称离散对数问题是(τ,ϵ)困难的.

定义 2. 计算型 co-Diffie-Hellman 问题. 对于一个阶为素数q的群$\mathbb{G}_1=\langle g_1\rangle$,$\mathbb{G}_2=\langle g_2\rangle$,给定$g_1$,$g_1^a\in\mathbb{G}_1$,$h\in\mathbb{G}_2$,称计算$h^a\in\mathbb{G}_2$为计算型 co-Diffie-Hellman 问题. 定义敌手的优势$\mathrm{Adv}_{\mathbb{G}_1,\mathbb{G}_2}^{\text{co-CDH}}$为

$$Pr[y=g_1^{\alpha\beta}:(\alpha,\beta)\xleftarrow{\$}\mathbb{Z}_q^2,y\leftarrow\mathbb{A}(g_1^\alpha,g_1^\beta,g_2^\beta)],$$

这里的可能性取决于$\mathcal{A}$的随机选取,以及(α,β)的随机选择. 这里定义的优势是合理的:

(1)$g_1^\alpha,g_1^\beta,g_2^\beta$是三个独立的数.

(2)由$g_1^\alpha,g_1^\beta,g_2^\beta$任意两个数求解$g_1^{\alpha\beta}$不比计算型 co-Diffie-Hellman 问题容易.

如果敌手在至多时间τ内,攻破方案的优势为$\mathrm{Adv}_{\mathbb{G}_1,\mathbb{G}_2}^{\text{co-CDH}}\geq\epsilon$,则称$\mathcal{A}(\tau,\epsilon)$攻破了 co-CDH 问题,如果没有这样的敌手出现,称 co-CDH 问题是(τ,ϵ)困难的.

2.3 广义分叉引理

在随机预言机模型中,通常使用 Pointcheval 和 Stern[19]的分叉引理来证明基于 Schnorr 签名[20]的方案的安全性,他们的引理后来被推广应用于更广泛的一类方案[6,21].

令$\mathcal{A}$是一个算法,输入in并与随机预言机$H:\{0,1\}^*\rightarrow\mathbb{Z}_q$. 令$f=(\rho,h_1,\cdots,h_{q_H})$是$\mathcal{A}$的随机输入,这里$\rho$是$\mathcal{A}$的随机带,$h_i$是$\mathcal{A}$的对于$H$的第$i$次问询,$h_i\in\overline{H}$,$\overline{H}$至少包含两个元素,$q_H$是询问随机预言机的最大次数,令$\Omega$是所有这样的$f$的空间,令$f|_i=(\rho,h_1,\cdots,h_{i-1})$,考虑$\mathcal{A}$的行为,输入$in$和随机的$f$,表示为$\mathcal{A}(in,f)$,如果输出$(J,\{out_j\}_{j\in J})$,则代表成功,其中$J$是$\{1,\cdots,q_H\}$的非空子集,$\{out_j\}_{j\in J}$是输出的子集,若$J=\varnothing$,称$\mathcal{A}$失败,对于新的随机的$f\xleftarrow{\$}\Omega$和由输入生成器 IG 输入的$in\xleftarrow{\$}\mathrm{IG}$,令$\epsilon$为$\mathcal{A}(in,f)$成功的概率.

对于给定的输入in广义分叉引理定义如下:

$\mathcal{GF}_{\mathcal{A}}(in)$:

$f=(\rho,h_1,\cdots,h_{q_H})\xleftarrow{\$}\Omega$

$(J,\{out_j\}_{j\in J})\leftarrow\mathcal{A}(in,f)$

如果$J=\varnothing$,则输出失败

令$J=\{j_1,\cdots,j_n\}$,这里$j_1\leq\cdots\leq j_n$

对于$i=1,\cdots,n$循环

$succ_i\leftarrow 0$;$k_i\leftarrow 0$;$k_{\max}\leftarrow 8nq_H/\epsilon\cdot\ln(8n/\epsilon)$

重复上述过程直到$succ_i=1$或者$k_i>k_{\max}$

$f''\xleftarrow{\$}\Omega$使得$f''|_{j_i}=f|_{j_i}$

令$f''=(\rho,h_1,\cdots,h_{j_i-1},h''_{j_i},\cdots,h''_{q_H})$

$(J'',\{out''_j\}_{j\in J''})\leftarrow\mathcal{A}(in,f'')$

若$h''_{j_i}\neq h_{j_i}$,$J''\neq\varnothing$,$j_i\in J''$,那么

$out'_{j_i}\leftarrow out''_{j_i}$;$succ_i\leftarrow 1$

如果$succ_i=1$对于所有的$i=1,\cdots,n$

那么输出$(J,\{out_j\}_{j\in J},\{out'_j\}_{j\in J})$,否则输出失败

Bagherzandi 等人证明了这个分叉算法的以下引理.

引理. 广义分叉引理[21]. 令 IG 是一个随机算法,$\mathcal{A}$是一个在时间τ内运行至多q_H次的成功率为ϵ的随机算法,若$q>8nq_H/\epsilon$,那么$\mathcal{GF}_{\mathcal{A}}(in)$在时间$\tau$内至多运行$\tau\cdot 8n^2q_H/\epsilon\cdot\ln(8n/\epsilon)$次,至少可以达到$\epsilon/8$的成功率,这里成功率取决于$in\xleftarrow{\$}\mathrm{IG}$与$\mathcal{GF}_{\mathcal{A}}$.

3 多重签名的定义以及安全模型

3.1 多重签名的定义

一个多重签名方案:Pg(参数生成)、Kg(密钥生成)、Sign(签名)、KAg(密钥聚合)、Vf(验证).

(1)Pg:可信方生成参数$par\leftarrow\mathrm{Pg}$.

(2)Kg:每个签名者生成一个密钥对$(pk_i,sk_i)\xleftarrow{\$}\mathrm{Kg}(par)$,$i=1,\cdots,n$.

(3)Sign:签名者输入par,PK,sk_i,m,这里PK

是所有签名者的公钥集合，sk_i 是签名者的私钥，最后，每个签名者输出一个签名 σ_i.

(4) KAg：输入 par, PK，输出聚合公钥 apk.

(5) Vf：验证者输入 par, apk, m, σ，其中 σ 为多重签名，检查在聚合公钥 apk 下消息 m 的签名 σ 的有效性，算法输出 1 则签名有效，否则无效.

3.2 多重签名的安全模型

完备性. 对于任意的 n，如果 $(pk_i, sk_i) \xleftarrow{\$} \mathrm{Kg}(par)$，$i=1,\cdots,n$，并且对于任何消息 m，所有的签名者输入 $\mathrm{Sign}(par, PK, sk_i, m)$，输出签名 σ_i，我们有 $\mathrm{Vf}(par, \mathrm{KAg}(par, PK), m, \sigma)=1$.

不可伪造性. 多重签名 $MSP=(\mathrm{Pg}, \mathrm{Kg}, \mathrm{Sign}, \mathrm{KAg}, \mathrm{Vf})$ 不可伪造性定义有以下几个阶段：

(1) 系统参数生成. 挑战者生成参数 $par \leftarrow \mathrm{Pg}$ 和一个挑战密钥对 $(pk^*, sk^*) \xleftarrow{\$} \mathrm{Kg}(par)$，挑战者将公钥 pk^* 发给敌手 $\mathcal{A}$.

(2) 签名问询. 允许敌手 $\mathcal{A}$ 对任意消息 m 进行签名查询，对于任意的签名者公钥集合 PK，这里 $pk^* \in PK$，敌手 $\mathcal{A}$ 可以访问随机预言机 $\mathcal{O}^{\mathrm{Sign}(par, \cdots, sk^*, \cdots)}$，模拟诚实签名者在一个签名协议中与 PK 的其他签名者交互，签名消息 m，这里 $\mathcal{A}$ 可以询问任意次随机预言机.

(3) 伪造. 最后敌手输出一个多重签名的伪造 σ，一个消息 m，一个公钥集合 PK，如果 $pk^* \in PK$，$\mathcal{A}$ 没有访问过 m，并且 $\mathrm{Vf}(par, \mathrm{KAg}(par, PK), m, \sigma)=1$，则称敌手成功伪造.

定义 3.　如果在时间 τ 内，查询 q_S 次签名问询，做 q_H 次随机预言机问询，以至少 ϵ 的概率赢得上述游戏，则称 $\mathcal{A}$ 成功对多重签名方案 $MSP=(\mathrm{Pg}, \mathrm{Kg}, \mathrm{Sign}, \mathrm{KAg}, \mathrm{Vf})$ 进行了一个 $(\tau, q_S, q_H, \epsilon)$ 伪造，如果没有一个 $(\tau, q_S, q_H, \epsilon)$ 伪造，则称 MSP 方案是 $(\tau, q_S, q_H, \epsilon)$ 不可伪造的.

4　多重签名方案

4.1 BLS 多重签名方案

下面介绍 BLS 签名方案及其聚合机制，该方案需要一种有效计算的非退化双线性对 $e: \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$，这里 $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t$ 是阶为素数 q 的群，哈希函数 $H_0: \mathcal{M} \to \mathbb{G}_1$，BLS 签名方案如下：

(1) 密钥生成：选择一个随机数 $sk \xleftarrow{\$} \mathbb{Z}_q$，令 $pk \leftarrow g_2^{sk} \in \mathbb{G}_2$，输出 (pk, sk).

(2) 签名 (sk, m)：输出 $\sigma \leftarrow H_0(m)^{sk} \in \mathbb{G}_1$.

(3) 验证 (pk, m, σ)：若 $e(\sigma, g_2)=e(H_0(m), pk)$，则输出“接受”，否则输出“拒绝”.

BLS 签名方案支持简单的签名聚合过程，给定 (pk_i, m_i, σ_i)，对于 $i=1,\cdots,n$，任何人都可以聚合签名 $\sigma_1,\cdots,\sigma_n \in \mathbb{G}_1$ 得到一个短小的签名 σ，

$$\sigma \leftarrow \sigma_1 \cdots \sigma_n \in \mathbb{G}_1 \tag{1}$$

为了验证聚合签名 $\sigma \in \mathbb{G}_1$，需要验证：

$$e(\sigma, g_2)=e(H_0(m_1), pk_1)\cdots(H_0(m_n), pk_n) \tag{2}$$

注意到验证者需要所有的 (pk_i, m_i) 对于 $i=1,\cdots,n$，当所有签名的消息都一样时 $(m_1=\cdots=m_n)$，等式(2)简化为一个更简单的形式，只需要两个对：

$$e(\sigma, g_2)=e(H_0(m_1), pk_1 \cdots pk_n) \tag{3}$$

而且，验证者只需要获得一个短的聚合公钥 $pk_1 \cdots pk_n \in \mathbb{G}_2$.

流氓密钥攻击. 式(1)中的简单签名聚合方法本身是不安全的，易遭受流氓公钥攻击.

敌手 A 执行流氓密钥攻击即敌手 A 在诚实方 B 不知情的情形下签署 A 与 B 的联合签名. 敌手 A 选择 $\alpha \xleftarrow{\$} \mathbb{Z}_q$，敌手 A 注册一个非法公钥 $pk_2 := g_2^{\alpha} \cdot (pk_1)^{-1} \in \mathbb{G}_2$，这里 $pk_1 \in \mathbb{G}_2$ 是诚实方 Bob 的公钥，敌手可以声称签名 $\sigma := H_0(m)^{\alpha}$ 是他和 Bob 一起对消息 $m \in \mathbb{M}$ 的签名，验证方案如下：

$$e(\sigma, g_2)=e(H_0(m)^{\alpha}, g_2)=e(H_0(m), g_2^{\alpha})$$
$$=e(H_0(m), pk_1 \cdot pk_2).$$

防御. 一般有两种方式抵抗流氓密钥攻击.

(1) 要求每个签名者给出公钥所对应的私钥的证明，然而这需要复杂的计算，在实践中很难执行.

(2) 要求每个签名者签名的信息不一样，但是由于现在所有的消息都是不同的，不能像等式(3)那样对一个公共消息 m 上的签名者进行公钥聚合.

4.2 MSP 方案

为了实现签名聚合与密钥聚合，又能抵抗流氓密钥攻击，Boneh 等人[8] 提出了基于双线性对的 MSP 方案.

参数生成. $\mathrm{Pg}(\kappa)$ 建立双线性群 $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, g_1, g_2)$，设哈希函数 $H_0: \{0,1\}^* \to \mathbb{G}_1$ 和 $H_1: \{0,1\}^* \to \mathbb{Z}_q$.

密钥生成. $\mathrm{Kg}(par)$ 选择一个随机数 $sk \xleftarrow{\$} \mathbb{Z}_q$，令 $pk \leftarrow g_2^{sk} \in \mathbb{G}_2$，输出 (pk, sk).

密钥聚合. $\mathrm{KAg}(pk_1,\cdots,pk_n)$ 输出

$$apk \leftarrow \prod_{i=1}^{n} pk_i^{a_i}.$$

这里 $a_i \leftarrow H_1(pk_i, \{pk_1,\cdots,pk_n\})$.

签名. 签名是一个单轮协议 $\mathrm{Sign}(par, \{pk_1,\cdots,pk_n\}, sk_i, m)$ 输出 $s_i \leftarrow H_0(m)^{a_i \cdot sk_i}$，将 s_i 发送给指定

的合成者，由合成者生成多重签名 $\sigma\leftarrow\prod_{j=1}^{n}s_j$，这个指定的合成者可以是签名者之一，也可以是外部方.

多重签名验证. $\mathrm{Vf}(par,apk,m,\sigma)$输出 1，当且仅当

$$e(\sigma,g_2^{-1})\cdot e(H_0(m),apk)=1_{\mathbb{G}_t}.$$

4.3 具有可信第三方的 MSP 方案

通过引入可信第三方提高 MSP 方案的签名效率与验证效率.

4.3 具体方案

参数生成. $\mathrm{Pg}(\kappa)$，建立双线性群$(q,\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_t,e,g_1,g_2)$，设哈希函数 $H_0:\{0,1\}^*\rightarrow\mathbb{G}_1$和 $H_1:\{0,1\}^*\rightarrow\mathbb{Z}_q$.

密钥生成. $\mathrm{Kg}(par)$选择一个随机数 $sk\xleftarrow{\$}\mathbb{Z}_q$，令 $pk\leftarrow g_2^{sk}\in\mathbb{G}_2$，输出$(pk,sk)$.

密钥聚合. $\mathrm{KAg}(PK=\{pk_1,\cdots,pk_n\})$输出

$$apk\leftarrow\prod_{i=1}^{n}pk_i.$$

签名. 每个签名者生成 BLS 签名 $\sigma_i=H_0(m)^{sk_i}$，并发送给可信第三方，可信第三方 $\mathrm{Sign}(par,\{\sigma_1,\cdots,\sigma_n\},\{pk_1,\cdots,pk_n\},m)$执行以下操作，若

$$e\left(\prod_{i=1}^{n}\sigma_i^{a_i},g_2^{-1}\right)\cdot e\left(H_0(m),\prod_{i=1}^{n}pk_i^{a_i}\right)=1_{\mathbb{G}_t},$$

其中$a_i\leftarrow H_1(pk_i,PK)$，输出$pk_1,\cdots,pk_n$是合法公钥.

由合成者计算多重签名 $\sigma\leftarrow\prod_{j=1}^{n}\sigma_j$，这里的合成者可以是任意一个签名者，也可以是外部方，否则终止协议.

多重签名验证. $\mathrm{Vf}(par,apk,m,\sigma)$计算 $apk\leftarrow\mathrm{KAg}(pk_1,\cdots,pk_n)$，输出 1 当且仅当

$$e(\sigma,g_2^{-1})\cdot e(H_0(m),apk)=1_{\mathbb{G}_t}.$$

4.3.2 分析

(1)具有可信第三方的 MSP 方案需要保证：

① 每个签名者的签名被判别.

② 多重签名由可信第三方参与生成.

(2)为了保证每个签名者的签名被进行了判别，可以将多重签名改为 $\sigma\leftarrow\prod_{i=1}^{n}\sigma_i^{a_i}$，然而这样的多重签名使得计算聚合公钥 apk 变得繁琐.

(3)于是我们提出了双指数多重签名方案(DEMSP)，该方案具有以下性能：

① 保证每个签名者的签名被判别.

② 多重签名由可追责第三方参与生成.

③ 聚合公钥 apk 容易被计算.

5 双指数多重签名方案(DEMSP)

为提升 MSP 多重签名的签名效率与验证效率，我们提出 DEMSP 方案.

5.1 具体方案

参数生成. $\mathrm{Pg}(\kappa)$，建立双线性群$(q,\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_t,e,g_1,g_2)$，设哈希函数 $H_0:\{0,1\}^*\rightarrow\mathbb{G}_1$和 $H_1:\{0,1\}^*\rightarrow\mathbb{Z}_q$.

密钥生成. $\mathrm{Kg}(par)$选择一个随机数 $sk\xleftarrow{\$}\mathbb{Z}_q$，令 $pk\leftarrow g_2^{sk}\in\mathbb{G}_2$，输出$(pk,sk)$.

密钥聚合. $\mathrm{KAg}(PK=\{pk_{11},pk_{12},pk_2,\cdots,pk_n\})$输出

$$apk\leftarrow pk_{11}^{a_{11}}\,pk_{12}^{a_{12}}\prod_{i=2}^{n}pk_i,$$

这里 $a_{1i}\leftarrow H_1(pk_{1i},PK)$.

签名. 下面用第一个签名者代替可追责第三方，除第一个签名者以外的其余签名者生成 BLS 签名 $\sigma_i=H_0(m)^{sk_i}$，并发送给第一个签名者，第一个签名者 $\mathrm{Sign}(par,\{\sigma_2,\cdots,\sigma_n\},PK,sk_{11},sk_{12},m)$执行以下操作，若

$$e\left(\prod_{i=2}^{n}\sigma_i^{a_i},g_2^{-1}\right)\cdot e\left(H_0(m),\prod_{i=2}^{n}pk_i^{a_i}\right)=1_{\mathbb{G}_t},$$

这里$a_i\leftarrow H_1(pk_i,\{pk_2,\cdots,pk_n\})$，则输出

$$\sigma_1\leftarrow H_0(m)^{a_{11}\cdot sk_{11}}\,H_0(m)^{a_{12}\cdot sk_{12}}.$$

由合成者计算多重签名 $\sigma\leftarrow\prod_{j=1}^{n}\sigma_j$，这里的合成者可以是任意一个签名者，也可以是外部方，否则终止协议.

多重签名验证. $\mathrm{Vf}(par,apk,m,\sigma)$计算 $apk\leftarrow\mathrm{KAg}(PK)$，输出 1 当且仅当

$$e(\sigma,g_2^{-1})\cdot e(H_0(m),apk)=1_{\mathbb{G}_t}.$$

5.2 安全证明

定理. 如果随机预言机模型下的计算型 co-CDH 问题是困难的，则 DEMSP 方案在 EU-CMA 安全模型下是一个不可伪造的多重签名方案，更精确的是，如果$q>8q_H/\epsilon$，并且 co-CDH 问题是$((\tau+q_H\cdot\tau_{exp_1}+q_S(\tau_{exp_1^2}+\tau_{exp_1})+\tau_{exp_2^2})\cdot 8q_H^2/\epsilon\cdot\ln(8q_H/\epsilon),\epsilon/8q_H)$困难的，则 DEMSP 在随机预言机模型下是$(\tau,q_S,q_H,\epsilon)$不可伪造的. 这里 τ_{exp_1}，τ_{exp_2}分别表示在$\mathbb{G}_1$，$\mathbb{G}_2$下计算一个指数所需要的时间，$\tau_{exp_1^i}$，$\tau_{exp_2^i}$分别表示在$\mathbb{G}_1$，$\mathbb{G}_2$下计算 i 个指数加上 n 个数相乘所需要的时间.

证明. 假定有一个(τ,q_S,q_H,ϵ)伪造者$\mathcal{F}$可以

在 EU-CMA 安全模型下攻破 DEMSP 方案.

系统参数生成. 考虑一个随机的输入生成器 IG 生成随机数组$(A,B_1,B_2)=(g_1^{\alpha},g_1^{\beta},g_2^{\beta})$,这里$(\alpha,\beta)\xleftarrow{\$}\mathbb{Z}_q$,算法$\mathcal{A}$输入$(A,B_1,B_2)$和随机的 $f=(\rho,h_1,h_2)$,执行以下操作.

哈希问询. 算法$\mathcal{A}$选择指数 $k\xleftarrow{\$}(1,\cdots,q_H)$,在随机带 ρ 上输入 $pk_{1i}\leftarrow B_2(i\in 1,2)$来运行伪造者$\mathcal{F}$,计算 $H_0(m)$:

$$H_0(m)=\begin{cases}g_1^{r_i}, & i\neq k\\ A, & i=k\end{cases}.$$

如果 $i\neq k$,算法$\mathcal{A}$使用 $r_i\xleftarrow{\$}\mathbb{Z}_q$计算 $g_1^{r_i}$,用 $g_1^{r_i}$回复$\mathcal{F}$的第 i 个问询,用 A 回复第 k 个 H_0问询,假定$\mathcal{F}$没有对 H_0做重复问询.

$\mathcal{A}$回复$\mathcal{F}$的 H_1问询如下:赋值 $H_1(pk_{1i},PK)\leftarrow h_i$,对于其他情形选择$\mathbb{Z}_q$中的随机值作为回复,对于已经问询过的公钥回复之前已经选好的值.

签名问询. 当$\mathcal{F}$对消息 m 在签名者集 S 进行了签名问询,$\mathcal{A}$计算 $apk\leftarrow\mathrm{KAg}(par,PK)$并查询 $H_0(m)$,如果是 A,则$\mathcal{A}$放弃并输出$(0,\perp)$,否则回复 g_1^r,此时$\mathcal{A}$可以模拟诚实的签名者,计算 $s_{1i}=B_1^r$.

伪造. 当$\mathcal{F}$没有成功地给出一个伪造时,$\mathcal{A}$输出$(0,\perp)$,如果$\mathcal{F}$成功地给出了一个消息 m 的伪造,并且 $H_0(m)\neq A$,则$\mathcal{A}$输出$(0,\perp)$,否则$\mathcal{F}$输出一个伪造(σ,PK,m)使得

$$e(\sigma,g_2)=e(A,\mathrm{KAg}(par,PK)),$$

令 j_f使得 $H_1(pk_{1i},PK)=h_{j_f}$,令 $apk\leftarrow\mathrm{KAg}(par,PK),a_{1i}\leftarrow H_1(pk_{1i},PK)$,对于 $PK=\{pk_{11},pk_{12},pk_2,\cdots,pk_n\}$,$\mathcal{A}$输出$(J=\{j_f\},\{(\sigma,PK,apk,a_{11},a_{12})\})$.

利用伪造签名解决 co-CDH 困难问题. 通过构造算法$\mathcal{B}$来证明这个定理,输入一个 co-CDH 例子$(A,B_1,B_2)\in\mathbb{G}_1\times\mathbb{G}_1\times\mathbb{G}_2$和一个伪造者$\mathcal{F}$来解决$(\mathbb{G}_1,\mathbb{G}_2)$上的 co-CDH 问题.也就是说,$\mathcal{B}$运行引理 1 中的广义分叉引理$\mathcal{GF}_{\mathcal{A}}$,并用算法$\mathcal{A}$输入$(A,B_1,B_2)$,这里 co-CDH 例子与 IG 的分布一样.

若$\mathcal{GF}_{\mathcal{A}}$输出$(0,\perp)$,则算法$\mathcal{B}$输出失败,如果$\mathcal{GF}_{\mathcal{A}}$输出$(\{j_f\},\{out\},\{out'\})$,则$\mathcal{B}$输出 out 为$(\sigma,PK,apk,a_{11},a_{12})$,输出 out'为$(\sigma',PK',apk',a'_{11},a'_{12})$.

从$\mathcal{GF}_{\mathcal{A}}$的构造知道 out 与 out'是由$\mathcal{A}$的两次执行完成的,由分叉引理的条件$(f|_{j_f}=f'|_{j_f})$,容易得到 $PK=PK'$,再次利用分叉引理可知 $a_{1i}\neq a'_{1i}$,$i\in\{1,2\}$,由$\mathcal{A}$的构造,我们知道

$$apk=pk_{1i}^{a_{1i}}pk_{1j}^{a_{1j}}\prod_{k=2}^{n}pk_k,$$

$$apk'=pk_{1i}^{a'_{1i}}pk_{1j}^{a_{1j}}\prod_{k=2}^{n}pk_k,$$

其中 $j\in\{1,2\}$,$j\neq i$,易知

$$apk/apk'=pk_{1i}^{a_{1i}-a'_{1i}}.$$

由此知道$\mathcal{A}$的输出满足

$$e(\sigma,g_2)=e(A,apk)\text{和 }e(\sigma',g_2)=e(A,apk'),$$

因此

$$e(\sigma/\sigma',g_2)=e(A,B_2^{a_{1i}-a'_{1i}}),$$

于是,可得$(\sigma/\sigma')^{1/a_{1i}-a'_{1i}}$是 co-CDH 问题的解.

时间复杂度. 敌手$\mathcal{A}$运行的时间是$\mathcal{F}$的运行时间加上$\mathcal{A}$的计算时间,令 q_H表示$\mathcal{F}$所做过的所有哈希问询,也就是说对 H_0,H_1的问询,$\mathcal{A}$需要$\mathbb{G}_1$上的一个指数运算来回复 H_0的问询,所以他至多花费 $q_H\cdot\tau_{exp_1}$去回复哈希问询,对于签名问询,$\mathcal{A}$花费时间 τ_{exp_1}计算$\mathbb{G}_1$上的一个哈希函数,花费 $\tau_{exp_1^2}$计算$\mathbb{G}_1$上的一个双指数多重签名,共花费时间为 $q_S(\tau_{exp_1^2}+\tau_{exp_1})$,最后$\mathcal{A}$花费 $\tau_{exp_2^2}$计算 apk,$\mathcal{A}$的运行时间总共为 $\tau+q_H\cdot\tau_{exp_1}+q_S(\tau_{exp_1^2}+\tau_{exp_1})+\tau_{exp_2^2}$,$\mathcal{A}$成功的可能性为$\mathcal{F}$成功的概率乘以他猜测$\mathcal{F}$的伪造的哈希指数的正确的概率 $1/q_H$,因此,$\mathcal{A}$成功的概率为 $\epsilon_{\mathcal{A}}=\epsilon/q_H$.

使用引理 1 知道,如果 $q>8q_H/\epsilon$,则$\mathcal{B}$至多运行$(\tau+q_H\cdot\tau_{exp_1}+q_S\cdot(\tau_{exp_1^2}+\tau_{exp_1})+\tau_{exp_1^2})\cdot 8q_H^2/\epsilon\cdot\ln(8q_H/\epsilon)$,成功的概率为$\epsilon'\geq\epsilon/8q_H$. 证毕.

5.3 分析

安全性. 上述的安全证明表明 sk_{11},sk_{12}是生成签名的必要条件,即多重签名在没有签名者一的参与下无法生成,签名过程中第一个签名者所做的判别式保证了签名 $\sigma_2,\cdots,\sigma_n$由 $sk_2,\cdots,sk_n$生成,其余的签名者无法执行流氓密钥攻击,因此 DEMSP 方案保证了多重签名的安全性,即 $sk_{11},sk_{12},sk_2,\cdots,sk_n$是生成多重签名的必要条件.

双签名者情形. 当方案只有两个签名者时,第一个签名者不需要对其他签名者的签名进行判别,此时,如果敌手正是第一个签名者,那么假定敌手使用错误的 pk_{11}或者 pk_{12},方案的安全性容易规约到 DEMSP 方案上,错误指敌手不知道公钥所对应的私钥,如果此时敌手使用了正确的 pk_{11}与 pk_{12},那么方案的安全性容易规约到有密钥拥有证明的情形,正确指敌手知道公钥对应的私钥,进一步,签名人数为 n 时,只需要任意 $n-1$ 个签名者在计算签名时额

外计算一个哈希值，和 MSP 方案相比提升了签名效率与验证效率.

在线钱包是多重签名一个重要的应用场景，客户掌握一个私钥，而网站掌握另一个私钥，只有这两把私钥同时签名才能够动用余额，保证了无论是网站被黑还是客户的电脑被黑，资金都无法被窃取，除非两边同时被同一个黑客攻破，与采用 MSP 方案中的签名相比，根据对于双签名者分析，客户在签名时无需计算额外的哈希值也可以实现安全的多重签名，减少了客户的通信开销，提升了一倍聚合公钥的计算效率.

多签名者情形. 考虑签名者数量为 $n(n\geq 3)$的情形，特别的考虑三个签名者，其中诚实方为签名者二，若 pk_3是正确的，方案的安全性规约到双签名者情形，若 pk_3是错误的，签名者一随机选择一个随机数 s_1构造 pk_3，$pk_3=g_2^{s_1}/pk_2'$，于是签名者一容易伪造签名 $\sigma=H_0(M)^{s_2}$（这里 $s_2=s_1+sk_{11}\cdot H_1(pk_{11},PK)+sk_{12}\cdot H_1(pk_{12},PK)$），即第一个签名者有能力执行流氓密钥攻击，但是我们容易对第一个签名者进行监督，因为 $\sigma_2,\cdots,\sigma_n,pk_2,\cdots,pk_n$都是公开的，任何人都可以去验证签名者一是否诚实的计算判别式. 验证方式如下：

首先，第一个签名者生成并保存 σ'，这里

$$\sigma'=\prod_{i=2}^{n}\sigma_i^{a_i},$$

这里 $a_i\leftarrow H_1(pk_i,\{pk_2,\cdots,pk_n\})$，其他人容易验证签名者一是否诚实的计算判别式，验证算法为

$$e(\sigma',g_2^{-1})\cdot e\left(H_0(m),\prod_{i=2}^{n}pk_i^{a_i}\right)=1_{\mathbb{G}_t}.$$

签名者一拥有两个私钥的必要性. 一方面分叉引理中需保证 $\overline{H}$ 至少包含两个元素，因此签名者一需要拥有两个私钥. 另一方面，敌手通过签名者一所返回的多重签名 $\sigma_1\leftarrow H_0(m)^{a_1\cdot sk_1}$，其中 $a_1\leftarrow H_1(pk_1,\{pk_1,\cdots,pk_n\})$，容易计算得到 $H_0(m)^{sk_1}$，利用 $H_0(m)^{sk_1}$敌手可以执行流氓密钥攻击，使得验证者误认为签名者一未执行判别式. 进一步可追责第三方不能对不同的多签名方案签署同一个消息，否则敌手容易得到该消息的签名，并利用签名去进行流氓密钥攻击，使得验证者误认为签名者一未执行判别式.

6 DEMSP 方案的扩展

DEMSP 方案具有高签名效率，高验证效率，并实现了签名聚合与密钥聚合，该方案引入的可追责第三方易受监督，并且在出现安全问题时，可向第三方追责，这里的安全问题指敌手伪造的多重签名通过验证. 下文对 DEMSP 方案进行推广，分别是对可追责第三方的推广，对 ASM 方案的改进与对 MSDL 方案的改进.

可追责第三方是 DEMSP 方案的重要组成部分，本文根据可追责第三方的数量给出了两方面的分析，分别为在线交易，多可追责第三方的多重签名. ASM 方案指由多个签名者中的指定的部分签名者签名的方案，对于 ASM 方案的改进主要体现在签名之前的预处理阶段，也就是将 DEMSP 方案的思想应用到了群设置阶段，提高了群设置阶段的效率，对于 MSDL 的改进是自然的，MSDL 方案与 MSP 方案的区别体现在使用的数学源语不同，但其中的很多构造方法可以类比使用，因此将 DEMSP 方案的思想应用到了 MSDL 方案.

6.1 对可追责第三方的扩展

(1) 单可追责第三方. DEMSP 方案是只有一个可追责第三方的方案，一方面 DEMSP 方案签名效率与验证效率高，另一方面可追责第三方的引入使 DEMSP 方案有更多的功能，比如在在线交易中可追责第三方可以监督交易合法，处理商家与用户的矛盾.

在线交易的参与方有用户，商家，支付宝，交易方式为支付宝充当中介，用户先将钱交给支付宝，商家发货，用户收到货后，确认收货，商家收款，若物品有所损坏，商家与用户达成一致，商家退款，若产生了矛盾，由支付宝仲裁. 这里支付宝充当了可信第三方的角色，然而这样的中介平台需要保证它不能随意动用这笔钱，更不能卷款跑路.

相比支付宝，比特币的中介担保交易限制了中介者作恶的能力，在多重签名的支持下，用户、商家、中介这三者形成“2/3”，如果 2 人交易原本就能够成功，那么无需中介者的加入，只要用户和商家的 2 个私钥确认了交易，那么整个交易就完成了，而只有在用户和商家出现分歧时，中介者才有了权力，不管中介者最后是跟谁达成一致，最终都会得到两个签名来解锁账户，然后把钱转给商家，或者是部分或全额退款给用户.

比特币与多重签名虽然解决了支付宝的信任问题，然而缺少了第三方后导致出现了许多违法犯罪的活动，采用 DEMSP 方案，引入一个可追责第三方，用户将钱放在一个虚拟的联合或托管账户中，可

追责第三方负责监督用户与商家的交易是否合法，交易只有在可追责第三方的参与下才可以进行，如果用户与商家的意见一致，无论是同意交易还是协商退货，那么交易正常进行. 当产生矛盾时，可追责第三方充当仲裁者的角色，可追责第三方可以在不经过其中一方的同意下进行流氓密钥攻击，产生合法签名，完成交易. 这里产生的签名是可追责的，即如果可追责第三方作恶，或者仲裁不合理，可以申诉并产生可追责第三方进行流氓密钥攻击的证据，仲裁不合理属实的话由可追责第三方强制商家退钱给用户或者强制用户赔偿商家损失. 因为所有人都容易监督可追责第三方，因此可追责第三方不是恶意的.

上述交易中的流氓密钥攻击流程如下，首先在用户和商家都有各自绑定的公钥集合，这个公钥集合是公开的，并且可由公钥集合的拥有者随时在里面添加公钥，当可追责第三方进行仲裁时需要一方的参与，比如可追责第三方与用户达成了一致意见，那么可追责第三方就会对商家的公钥进行流氓密钥攻击，这样会为用户产生一个假公钥，将假的公钥添加到用户的公钥集合中，这样便完成了流氓密钥攻击.

(2) 多可追责第三方，签名人数 n 较大时，可以让 m 个签名者在计算签名时额外的计算一个哈希值，这里需要至少一个加指数的签名者对于其余的签名者进行判别验证，进而实现了 m 个可追责下的 n 人多重签名，和 MSP 方案相比此方案具有了更高的签名效率与验证效率.

具体方案如下：

参数生成. $\mathrm{Pg}(\kappa)$，建立双线性群 $(q,\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_t,e,g_1,g_2)$，设哈希函数 $H_0:\{0,1\}^*\to\mathbb{G}_1$ 和 $H_1:\{0,1\}^*\to\mathbb{Z}_q$.

密钥生成. $\mathrm{Kg}(par)$ 选择一个随机数 $sk\xleftarrow{\$}\mathbb{Z}_q$，令 $pk\leftarrow g_2^{sk}\in\mathbb{G}_2$，输出 (pk,sk).

密钥聚合. $\mathrm{KAg}(PK=\{pk_{11},pk_{12},\cdots,pk_{m1},pk_{m2},pk_{m+1},\cdots,pk_n\})$ 输出

$$apk\leftarrow\prod_{i=1}^{m}pk_{i1}^{a_{i1}}\,pk_{i2}^{a_{i2}}\prod_{i=m+1}^{n}pk_i,$$

这里 $a_{ij}\leftarrow H_1(pk_{ij},PK)$.

签名. 签名是一个单轮协议，从第 $m+1$ 到第 n 个签名者生成 BLS 签名 $\sigma_i=H_0(m)^{sk_i}$，并发送给前 m 个签名者，前 m 个签名者中至少有一个签名者 $\mathrm{Sign}(par,\{\sigma_{m+1},\cdots,\sigma_n\},PK,sk_{i1},sk_{i2},m)$ 计算以下判别式，若

$$e\Big(\prod_{i=m+1}^{n}\sigma_i^{a_i},g_2^{-1}\Big)\cdot e\Big(H_0(m),\prod_{i=m+1}^{n}pk_i^{a_i}\Big)=1_{\mathbb{G}_t},$$

这里 $a_i\leftarrow H_1(pk_i,\{pk_{m+1},\cdots,pk_n\})$. 则输出

$$\sigma_i\leftarrow H_0(m)^{a_{i1}\cdot sk_{i1}}H_0(m)^{a_{i2}\cdot sk_{i2}},$$

多重签名 $\sigma\leftarrow\prod_{j=1}^{n}\sigma_j$ 可以由任何人生成，否则终止协议.

多重签名验证. $\mathrm{Vf}(par,apk,m,\sigma)$ 计算 $apk\leftarrow\mathrm{KAg}(PK)$，输出 1 当且仅当

$$e(\sigma,g_2^{-1})\cdot e(H_0(m),apk)=1_{\mathbb{G}_t}.$$

方案中前 m 个签名者计算判别式的方式可以分为以下三类：

(1) 前 m 个签名者中至少有一个签名者计算过判别式，即要求至少有一个签名者给出了第 $m+1$ 到第 n 个签名者的密钥拥有证明.

(2) 前 m 个签名者分工进行判别，由前 m 个签名者各自给出第 $m+1$ 到第 n 个签名者中的部分签名者的密钥拥有证明.

(3) 通过增加第二个构造中前 m 个签名者提供的密钥拥有证明的数量，使得第 $m+1$ 到第 n 个签名者的密钥拥有证明被多次验证，起到相互监督的功能，从而构造适合实际应用的方案.

6.2　对于 ASM 方案的改进

DEASM 方案(Double Exponential Accountable-Subgroup Multi-signature scheme)

ASM 方案是指 n 个签名者的任意子集 S 对消息 m 进行联合签名的方案，该签名可被验证是由子集 S 生成，Boneh 等人提出了可以支持公钥聚合与交互式群设置的 ASM 方案[8]. DEASM 方案提升了 ASM 方案的效率，在群设置方面，签名者计算 $\mu_{i,j}$ 无需计算额外的哈希值，计算 apk 时无需计算额外的哈希值与指数，减小了签名者与验证者的计算负担. 参数生成、密钥生成阶段与 DEMSP 方案相同. 设哈希函数 $H_0:\{0,1\}^*\to\mathbb{G}_1$，$H_1:\{0,1\}^*\to\mathbb{Z}_q$ 和 $H_2:\{0,1\}^*\to\mathbb{G}_2$.

群设置. 群设置阶段 $(sk_i,\mathcal{PK}=\{pk_{11},pk_{12},\cdots,pk_n\})$ 检查 $pk_i\in\mathcal{PK}$，i 是否是 pk_i 在 $\mathcal{PK}$ 中的指数，每个签名者计算聚合公钥 $apk\leftarrow\mathrm{KAG}(\mathcal{PK})$

$$apk=pk_{11}^{a_{11}}\,pk_{12}^{a_{12}}\prod_{i=2}^{n}pk_i,$$

这里 $a_{1i}=H_1(pk_{1i},PK)$，第 j 个签名者 $(j\neq1)$ 将 $\mu_{i,j}=H_2(apk,i)^{sk_j}$ 发送给第 i 个签名者. 当第一个

签名者收到其余签名者 j 的 $\mu_{1,j}$ 时，先判断

$$e\left(\prod_{i=2}^{n}\mu_{1,i}^{a_i},g_2^{-1}\right)\cdot e\left(H_2(apk,1),\prod_{i=2}^{n}pk_i^{a_i}\right)=1_{\mathbb{G}_t},$$

这里 $a_i=H_1(pk_i,\{pk_2,\cdots,pk_n\})$，若成立，则计算 $\mu_{j,1}=H_2(apk,j)^{a_{11}\cdot sk_{11}}H_2(apk,j)^{a_{12}\cdot sk_{12}}$，将它发送给第 j 个签名者. 然后计算 $\mu_{1,1}=H_2(apk,1)^{a_{11}\cdot sk_{11}}H_2(apk,1)^{a_{12}\cdot sk_{12}}$，最后每个签名者生成成员密钥 $mk_i\leftarrow\prod_{j=1}^{n}\mu_{i,j}$. 我们有

$$e(g_1,mk_i)=e(apk,H_2(apk,i)).$$

签名. 签名是一个单轮协议. Sign(par,PK,S,sk_i,mk_i,m)计算 $apk\leftarrow\mathrm{KAg}(\mathcal{PK})$，

$$s_i\leftarrow H_0(apk,m)^{sk_i}\cdot mk_i,$$

将(pk_i,s_i)发送给指定的合成者，这个指定的合成者可以是签名者之一，也可以是外部方，合成者计算

$$pk\leftarrow\prod_{j\in S}pk_j,\ s\leftarrow\prod_{j\in S}s_j,$$

输出多重签名 $\sigma:=(pk,s)$.

多重签名验证. Vf(par,apk,S,m,σ)输出 1，当且仅当

$$e(H_0(apk,m),pk)\cdot e\left(\prod_{j\in S}H_2(apk,j),apk\right)=e(s,g_2).$$

6.3 对于 MSDL 方案的改进

DEMSDL 方案(Double Exponential Discrete-logarithm based Multi-Signature Scheme)

Maxwell 等人[7]提出了实现聚合密钥的基于 Schnorr签名的多重签名方案，并将Bellare 和 Neven[6]中的多重签名方案的轮数由 3 轮降低为 2 轮，但是安全证明存在缺陷[11]，Boneh 等人[8]通过加入承诺提出 MSDL 方案，避免了这个缺陷，下面，将利用 DEMSP 方案提升 MSDL 方案的签名效率与验证效率.

基于离散对数的双指数多重签名方案 DEMSDL 使用哈希函数 $H_0,H_1,H_2:\{0,1\}^*\rightarrow\mathbb{Z}_q$.

参数生成. Pg(κ)建立一个阶为素数 q 的群 $\mathbb{G}$，生成元为 g，q 是一个 κ 比特的素数，输出 $par\leftarrow(\mathbb{G},g,q)$.

密钥生成. Kg(par)选择一个随机数 $sk\xleftarrow{\$}\mathbb{Z}_q$，令 $pk\leftarrow g_2^{sk}\in\mathbb{G}_2$，输出($pk,sk$). 密钥生成算法 Kg($par$)选择 $sk\xleftarrow{\$}\mathbb{Z}_q$.

密钥聚合. KAg($PK=\{pk_{11},pk_{12},pk_2,\cdots,pk_n\}$)输出

$$apk\leftarrow pk_{11}^{a_{11}}pk_{12}^{a_{12}}\prod_{i=2}^{n}pk_i,$$

这里 $a_{1i}\leftarrow H_1(pk_{1i},PK)$.

签名. 签名是一个交互式的三轮协议，输入 Sign(par,PK,sk,m)，签名者 i 执行以下三轮协议(注意这里当 $i=1$ 时，令 $r_1=(r_{11},r_{12})$，类似有 R_1,t_1,sk_1,s_1的表示).

第一轮：选择 $r_i\xleftarrow{\$}\mathbb{Z}_q$ 计算 $R_i\leftarrow g^{r_i}$，令 $t_i\leftarrow H_2(R_i)$，将 t_i 发送给 $pk_{11},pk_{12},pk_2,\cdots,pk_n$ 所对应的签名者，等待接受其他签名者的 $t_j(j\neq i)$.

第二轮：将 R_i 发送给 $pk_{11},pk_{12},pk_2,\cdots,pk_n$ 所对应的签名者，等待接受其他签名者的 $R_j(j\neq i)$，检验 $t_j=H_2(R_i)$ 对于所有的 $j=1,\cdots,n$.

第三轮：计算 $apk\leftarrow\mathrm{KAg}(PK)$，除第一个签名者以外的签名者计算 $R\leftarrow R_{11}R_{12}\prod_{j=2}^{n}R_j$，$c\leftarrow H_0(R,apk,m)$，签名者 i 计算 $s_i\leftarrow r_i+c\cdot sk_i \bmod q$，将 s_i 发送给第一个签名者.

第一个签名者判断

$$\prod_{i=2}^{n}g^{s_ia_i}=\prod_{i=2}^{n}R_i^{a_i}\prod_{i=2}^{n}pk_i^{a_ic},$$

其中 $a_i=H_1(pk_i,\{pk_2,\cdots,pk_n\})$，若成立，则计算 $R\leftarrow R_{11}R_{12}\prod_{j=2}^{n}R_j$，$c\leftarrow H_0(R,apk,m)$，并计算

$$s_{1i}\leftarrow r_{1i}+c\cdot sk_{1i}\cdot a_{1i} \bmod q,$$

其中 $a_{1i}=H_1(pk_{1i},PK)$，

$$s\leftarrow s_{11}+s_{12}+\sum_{j=2}^{n}s_j,$$

输出 $\sigma\leftarrow(R,s)$ 作为多重签名.

验证. Vf(par,apk,m,σ)将 σ 划分为 $(R,s)\in\mathbb{G}\times\mathbb{Z}_q$，计算 $c\leftarrow H_0(R,apk,m)$，输出 1 当且仅当

$$g^s\cdot apk^{-c}=R.$$

正确性分析.

$$\begin{aligned}g^s\cdot apk^{-c}&=g^{s_{11}+s_{12}+\sum_{i=2}^{n}s_i}\left(pk_{11}^{a_{11}}pk_{12}^{a_{12}}\prod_{i=2}^{n}pk_i\right)^{-c}\\&=g^{s_{11}+s_{12}+\sum_{i=2}^{n}s_i}\left(g^{sk_{11}a_{11}}g^{sk_{12}a_{12}}\prod_{i=2}^{n}g^{sk_i}\right)^{-c}\\&=g^{r_{11}}g^{r_{12}}g^{\sum_{i=2}^{n}r_i}\\&=R_{11}R_{12}\prod_{j=2}^{n}R_j\\&=R.\end{aligned}$$

7 方案对比

表 1 从方案是否与 Schnorr 签名或者 BLS 签名兼容(S/B)、签名时间、验证时间、有无可信第三方和安全性对 DEMSP 方案与其他多重签名方案进行对比，多重签名与 Schnorr 签名或者 BLS 签名兼容有助于支持在加密货币中用多重签名替代普通签名.

表 1　方案对比

方案	S/B	签名时间	验证时间	第三方	安全性
BN 方案	否	$\tau_1+\tau_{H_0}+\tau_R$	*	无	DL
MuSig 方案	是	$2\tau_1+\tau_{H_0}+\tau_{H_1}+\tau_R$	τ_2^n	无	DL
KR 模型	是	$\tau_1+\tau_{H_0}$	τ_2^0	有	CDH
KV 模型	是	$\tau_1+\tau_{H_0}$	τ_2^0	无	CDH
MSP 方案	是	$2\tau_1+\tau_{H_0}+\tau_{H_1}$	τ_2^n	无	co-CDH
DEMSP 方案	是	$\tau_1+\tau_{H_0}$	τ_2^1	有	co-CDH

注：τ_1，τ_2分别表示在$\mathbb{G}_1$，$\mathbb{G}_2$下计算一个指数所需要的时间，τ_1^i，τ_2^i分别表示在$\mathbb{G}_1$，$\mathbb{G}_2$下计算 i 个指数加上 n 个数相乘所需要的时间，τ_{H_0}，τ_{H_1} 分别代表访问 H_0，H_1需要的时间，τ_R指计算 $R=g^{r_i}$（r_i是每个签名者选择的随机数）所用的时间.

考虑多重签名方案的效率时，主要关注签名大小、签名时间与验证时间，上述几个多重签名方案都实现了签名聚合，使得签名大小与单个签名者的签名大小相同，考虑签名时间，DEMSP 方案、BN 方案、KR 模型与 KV 模型的签名时间与单人签名方案是一样的，而 MuSig 方案，MSP 方案需要进行额外的哈希运算，表 1 中的验证时间是指计算 apk 所需要的时间，BN 方案不能实现密钥聚合，因此这里并不考虑它的验证时间. DEMSP 方案计算聚合密钥的效率几乎和 KR 模型与 KV 模型一样，比 MSP 方案与 MuSig 方案快数倍，这里的倍数与签名者人数呈正相关.

考虑多重签名方案的安全性时，主要关注流氓密钥攻击，为抵抗流氓密钥攻击，DEMSP 方案以及 MSP 方案采用了“添加哈希值与指数来提供密钥拥有证明”的聚合技术，并且在这两个多重签名方案在验证时仅需要计算一个双线性对，DEMSP 方案可以看作在 MSP 方案中加入了可追责第三方，然而密钥可验证模型需要计算 n 个双线性对验证公钥的合法性来抵抗流氓密钥攻击，KR 模型可以看作在 KV 模型中加入了可信第三方.

多重签名方案中提供公钥所对应的私钥的知识证明是实现安全性的关键，根据这种知识证明多重签名方案分为两种模型，密钥拥有证明模型与普通公钥模型，KR 模型与 KV 模型属于密钥拥有证明模型，密钥拥有证明模型中需要签名者提供公钥所对应的私钥的知识证明，而提供知识证明的代价是昂贵的，密钥拥有模型适合于公钥小，公钥使用多次的场景，BN 方案、MuSig 方案和 MSP 方案属于普通公钥模型，适合于公钥大，公钥使用次数少的场景，普通公钥模型中公钥在需要用时随时生成，适合于更多的实际场景，DEMSP 方案也属于普通公钥模型，有效提升了 MSP 方案的签名效率与验证效率.

除 KR 模型之外，所有的方案均未引入可信第三方，引入可信第三方使得 KR 模型的效率相比其他方案都具有优势，然而引入可信第三方的代价是昂贵的，应尽量避免可信第三方的出现，DEMSP 方案通过引入可追责第三方实现了高签名效率与高验证效率，由于可追责第三方通过计算一个判别式就可以保证其他签名者的公钥合法性，并且任何人可以有效地对可追责第三方实行监督，因此 DEMSP 方案中引入可追责第三方产生的代价是低廉的.

8　数据分析

实验采用 Java 编程，图 3 是MSP方案与DEMSP方案的签名时间的对比，这里的签名时间指多重签名的签名时间，而不是每个签名者的签名时间，从图 3 中可以看出无论签名人数怎么变化，MSP 方案与 DEMSP 方案签名所用的时间基本是相同的，然而对于 MSP 方案中的签名者，在计算签名时需要其他签名者的公钥，这增加了签名者的通讯开销，该实验结果也表明 MSP 方案在实现签名聚合的同时，并没有对签名时间产生影响.

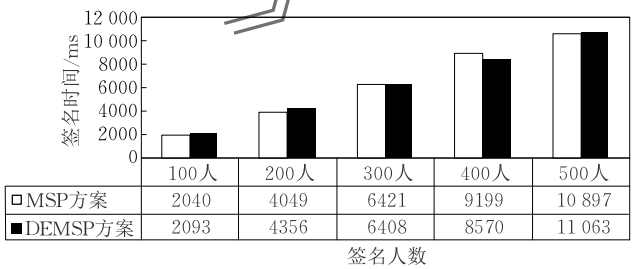

图 3　签名时间

图 4 是 MSP 方案与 DEMSP 方案的聚合密钥生成时间在签名人数较少时的对比，从图 4 中可以看出 DEMSP 方案的聚合密钥的生成时间基本不变，这里的时间主要是计算两个指数运算的时间，而 MSP 方案的聚合密钥的生成时间随签名者人数的增长线性增长，进一步可得在计算聚合公钥的效率上，DEMSP 方案比 MSP 方案高效数倍，倍数与签名者人数呈线性关系.

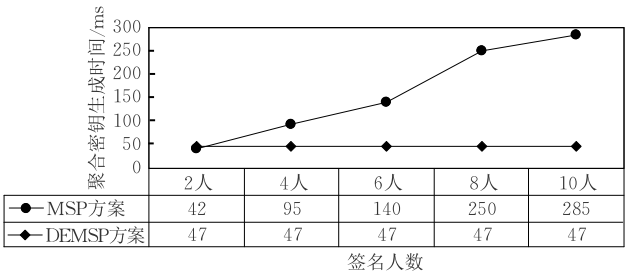

图 4 聚合密钥生成时间在签名人数较少时的对比

图 5 是 MSP 方案与 DEMSP 方案的聚合密钥生成时间在签名人数较多时的对比,图中左面的坐标代表 MSP 方案的聚合密钥的生成时间,右面的坐标代表 DEMSP 方案的聚合密钥的生成时间,可以看出图 4 中分析的"倍数与签名者人数呈线性关系"只适用于人数较少的情况,倍数的增长速度会随着人数的增加逐渐变慢直到 100 倍.

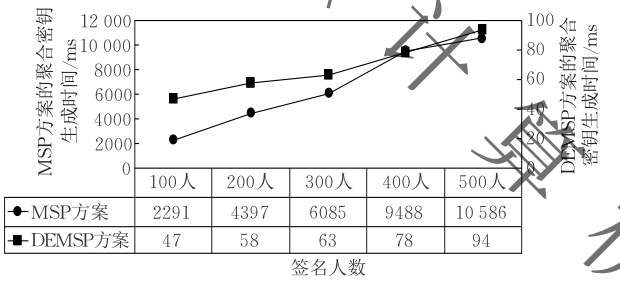

图 5 聚合密钥生成时间在签名人数较多时的对比

9 总 结

本文提出了一个高效率的双指数多重签名DEMSP方案,具有以下特点:

(1)实现了签名聚合与密钥聚合.

(2)实现了高效率的签名,每个签名者在DEMSP方案的签名阶段只需执行 BLS 签名,不需要获得其他签名者的公钥,与 MSP 方案相比减小了通信负担.

(3)实现了高效的验证,验证阶段计算聚合公钥的效率比 MSP 方案成倍提升,倍数与签名者人数呈正相关,签名者数量少时,倍数与签名者人数呈线性关系.随着签名者人数增多,倍数增长速度逐渐变慢直到 100 倍.

(4)引入可以被有效监督的可追责第三方,使得 DEMSP 方案高效且安全.

此外,本文给出了 DEMSP 方案的应用场景:在线交易与在线钱包,并将 DEMSP 方案拓展至多权威机构多重签名方案.DEMSP 方案推广到 ASM 方案和 MSDL 方案,使得计算聚合公钥的效率比原方案提升数倍.

参 考 文 献

[1] Itakura K, Nakamura K. A public key cryptosystem suitable for digital multi-signatures. NEC Research and Development, 1983, 71: 1-8

[2] Li C, Hwang T, Lee N. Threshold-multisignature schemes where suspected forgery implies traceability of adversarial shareholders//Santis D, eds. Advances in Cryptology-EUROCRYPT. Berlin, Germany, 1994, 950: 194-204

[3] Micali S, Ohta K, Reyzin L. Accountable-subgroup multisignatures//Proceedings of the 8th ACM Conference on Computer and Communications Security (CCS'01). New York, USA: ACM Press, 2001: 245-254

[4] Ristenpart T, Yilek S. The power of proofs-of-possession: Securing multiparty signatures against rogue-key attacks//Naor M, eds. Advances in Cryptology-EUROCRYPT 2007. Berlin, Germany, 2007, 4515: 228-245

[5] Bagherzandi A, Jarecki S. Multisignatures using proofs of secret key possession, as secure as the Diffie-Hellman problem //Proceedings of the 6th International Conference on Security and Cryptography for Networks (SCN'08). Berlin, Germany: Springer-Verlag, 2008: 218-235

[6] Bellare M, Neven G. Multi-signatures in the plain public-key model and a general forking lemma//Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS'06). New York, USA: ACM Press, 2006: 390-399

[7] Maxwell G, Poelstra A, Seurin Y, et al. Simple schnorr multi-signatures with applications to Bitcoin. Designs, Codes and Cryptography, 2019, 87: 2139-2164

[8] Boneh D, Drijvers M, Neven G. Compact multi-signatures for smaller blockchains//Peyrin T, Galbraith S, eds. Advances in Cryptology-ASIACRYPT 2018. Lecture Notes in Computer Science. Cham: Springer, 2018: 435-464

[9] Boneh D, Lynn B, Shacham H. Short signatures from the Weil pairing//Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT'01). London, UK: Springer-Verlag, 2001: 514-532

[10] Freeman D, Scott M, Teske E. A taxonomy of pairing-friendly elliptic curves. Journal of Cryptology, 2010, 23(2): 224-280

[11] Drijvers M, Edalatnejad K, Ford B, et al. On the security of two-round multi-signatures//Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP). San Francisco, USA, 2019: 1084-1101

[12] Nick J, Ruffing T, Seurin Y, Wuille P. MuSig-DN: Schnorr multi-signatures with verifiably deterministic nonces//Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. New York, USA, 2020: 1717-1731

[13] Alper H K, Burdges J. Two-round trip schnorr multi-signatures via delinearized witnesses//Malkin T, Peikert C, eds. Advances in Cryptology-CRYPTO 2021. Lecture Notes in Computer Science. Cham: Springer, 2021: 157-188

[14] Nick J, Ruffing T, Seurin Y. MuSig2: Simple two-round schnorr multi-signatures//Malkin T, Peikert C, eds, Advances in Cryptology-CRYPTO 2021. Lecture Notes in Computer Science. Cham: Springer, 2021: 189-221

[15] Le D, Yang G, Ghorbani A. A new multi-signature scheme with public key aggregation for blockchain//Proceedings of the 2019 17th International Conference on Privacy, Security and Trust (PST). Fredericton, Canada, 2019: 1-7

[16] Gorbunov S, Reyzin L, Wee H, Zhenfei Z. Pointproofs: Aggregating proofs for multiple vector commitments// Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. New York, USA, 2020: 2007-2023

[17] Hébant C, Phan D H, Pointcheval D. Linearly-homomorphic signatures and scalable mix-nets//Kiayias A, Kohlweiss M, Wallden P, Zikas V, eds. Public-Key Cryptography-PKC 2020. Lecture Notes in Computer Science. Cham: Springer, 2020: 597-627

[18] He S, Tang Q, Wu C Q, Shen X. Decentralizing IoT management systems using blockchain for censorship resistance. IEEE Transactions on Industrial Informatics, 2020, 16(1): 715-725

[19] Pointcheval D, Stern J. Security arguments for digital signatures and blind signatures. Journal of Cryptology, 2000, 13(3): 361-396

[20] Schnorr C P. Efficient signature generation by smart cards. Journal of Cryptology, 1991, 4(3): 161-174

[21] Bagherzandi A, Cheon J H, Jarecki S. Multi-signatures secure under the discrete logarithm assumption and a generalized forking lemma//Ning P, Syverson P F, Jha S, eds. Proceedings of the 15th Conference on Computer and Communications Security. Alexandria, Virginia, USA: ACM Press, 2008: 449-458

WANG Wen-Chao, Ph. D. candidate. His research interest is public key cryptography.

LIU Jin-Lu, Ph. D. candidate. Her research interests include cloud computing security and public key searchable encryption.

QIN Jing, Ph. D., professor, Ph. D. supervisor. Her research interests include cryptography and cloud computing security.

Background

Multi-signature schemes were proposed by Itakura and Nakamura in 1983, in which a group of signers sign the same message and make the verifier sure that each signer participates in the signature scheme. A trivial way to implement multiple signatures is to validate each signature one by one. This has two drawbacks:

(1) Multiple signatures and multiple public keys occupy a large amount of memory.

(2) n bilinear pairs need to be calculated to verify the signature of each message one by one, where n is the number of signers.

Therefore, The purpose of multiple signatures is to:

(1) Implement signature aggregation and key aggregation.

(2) Effectively resist rogue key attacks.

In order to ensure that the multi-signature scheme can not only realize signature aggregation and key aggregation, but also prevent rogue key attacks, a lot of work has been done.

(1) Key registration models and key verifiability models with a proof of possession of the secret key were presented. However, these schemes are all applied to zero-knowledge proof, which will incur expensive computational overhead.

(2) So far, the more practical model is the plain public key model, and representative schemes are the MSP scheme and the MuSig scheme.

The verification mode of the MSP scheme is:

$$e(\sigma, g_2^{-1}) \cdot e(H_0(m), apk) = 1_{\mathbb{G}_t},$$

$$apk \leftarrow \prod_{i=1}^{n} pk_i^{a_i},$$

where $a_i \leftarrow H_1(pk_i, \{pk_1, \cdots, pk_n\})$,

$$\sigma \leftarrow \prod_{j=1}^{n} s_j,$$

where $s_i \leftarrow H_0(m)^{a_i \cdot sk_i}$.

It can be seen from the above formula that the MSP scheme realizes signature aggregation and key aggregation on

the premise of resisting rogue key attacks. However, the MSP scheme has low signature efficiency and requires tedious calculation of aggregate public key.

The DEMSP scheme proposed in this paper has high signature efficiency and high verification efficiency, realizes signature aggregation and key aggregation, and can effectively resist rogue key attacks. The DEMSP scheme verification algorithm is:

$$e(\sigma, g_2^{-1}) \cdot e(H_0(m), apk) = 1_{\mathbb{G}_t},$$

$$apk \leftarrow pk_{11}^{a_{11}} pk_{12}^{a_{12}} \prod_{i=2}^{n} pk_i,$$

where $a_{1i} \leftarrow H_1(pk_{1i}, PK)$,

$$\sigma \leftarrow \prod_{j=1}^{n} \sigma_j,$$

where $\sigma_1 \leftarrow H_0(m)^{a_{11} \cdot sk_{11}} H_0(m)^{a_{12} \cdot sk_{12}}$, $\sigma_i = H_0(m)^{sk_i} (i \neq 1)$.

The DEMSP scheme has higher signature efficiency and verification efficiency than the MSP scheme. The verification efficiency refers to the time to calculate the aggregate public key. Specifically, the DEMSP scheme reduces the communication cost of signature and the computation cost of verification.

The DEMSP scheme introduces the accountable third party. On the one hand, the multi-signature requires the involvement of an accountable third party who is responsible for the multi-signature. On the other hand, the third party discriminates other signers to prevent other signers from carrying out rogue key attacks. Moreover this judgment can be effectively monitored by anyone, ensuring that the accountable third party will not cheat.