

基于同态代理重认证的可验证联邦聚合方法

杨帆¹, 孙奕¹, 陈性元^{1,2}, 高琦¹

(1.信息工程大学密码工程学院, 河南 郑州 450001; 2.密码科学技术国家重点实验室, 北京 100094)

摘要: 联邦学习可以通过共享梯度参数训练模型, 但在模型聚合过程中面临恶意服务器进行不诚实数据聚合的风险, 同时不受信任的用户参与联邦学习还可能会毒害全局模型, 模型训练过程中的可靠性和安全性受到威胁。针对联邦学习中用户不可信、聚合结果不可信等问题, 首次引入同态代理重认证, 提出了适用于多方聚合计算的双向认证方法, 并结合双掩码技术构造了具有隐私保护、高效可信联邦学习聚合方法, 不仅可以实现用户对全局模型聚合结果的正确性验证, 还能够使聚合服务器完成对用户上传模型来源的可信性验证和模型完整性验证, 防止攻击者恶意操控用户破坏安全聚合, 同时在验证过程中不会泄露用户隐私数据。通过形式化的安全性分析证明了可验证联邦聚合方法的安全性, 有效抵抗了伪造攻击和Sybil攻击, 且具有良好的鲁棒性。通过仿真实验进一步表明了所提方法能够在不影响联邦训练的情况下实现聚合结果的可信性验证, 且验证不会受到用户中途退出的影响。

关键词: 联邦学习; 可验证计算; 同态代理重认证; 完整性验证

中图分类号: TP309

文献标志码: A

DOI: 10.11959/j.issn.2096-109x.2024042

Verifiable federated aggregation method based on homomorphic proxy re-authentication

YANG Fan¹, SUN Yi¹, CHEN Xingyuan^{1,2}, GAO Qi¹

1.School of Cryptographic Engineering, Information Engineering University, Zhengzhou 450001, China

2.State Key Laboratory of Cryptology, Beijing 100094, China

Abstract: Federated learning, capable of training models by sharing gradient parameters, faces the risk of dishonest data aggregation by malicious servers during the model aggregation process. Untrusted users participating in federated learning may also pose a threat by poisoning the global model, thereby compromising the reliability and security of the model training process. To address these issues, homomorphic proxy re-authentication was introduced for the first time to propose a bi-directional authentication method suitable for multi-party aggregation

收稿日期: 2023-06-21; 修回日期: 2024-03-04

通信作者: 孙奕, 11112072@bjtu.edu.cn

基金项目: 河南省自然科学基金项目 (242300420297)

Foundation Item: The Natural Science Foundation of Henan Province (242300420297)

引用格式: 杨帆, 孙奕, 陈性元, 等. 基于同态代理重认证的可验证联邦聚合方法[J]. 网络与信息安全学报, 2024, 10(3): 91-106.

Citation Format: YANG F, SUN Y, CHEN X Y, et al. Verifiable federated aggregation method based on homomorphic proxy re-authentication[J]. Chinese Journal of Network and Information Security, 2024, 10(3): 91-106.

computation. Additionally, a privacy-preserving, efficient, and trustworthy federated learning aggregation method was constructed, combining the double mask technique. This method not only enables users to verify the correctness of global model aggregation results but also allows the aggregation server to assess the trustworthiness and model integrity of the model sources uploaded by the users. It prevents attackers from maliciously manipulating users to disrupt secure aggregation, without leaking users' private data during the verification process. The security of the verifiable federated aggregation method was demonstrated through formal security analysis, which effectively resists forgery attacks and Sybil attacks, exhibiting good robustness. Simulation experiments further illustrated that the proposed method can achieve credible verification of the aggregation results without impacting federated training. Moreover, the verification process remains unaffected even if users quit in the middle.

Keywords: federated learning, verifiable calculation, homomorphic proxy re-authentication, integrity verification

0 引言

随着万物互联的大数据时代的到来, 如何高效安全地处理海量数据成为一个亟待解决的问题。由谷歌提出的分布式机器学习框架——联邦学习 (federated learning, FL)^[1], 因其无须收集用户原始数据的训练网络能力逐渐受到学术界和业界的关注, 被广泛应用于疾病暴发预测、物联网入侵检测和自动驾驶等领域^[2-4]。FL 能够使参与实体独立并行地对其本地数据进行模型训练, 且只需共享模型梯度就能在聚合服务器的协调下协作训练全局模型, 大大降低了数据隐私泄露风险, 提高了训练效率和可扩展性。

在联邦学习的训练过程中服务器还可以通过分析用户模型的梯度向量来了解有关训练模型的信息。为了实现安全的联邦学习, Bonawitz 等^[5]利用秘密共享协议提出了在有用户中途退出时仍能实现隐私保护的联邦学习架构。但在联邦学习中仍然存在安全风险^[6], 有些“偷懒的”聚合服务器可能会为了节省计算成本不按要求完成所有聚合操作, 或是恶意向用户提供伪造的聚合结果来随意操作全局模型导致训练失败。因此, 为了防止错误的聚合结果影响模型训练, 还需要验证全局模型的正确性。例如, VerifyNet^[7]和 VeriFL^[8]等方法利用同态的思想, 实现了全局模型聚合结果的正确性验证。

但是, 仅能对聚合结果验证的联邦学习方法依旧是不完善的。最近的研究表明, 在分布式架构中, 联邦学习还容易受到 Sybil 攻击^[9]。中心聚合服务器只能要求用户上传本地模型, 不能访问用户的私有数据。相似度过高的用户模型很可能

受到 Sybil 攻击, 没有检测能力的服务器很容易被攻击者控制, 攻击者可以通过恶意操控用户来任意修改局部模型, 从而使聚合服务器用“有毒的”数据训练全局模型, 最终导致整个联邦学习过程失败。

综上所述, 现有的可验证联邦学习方法还不能够在实现对全局模型聚合结果正确性验证的同时, 实现对用户上传模型来源的可信性验证和模型的完整性验证。针对上述问题, 本文提出了基于同态代理重认证的具有隐私保护、高效可信的联邦学习方法, 主要的研究工作如下。

1) 本文首次引入同态代理重认证技术, 设计了适用于多方聚合计算的双向认证方法, 聚合者可以将不同用户私钥下单个消息向量的签名转换为接收方验证密钥下聚合计算结果的消息验证码 (message authentication code, MAC), 能在多用户数据聚合场景下, 提供可验证性的安全保证。遭到恶意攻击的用户不会影响系统中其他诚实参与聚合计算的用户得到真实聚合结果。当有用户遭到攻击时, 可为其单独重新分配新的密钥, 不会影响整个系统的安全。

2) 将本文提出的双向认证方法与双掩码加密技术相结合, 设计了具有隐私保护、高效可信的联邦学习方法, 实现了全局模型聚合结果的正确性验证, 同时还能实现对用户上传模型来源的可信性验证和模型的完整性验证, 防止攻击者恶意操控用户模型破坏安全聚合, 且在验证过程中不会泄露用户隐私数据。

3) 通过安全性分析证明本验证方法具有正确性和不可伪造性, 可以提供全局模型聚合结果的正确性验证以及用户模型来源可信性和完整性

验证。通过仿真实验对方法的计算开销和通信开销进行对比分析,结果表明本文方法能在不影响联邦学习聚合训练的情况下,实现高效可信的验证,且模型梯度的维度变化和用户退出不会影响本验证方法的性能。

1 相关工作

最近,有许多研究^[7-8, 10-13]基于可验证外包计算^[14]的思想实现了联邦学习全局模型聚合结果的正确性验证。其中,同态认证(homomorphic authenticators, HAs)^[15]是外包数据委托计算问题的最常用的解决方法。HAs能够用于验证大量数据的计算,不受信任的服务器可以根据用户提供的附加信息生成简短的“证明”,以保证通过外包数据计算的函数输出的正确性。在联邦学习训练过程中,每个用户在上传本地梯度时附加一些额外信息,聚合服务器在正确计算全局模型梯度的同时,根据这些附加信息生成聚合结果的正确性“证明”,并向每个参与训练的用户返回聚合梯度和“证明”,用户可根据“证明”检查聚合模型梯度是否正确。Xu等^[7]结合同态哈希和伪随机函数提出了第1个具有隐私保护的可验证联邦学习框架VerifyNet,但为了验证聚合结果正确性,使用了大量复杂的同态哈希和双线性对计算,使验证开销过大。为了在提高验证效率的同时降低通信开销,Guo等^[8]提出了高效的可验证联邦聚合方法VeriFL,将线性同态哈希(linear homomorphic hash, LHH)与承诺方法相结合,采用批量验证的方法平摊昂贵的哈希计算开销,实现了联邦学习聚合结果的高效验证,但随着用户退出次数的增加,为了恢复用户的相关验证参数,通信开销会大大增加。Ren等^[10]也通过结合LHH和数字签名,实现了可追溯的可验证联邦学习方法,同时基于错误学习加密和双掩码技术支持用户模型和全局模型的隐私保护。

但在实际应用中,文献[7-8, 10]需要用户之间传递大量中间参数,这会带来很大的通信开销,同时参数泄露会带来严重的安全问题。而且,联邦学习用户的数量可能非常大,更加需要计算效率高、通信开销小的验证方法。文献[11-12]在上述方法的基础上进行改进,构造具有加法同态性

质的双重聚合验证,但其方法中的共享参数设定不合理。Fu等^[13]利用拉格朗日插值技术实现了可验证的高效联邦学习(verifiable federated learning, VFL)方法,还结合单掩码加密实现了用户模型和全局模型的隐私保护。但其在处理用户退出的问题时并不完善,而且由于使用了拉格朗日插值,通信和验证开销大幅增加。

此外,联邦学习中还存在一种常见的安全威胁——Sybil攻击^[9, 16-17]。攻击者通过控制分布式系统中的多个节点用户,将故意设计的“有毒”参数上传到中心聚合服务器。攻击者利用这些“有毒”参数参与全局模型的训练,达到控制训练结果的目的。因此,安全的可验证联邦学习方法还应该能够保证参与训练的模型梯度来自系统内合法的安全用户,且与原始数据保持一致,即中心聚合服务器能够验证用户模型梯度的来源可信性和完整性。但是,以上可验证联邦学习方法的本质都是单密钥的,仅允许对使用相同密钥验证的数据进行同态计算,并不适合用于由多个用户提供数据进行验证的现实场景。由于所有用户都使用相同的密钥,任何用户都可以模仿其他用户,没有技术或法律上的方法来区分用户。而且,共享相同的密钥会使整个系统暴露在更高的攻击风险中,如果单个用户受到威胁,整个系统也会受到威胁,所有共享参数都必须重新初始化,受到攻击的系统恢复变得更加困难。

2 基础知识

本节主要描述方法中用到的密码学相关知识,主要包括伪随机生成器(pseudorandom generator, PRG)、密钥协商、秘密分享、同态代理重认证、双线性映射和困难问题假设。

2.1 伪随机生成器

PRG是由Blum等^[18]提出的密码学工具。它可以定义为可以生成一系列不可预测的比特序列函数 $\text{PRG}(S)$,将随机数“种子” S 作为输入,输出一串伪随机比特序列 $\{0, 1\}^m$ 。

2.2 密钥协商

Diffie-Hellman密钥协商^[19-20]能够为任意两个用户创建共享密钥,其安全性建立在计算离散对数的困难问题上。具体来说,首先给定一个阶为

素数 q 的群 G , 为每个用户 u 创建公私密钥对 $\text{KA.gen}(G, g, q) \rightarrow (\text{SK}_u, g^{\text{SK}_u})$, 其中 g 是群 G 的生成元, SK_u 和 g^{SK_u} 分别是私钥和公钥; 然后, 对于给定的用户 v 公钥 g^{SK_v} , 用户 u 可以通过 $\text{KA.agree}(\text{SK}_u, g^{\text{SK}_v}) \rightarrow s_{u,v}$ 生成用户 u 和用户 v 之间的共享密钥 $s_{u,v}$ 。在实际应用中, 为了方便计算, 通常用 $H((g^{\text{SK}_u})^{\text{SK}_v})$ 来构造 $s_{u,v}$, 显然 $s_{u,v} = s_{v,u}$ 。

2.3 秘密分享

Shamir^[21]的 t -out-of- N 秘密分享协议将秘密 s 划分为 N 个单独的部分, 其中 N 表示模型中的用户数, t 是阈值, 任何分享子集阈值大于 t 都可以恢复出秘密 s 。具体来说, 实现秘密分享协议主要有以下步骤。

(1) $\text{SS.share}(s, t, U) \rightarrow \{(u, s_u)\}_{u \in U}$: 输入阈值 $t \leq |U|$ 和秘密 s , 输出每个用户 u 的共享 s_u , 其中 U 表示指定的用户集合, 且 $|U| = N$ 。

(2) $\text{SS.recon}(\{(u, s_u)\}_{u \in M}) \rightarrow s$: 输入分享的子集 M , 其中 $u \in M \subseteq U$ 且 $t \leq |M|$, 输出秘密 s 。

2.4 同态代理重认证

Derler 等^[22]结合线性同态消息验证码^[23]和代理重加密^[24]的思想提出了同态代理重认证 (homomorphic proxy re-authenticator, HPRA)。HPRA 允许不同的发送方使用自己的密钥对数据进行身份认证, 而持有从发送方到接收方的重加密密钥的聚合者可以在不知情的情况下将这些单个签名或 MAC 转换为接收方密钥下的 MAC。而且, 聚合者可以基于同态性质对输入进行相应的函数计算, 从而得到对应于计算结果的 MAC 值。最后, 接收方可以根据 MAC 密钥验证计算的正确性。

定义 1 (同态代理重认证): 一个关于函数族 $\{F_{pp}\}$ 的同态代理重新认证是一组概率多项式时间 (probabilistic polynomial time, PPT) 算法 HPRA (Gen、SGen、VGen、Sign、Verify、SRGen、VRGen、Agg、AVerify), 它们的定义如下。

1) $\text{Gen}(1^\kappa, l)$: 以安全参数 κ 和向量长度 l 为输入, 输出参数 pp , 包括消息空间 M^l 、函数 $f: (M^l)^n \rightarrow M^l$, 以及在 κ 内呈指数级增大的标签空间, 其中 $l, n \leq \text{poly}(\kappa)$ 。

2) $\text{SGen}(pp)$: 以参数 pp 为输入, 输出一个签名者密钥 $(\text{id}, \text{sk}, \text{pk})$ 。

3) $\text{VGen}(pp)$: 以参数 pp 为输入, 输出 MAC 密钥 mk 和辅助信息 aux 。

4) $\text{Sign}(\text{sk}, m, \tau)$: 以签名者密钥 sk 、消息向量 m 和标记 τ 作为输入, 输出签名 σ 。

5) $\text{Verify}(\text{pk}, m, \tau, \sigma)$: 以签名公钥 pk 、消息向量 m 、标签 τ 和签名 σ 作为输入, 输出验证结果。

6) $\text{SRGen}(\text{sk}_i, \text{aux})$: 以签名者密钥 sk_i 和一些辅助信息 aux 为输入, 输出重新加密密钥 rk_i 。

7) $\text{VRGen}(\text{pk}_i, \text{mk}, \text{rk}_i)$: 以签名者公钥 pk_i 、MAC 密钥 mk 和重新加密密钥 rk_i 作为输入, 输出聚合密钥 ak_i 。

8) $\text{Agg}((\text{ak}_i)_{i \in [n]}, (\sigma_i)_{i \in [n]}, \tau, f)$: 以 n 个聚合密钥 $(\text{ak}_i)_{i \in [n]}$ 、 n 个签名 $(\sigma_i)_{i \in [n]}$ 、标签 τ 和函数 $f \in F_{pp}$ 为输入, 输出一个聚合认证消息向量 A 。

9) $\text{AVerify}(\text{mk}, A, \text{ID}, f)$: 以 MAC 密钥 mk 、聚合认证消息向量 A 、 n 个标识符 $\text{ID} = (\text{id}_i)_{i \in [n]}$ 和函数 $f \in F_{pp}$ 为输入, 若输出消息向量和标签 (m, τ) 表示验证成功, 否则输出 $(\emptyset, \emptyset)$ 。

2.5 双线性映射和困难问题假设

定义 2 (双线性映射): 对于阶为素数 p 的循环群 G 和 G_T , g 为群 G 的生成元, 双线性映射 $e: G \times G \rightarrow G_T$ 具有如下性质。

1) 可计算性: 对于任意的元素 $u, v \in G$, $e(u, v)$ 是可计算的。

2) 非退化性: $e(g, g) \neq 1_{G_T}$ 。

3) 双线性: 对于任意 $u, v \in G$ 及 $a, b \in \mathbb{Z}_p$, 有等式 $e(u^a, v^b) = e(u, v)^{ab}$ 成立。

定义 3 (双线性判断 DH 假设) (bilinear decisional Diffie-Hellman, BDDH): 给定阶为素数 p 的循环群 G 和 G_T , g 为群 G 的生成元, a, b, c, z 为 $\mathbb{Z}_p^*$ 中随机选取元素, $e: G \times G \rightarrow G_T$ 表示双线性映射。对于任意概率多项式时间算法 A , 存在可忽略函数 negl 满足:

$$|\Pr[A(g, g^a, g^b, g^c, e(g, g)^{abc}) = 1] -$$

$$\Pr[A(g, g^a, g^b, g^c, e(g, g)^z) = 1]| \leq \text{negl}(\lambda) \quad (1)$$

定义 4 (扩展的双线性计算 DH 假设) (extended bilinear computational Diffie-Hellman,

eCDH): 给定阶为素数 p 的循环群 G 和 G_T , g 为群 G 的生成元, a, b, c 为 $\mathbf{Z}_p^*$ 中随机选取元素, $e: G \times G \rightarrow G_T$ 表示双线性映射。对于任意概率多项式时间算法 $\mathcal{A}$, 存在可忽略函数 negl 满足:

$$\Pr[\mathcal{A}(g, g^a, g^b, g^c) \rightarrow e(g, g)^{abc}] \leq \text{negl}(\lambda) \quad (2)$$

3 系统模型与威胁模型

3.1 系统模型

本文提出的基于同态代理重认证的可验证联邦学习聚合方法的系统模型如图1所示, 主要包含3个实体: ①可信授权中心主要负责初始化, 生成公共参数和密钥; ②用户在每次联邦学习迭代中训练本地模型, 并生成模型梯度的签名; ③聚合服务器是具有强大计算和存储能力的第三方不可信实体, 主要负责验证用户梯度签名, 并聚合全局模型, 同时生成结果的完整性证明。

各实体间主要的交互流程如下。首先, 由可信授权中心进行初始化, 生成公共参数和密钥后, 向全局用户分发。此外, 可信授权中心可以为受到攻击的用户重新分配密钥。然后, 用户生成与聚合验证有关的密钥参数, 本地保存签名私钥和验证密钥。在每轮训练过程中, 用户通过本地训练得到梯度向量后, 根据自己的私钥对本地梯度签名, 并将加密后的模型梯度与签名一起上

传给服务器。聚合服务器首先验证每个用户签名的正确性, 通过验证的用户模型梯度才能参与全局模型的训练, 同时根据参与训练的用户模型梯度签名生成全局模型聚合结果证明, 并将全局模型聚合梯度和证明一起返回给每个参与训练的用户。最后, 用户根据验证密钥验证全局模型计算结果的正确性。

3.2 威胁模型

本文方法的威胁模型是“半诚实的”, 即聚合服务器和参与用户是诚实但好奇的(假设可信授权中心是值得信赖的, 不会与任何实体串通, 只负责初始化整个系统, 分发完公共参数和密钥后下线)。本文主要考虑了以下两种类型的攻击。

1) 聚合服务器伪造聚合结果欺骗其他诚实用户, 致使联邦学习模型训练失败。聚合服务器按照约定协议执行计算并将结果返回给用户, 但会保存所有接收的数据和中间的计算结果, 试图从中推断出任意单个用户的隐私信息, 也可能通过恶意手段利用训练中获取的信息来推测用户的隐私数据, 从而修改聚合模型结果并伪造证明, 欺骗其他诚实用户。

2) 在用户向聚合服务器上传本地模型的过程中可能会受到敌手的恶意篡改或伪造等攻击。敌手在用户向聚合服务器上传本地模型的过程中

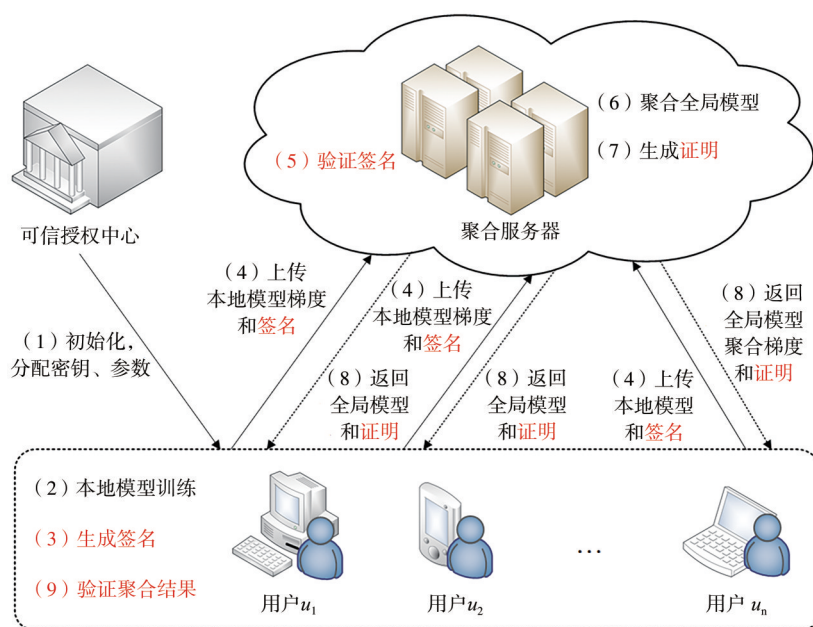


图1 系统模型
Figure 1 System model

监听并拦截传输数据, 恶意篡改数据或直接伪造出虚假的本地模型替换原本的数据, 并将其成功上传给聚合服务器, 最终导致聚合服务器对“有毒的”用户模型进行训练, 致使联邦学习失败。

4 方法设计

4.1 适用于多方聚合计算的双向认证方法

为了实现对聚合计算结果的正确性验证, 同时又能实现对参与聚合数据的来源可信性和完整性验证, 引入同态代理重认证^[22]的概念, 提出了能实现多方向量聚合计算的双向认证方法, 每个算法的具体定义如下。

1) **Setup**($1^\kappa, l$): 以安全参数 κ 和向量长度 l 为输入, 输出参数 pp 。其中, pp 包括双线性对映射 $e: G \times G \rightarrow G_T$, G 是阶为素数 q 、生成元为 g 的乘法循环群, 随机参数 $(g_i)_{i \in [l]} \xleftarrow{R} G^l$ 和哈希函数 $H: \mathbf{Z}_q \rightarrow G$ 。

2) **VGen**(pp): 以参数 pp 为输入, 随机选择参数 $\alpha \xleftarrow{R} \mathbf{Z}_q$, 输出验证密钥 $mk = \alpha$ 。

3) **SGen**(pp): 以参数 pp 为输入, 随机选择秘密参数 $\beta_u \xleftarrow{R} \mathbf{Z}_q$, 输出签名者 u 的密钥 (id_u, sk_u, pk_u) 。其中 $id_u = g^{1/\beta_u}$, $pk_u = g^{\beta_u}$, $sk_u = \beta_u$ 。

4) **Sign**($id_u, sk_u, M_u, (g_i)_{i \in [l]}$): 该算法由参与聚合计算的用户执行, 以签名者身份 id_u 、私钥 sk_u 、长度为 l 的消息向量 $M_u = (m_{u1}, m_{u2}, \dots, m_{ul})$ 和安全参数 $(g_i)_{i \in [l]}$ 作为输入, 输出消息向量 M_u 的签名 $\sigma_u = \left(H(id_u) \cdot \prod_{i=1}^l g_i^{m_{ui}} \right)^{sk_u}$ 。

5) **Verify**($id_u, pk_u, M_u, \sigma_u$): 该算法由聚合者执行, 以签名者身份 id_u 、公钥 pk_u 、消息向量 M_u 和签名 σ_u 作为输入, 判断等式 $e(\sigma_u, g) = e\left(H(id_u) \cdot \prod_{i=1}^l g_i^{m_{ui}}, pk_u\right)$ 是否成立, 输出验证结果 1 或 0, 其中 1 表示验证成功, 0 表示验证失败。

6) **RGen**(id_u, mk): 该算法由参与聚合计算的用户执行, 以签名者身份 id_u 和验证密钥 mk 为输入, 输出聚合密钥 ak_u , 其中 $ak_u = (id_u)^{mk}$ 。

7) **ASign**((ak_u) $_{u \in [n]}$, (σ_u) $_{u \in [n]}$): 该算法由聚合者执行, 以 n 个聚合密钥 (ak_u) $_{u \in [n]}$ 和 n 个签名

(σ_u) $_{u \in [n]}$ 为输入, 输出一个聚合结果的证明 $\sigma = \prod_{u \in [n]} e(\sigma_u, ak_u)$ 。

8) **AVerify**($M, \sigma, (id_u)_{u \in [n]}, (g_i)_{i \in [l]}, g, mk$): 该算法由参与聚合计算的用户执行, 以长度为 l 的聚合向量 $M = M_1 + M_2 + \dots + M_n = (m_1, m_2, \dots, m_l)$ 、聚合结果证明 σ 、 n 个签名者身份 (id_u) $_{u \in [n]}$ 、安全参数 $(g_i)_{i \in [l]}$ 、 g 和验证密钥 mk 为输入, 计算 $\sigma' = \left(\prod_{u \in [n]} e(g, H(id_u)) \cdot e\left(\prod_{i=1}^l g_i^{m_i}, g\right) \right)^{mk}$, 判断等式 $\sigma' = \sigma$ 是否成立。输出向量聚合计算的验证结果 1 或 0, 其中 1 表示验证成功, 0 表示验证失败。

定义 5 (正确性): 如果对于适用于向量长度为 l 且满足安全参数 κ 的聚合计算, 初始化生成安全参数 $pp \leftarrow \text{Setup}(1^\kappa, l)$, 对于所有适用于该聚合计算的用户个数 n , 生成所有用户的密钥 $((id_u, sk_u, pk_u) \leftarrow \text{SGen}(pp))_{u \in [n]}$ 和验证密钥 $mk \leftarrow \text{VGen}(pp)$, 对于所有的消息向量 $(M_u)_{u \in [n]}$, 生成签名 $\left(\sigma_u \leftarrow \text{Sign}(id_u, sk_u, M_u, (g_i)_{i \in [l]}) \right)_{u \in [n]}$ 和聚合密钥 $(ak_u \leftarrow \text{RGen}(id_u, mk))_{u \in [n]}$, 对于聚合计算 $M = M_1 + M_2 + \dots + M_n = (m_1, m_2, \dots, m_l)$, 生成聚合结果的证明 $\sigma \leftarrow \text{ASign}((ak_u)_{u \in [n]}, (\sigma_u)_{u \in [n]})$, 都有 $(\text{Verify}(id_u, pk_u, M_u, \sigma_u) = 1)_{u \in [n]}$ 和 $\text{AVerify}(M, \sigma, (id_u)_{u \in [n]}, (g_i)_{i \in [l]}, g, mk) = 1$ 成立, 那么本文提出的适用于多方聚合计算的双向认证方法是正确的。

定义 6 (不可伪造性): 本文提出的适用于多方聚合计算的双向认证方法的不可伪造性主要包含用户向量签名的不可伪造性和向量聚合结果证明的不可伪造性两个方面。为了形式化地描述不可伪造性, 定义多项式时间敌手 $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ 与挑战者 $\mathcal{C}$ 的安全性游戏如下。

1) 设置: 挑战者 $\mathcal{C}$ 首先根据 $\text{Setup}(1^\kappa, l)$ 生成公开参数 pp ; 然后, 根据 $\text{VGen}(pp)$ 生成验证密钥 mk , 根据 $\text{SGen}(pp)$ 为每个用户生成 $(id_u, sk_u, pk_u)_{u \in [n]}$; 最后, 将公开参数 pp 和 $(id_u, pk_u)_{u \in [n]}$ 发送给敌手 $\mathcal{A}$ 。

2) 签名问询：敌手 $\mathcal{A}_1$ 向挑战者 $\mathcal{C}$ 多次询问向量 $\mathbf{M}_u$ 的签名，挑战者 $\mathcal{C}$ 运行 Sign 算法生成签名 σ_u 并返回给敌手 $\mathcal{A}_1$ 。

3) 聚合密钥问询：敌手 $\mathcal{A}_2$ 向挑战者 $\mathcal{C}$ 多次询问身份为 id_u 的用户的聚合密钥，挑战者 $\mathcal{C}$ 运行 RGen 算法生成聚合密钥 ak_u 并返回给敌手 $\mathcal{A}_2$ 。

4) 聚合结果证明问询：敌手 $\mathcal{A}_2$ 向挑战者 $\mathcal{C}$ 多次询问不同组 $\mathbf{M}_1, \mathbf{M}_2, \dots, \mathbf{M}_n$ 的聚合结果证明，挑战者 $\mathcal{C}$ 依次运行 Sign 和 ASign 算法，将聚合结果证明 σ 返回给敌手 $\mathcal{A}_2$ 。

5) 伪造签名：敌手 $\mathcal{A}_1$ 输出一个伪造的向量 $\mathbf{M}_u^*$ 以及对应的签名 σ_u^* ，如果 $\text{Verify}(\text{id}_u, \text{pk}_u, \mathbf{M}_u^*, \sigma_u^*) = 1$ 成立，则称敌手 $\mathcal{A}_1$ 赢得了该游戏。

6) 伪造聚合结果证明：敌手 $\mathcal{A}_2$ 输出一个伪造的聚合结果 $\mathbf{M}^*$ 以及对应的聚合结果证明 σ^* ，如果 $\text{AVerify}(\mathbf{M}^*, \sigma^*, (\text{id}_u)_{u \in [n]}, (g_i)_{i \in [l]}, g, \text{mk}) = 1$ 成立，则称敌手 $\mathcal{A}_2$ 赢得了该游戏。

在定义 6 定义的安全性游戏中，如果对于任意多项式时间敌手 $\mathcal{A}$ 赢得游戏的概率是可忽略的，那么本文提出的适用于多方聚合计算的双向认证方法是不可伪造的。

4.2 可验证联邦聚合方法

本节具体介绍基于同态代理重认证的可验证联邦聚合方法，方法流程如图 2 所示，每个阶段的具体步骤如下。

4.2.1 初始化阶段

步骤 1 生成公共参数

可信授权中心进行初始化，以安全参数 κ 和 l 作为输入，根据初始化算法 $\text{Setup}(1^\kappa, l)$ 生成有关公共参数 pp ，包括双线性对映射 $e: G \times G \rightarrow G_T$ ， G 是阶为素数 q 、生成元为 g 的乘法循环群，安全的哈希函数 $H: \mathbf{Z}_q \rightarrow G$ ，用于验证的随机参数 $(g_i)_{i \in [l]}$ ，其中 l 为梯度向量的长度。每个用户 u 本地的梯度向量可以表示为 $\mathbf{X}_u = (x_{u1}, x_{u2}, \dots, x_{ul})$ 。

可信授权中心根据 VGen(pp) 算法生成验证密钥 mk ，随机选择秘密参数 $\beta_u \xleftarrow{\mathbf{R}} \mathbf{Z}_q$ ，生成签名密钥组 $(\text{id}_u, A_u^{\text{PK}}, A_u^{\text{SK}}) \leftarrow \text{SGen}(\text{pp})$ ，其中身份 $\text{id}_u = g^{1/\beta_u}$ ，公钥 $A_u^{\text{PK}} = g^{\beta_u}$ ，私钥 $A_u^{\text{SK}} = \beta_u$ 。当有用户遭到恶意攻击时，可信授权中心可为其重新生

成 $(\text{id}_u, A_u^{\text{PK}}, A_u^{\text{SK}})$ ，不影响其他安全用户的正常模型训练。

可信授权中心将公共参数和有关密钥打包成消息 $\{\text{pp}, (g_i)_{i \in [l]}, g, \text{mk}, \text{id}_u, A_u^{\text{PK}}, A_u^{\text{SK}}\}$ ，通过安全通道分别发送给对应的用户 u 。

步骤 2 密钥生成

用户 u 接收可信授权中心发送的消息 $\{\text{pp}, (g_i)_{i \in [l]}, g, \text{mk}, \text{id}_u, A_u^{\text{PK}}, A_u^{\text{SK}}\}$ ，然后根据双向认证算法生成重签名密钥 $\text{ak}_u \leftarrow \text{RGen}(\text{id}_u, \text{mk})$ ，其中 $\text{ak}_u = (\text{id}_u)^{\text{mk}} = (g^{1/\beta_u})^{\text{mk}}$ ，根据 Diffie-Hellman 密钥协商算法生成用于第 1 层掩码的密钥对 $(S_u^{\text{PK}}, S_u^{\text{SK}}) \leftarrow \text{KA.gen}(\text{pp})$ 。

用户 u 将 $(S_u^{\text{PK}}, A_u^{\text{PK}}, \text{ak}_u, \text{id}_u)$ 打包，并通过安全通道发送给服务器。

服务器至少需要收集 t 条来自用户的消息 (t 指的是 Shamir 秘密分享方法中的门限)，并用 U_1 表示这组用户。如果收到的消息数少于 t ，则中止，重新开始。最后，向 U_1 中的所有用户广播列表 $\{(v, S_v^{\text{PK}}, \text{程id}_v)\}_{v \in U_1}$ 。

4.2.2 密钥分享阶段

步骤 1 生成密钥分享

用户 u 接收到服务器广播的消息列表 $\{(v, S_v^{\text{PK}}, \text{id}_v)\}_{v \in U_1}$ ，检查 $|U_1| \geq t$ 以及公钥是否都不相同。用户 u 选择随机元素 b_u 作为生成第 2 层掩码的 PRG 算法的种子密钥，生成 b_u 的 $t - |U_1|$ 秘密分享 $\{(v, b_{u,v})\}_{v \in U_1} \leftarrow \text{SS.share}(b_u, t, U_1)$ ，其中 $b_{u,v}$ 表示用户 u 分享给用户 v 的秘密分享。生成 S_u^{SK} 的秘密分享 $\{(v, S_{u,v}^{\text{SK}})\}_{v \in U_1} \leftarrow \text{SS.share}(S_u^{\text{SK}}, t, U_1)$ ，其中 $S_{u,v}^{\text{SK}}$ 表示用户 u 分享给用户 v 的秘密分享。

步骤 2 加密封装

用户 u 为每个用户 $v \in U_1 \setminus \{u\}$ 计算加密封装的密钥 $A_{u,v} = (A_v^{\text{PK}})^{A_u^{\text{SK}}}$ ，并计算密文 $e_{u,v} \leftarrow \text{AE.enc}(A_{u,v}, u \| v \| S_{u,v}^{\text{SK}} \| b_{u,v})$ ，其中 $\text{AE.enc}()$ 是一个密钥为 $A_{u,v}$ 的对称加密函数，随后打包所有密文，并将列表 $\{e_{u,v}\}_{v \in U_1}$ 发送给服务器。

服务器需要至少收集 t 个用户发来的消息，并用 $U_2 \subseteq U_1$ 表示这组用户。如果收到的消息数少于 t ，则中止，重新开始。然后，服务器向每个 U_2 中的用户 v 广播密文列表 $\{e_{u,v}\}_{v \in U_2}$ 。

4.2.3 加密-签名阶段

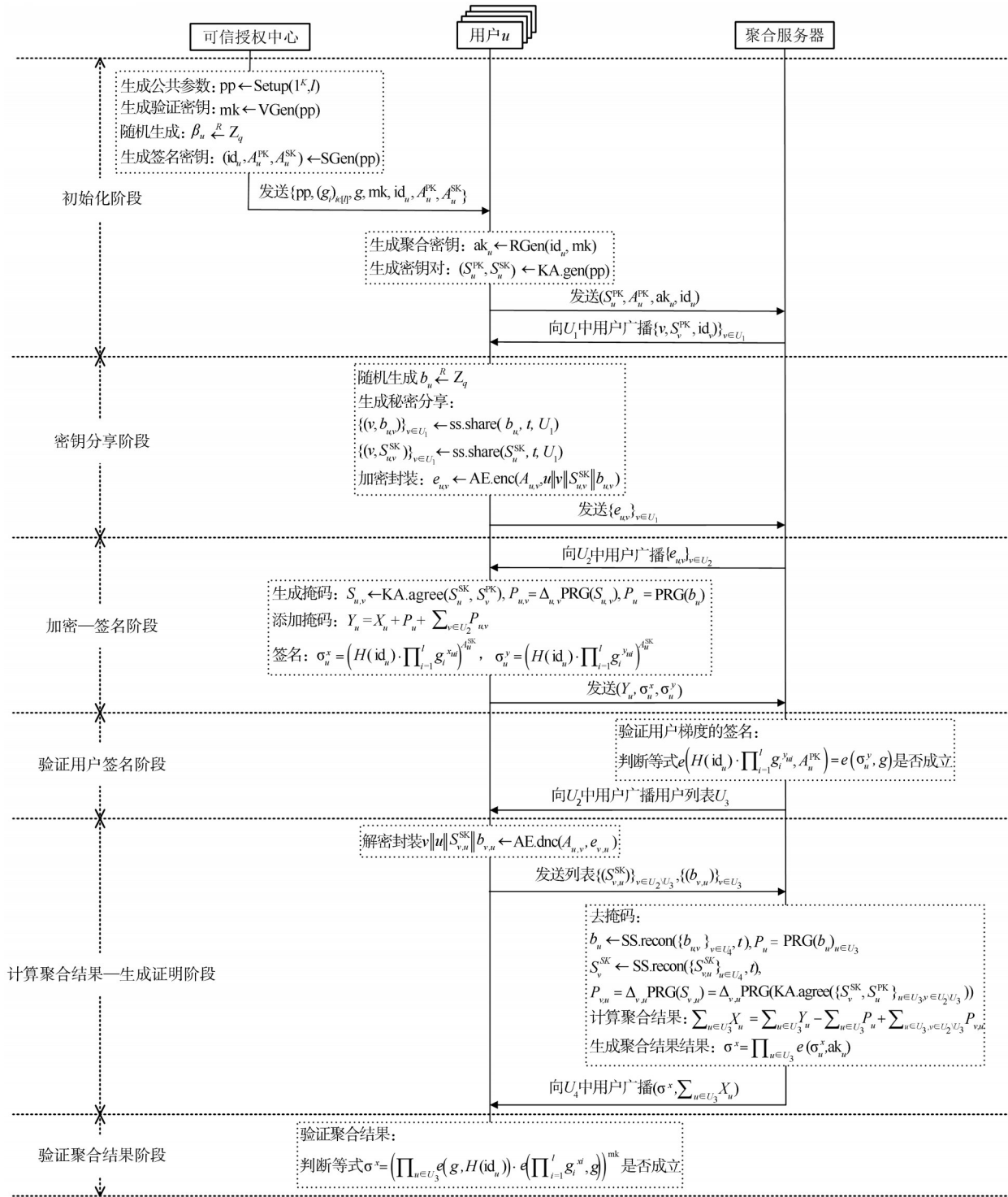


图2 方法流程
Figure 2 Scheme flow

步骤1 计算掩码

用户 u 从服务器接收并存储密文列表 $\{e_{u,v}\}_{v \in U_2}$, 检查是否满足 $U_2 \subseteq U_1$ 和 $|U_2| \geq t$ 。如果不满足, 则中止, 重新开始。然后, 用户 u 为用户 $v \in U_2 \setminus \{u\}$

计算 $S_{u,v} \leftarrow \text{KA.agree}(S_u^{\text{SK}}, S_v^{\text{PK}})$, 用伪随机生成算法将其扩展为随机向量 $P_{u,v} = \Delta_{u,v} \cdot \text{PRG}(S_{u,v})$, 并将其作为第1层掩码。当 $u < v$ 时, $\Delta_{u,v} = 1$, 当 $u > v$ 时, $\Delta_{u,v} = -1$, 由此可以得到 $\forall u \neq v, P_{u,v} +$

$P_{v,u} = 0$ 。同理，使用PRG算法将 b_u 扩展为随机向量 $P_u = \text{PRG}(b_u)$ ，并将其作为第2层掩码。

步骤2 生成加密后梯度向量

用户 u 计算加密后的本地梯度向量为 $Y_u = X_u + P_u + \sum_{v \in U_2} P_{u,v}$ 。

如果上述任何一个操作（包括密钥协商、PRG和对称加密）失败，则中止，重新开始。

步骤3 签名

用户 u 根据算法分别对本地模型梯度向量 X_u 和加密后的本地模型梯度向量 Y_u 进行签名，得到向量 X_u 的签名 $\sigma_u^x = \left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{x_u} \right)^{A_u^{\text{SK}}}$ 和向量 Y_u 的签名 $\sigma_u^y = \left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{y_u} \right)^{A_u^{\text{SK}}}$ 。随后，用户 u 将加密后的梯度向量和签名打包为 $(Y_u, \sigma_u^y, \sigma_u^x)$ 发送给服务器。

4.2.4 验证用户签名阶段

首先，服务器至少需要收集 t 个用户发来的消息，并用 $U_3 \subseteq U_2$ 表示这组用户。如果收到的消息少于 t 个，则中止，重新开始。

然后，服务器根据算法 $\text{Verify}(\text{id}_u, A_u^{\text{PK}}, Y_u, \sigma_u^y)$ 验证 U_3 中每个用户的签名 σ_u^y ，判断等式 $e(\sigma_u^y, g) = e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{y_u}, A_u^{\text{PK}}\right)$ 是否成立，如果验证不通过，则从用户列表 U_3 中去掉此用户。

最后，服务器给 U_2 中的每个用户广播用户列表 U_3 。

4.2.5 计算聚合结果 - 生成证明阶段

步骤1 解密封装

用户 u 检查是否满足 $U_3 \subseteq U_2$ 和 $|U_3| \geq t$ 。如果不满足，则中止，重新开始。随后，解密用户 $v \in U_2 \setminus \{u\}$ 加密封装的密文 $\{e_{v,u}\}_{v \in U_2 \setminus \{u\}}$ ，得到 $v||u||S_{v,u}^{\text{SK}}||b_{v,u} \leftarrow \text{AE.dnc}(A_{u,v}, e_{v,u})$ ，然后发送列表 $\{(S_{v,u}^{\text{SK}})\}_{v \in U_2 \setminus \{u\}}$ 和 $\{(b_{v,u})\}_{u \in U_3}$ 给服务器，其中 $U_2 \setminus U_3$ 指的是已经将加密封装的密钥消息发送给了服务器，但在发送加密梯度向量时退出的用户。

步骤2 去掩码

服务器需要收集至少 t 个用户消息，并用 $U_4 \subseteq U_3$ 表示这组用户。如果收到的消息少于 t 个，则中止，重新开始。然后，为用户 $u \in U_3$

重构 $b_u \leftarrow \text{SS.recon}(\{b_{u,v}\}_{v \in U_4}, t)$ 并计算 $P_u = \text{PRG}(b_u)$ ，为中途退出的用户 $v \in U_2 \setminus U_3$ 重构 $S_v^{\text{SK}} \leftarrow \text{SS.recon}(\{S_{v,u}^{\text{SK}}\}_{u \in U_4}, t)$ 并计算用户 $u \in U_3, v \in U_2 \setminus U_3$ 的 $P_{v,u} = \Delta_{v,u} \text{PRG}(S_{v,u}) = \Delta_{v,u} \text{PRG}(\text{KA.agree}(\{S_v^{\text{SK}}, S_u^{\text{PK}}\}_{u \in U_3, v \in U_2 \setminus U_3}))$ 。

步骤3 聚合用户梯度

服务器计算本轮训练的全局模型聚合结果为 $\sum_{u \in U_3} X_u = \sum_{u \in U_3} Y_u - \sum_{u \in U_3} P_u + \sum_{u \in U_3, v \in U_2 \setminus U_3} P_{v,u}$ 。

步骤4 生成证明

服务器根据算法 $\text{ASign}((\text{ak}_u)_{u \in [n]}, (\sigma_u)_{u \in [n]})$ 得到 $\sigma^x = \prod_{u \in U_3} e(\sigma_u^x, \text{ak}_u)$ ，并将其作为全局模型聚合结果的正确性证明。

服务器向 U_4 中所有用户广播聚合结果及证明 $(\sigma^x, \sum_{u \in U_3} X_u)$ 。

4.2.6 验证聚合结果阶段

用户 u 得到的全局模型的聚合梯度向量为 $\sum_{u \in U_3} X_u = (x_1, x_2, \dots, x_l)$ ，根据验证聚合结果算法 $\text{AVerify}(X_u, \sigma^x, (\text{id}_u)_{u \in [n]}, (g_i)_{i \in [l]}, g, \text{mk})$ ，判断等式 $\sigma^x = \left(\prod_{u \in U_3} e(g, H(\text{id}_u)) \cdot e\left(\prod_{i=1}^l g_i^{x_u}, g\right) \right)^{\text{mk}}$ 是否成立，如果验证通过，则证明聚合结果正确，用户 u 认可聚合结果。

5 方法分析

5.1 安全性分析

5.1.1 双向认证方法的安全性

本节从正确性和不可伪造性两个方面具体分析第4.1节提出的适用于多方聚合计算的双向认证方法的安全性，并给出安全定理和证明。

定理1（正确性）：适用于多方聚合计算的双向认证方法的正确性，包括消息向量签名验证的正确性和消息向量聚合结果验证的正确性两个部分。

证明 在验证用户签名阶段，聚合者根据用户 u 的身份 id_u 、公钥 pk_u 和消息向量 M_u ，通过判断等式 $e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{m_u}, \text{pk}_u\right) = e(\sigma_u, g)$ 是否成立来验证消息向量 M_u 的签名 σ_u 的正确性。

$$e(\sigma_u, g) = e\left(\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{m_u}\right)^{\text{sk}_u}, g\right) =$$

$$\begin{aligned}
& e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{m_i}, g\right) = \\
& e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{m_i}, g^{\beta_u}\right) = \\
& e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{m_i}, \text{pk}_u\right) \quad (3)
\end{aligned}$$

如果式 (3) 成立, 则表示签名 σ_u 验证通过, 聚合者能够相信消息向量 $\mathbf{M}_u$ 的来源可信性和完整性。

在本文方法中的用户验证聚合结果阶段, 用户根据聚合者返回的聚合结果 $\mathbf{M}$ 和证明 σ , 通过判断 $\sigma = \left(\prod_{u \in [n]} e(g, H(\text{id}_u)) \cdot e\left(\prod_{i=1}^l g_i^{m_i}, g\right)\right)^{\text{mk}}$ 等式是否成立来验证聚合结果的正确性。

$$\begin{aligned}
\sigma &= \prod_{u \in [n]} e(\sigma_u, \text{ak}_u) = \\
& \prod_{u \in [n]} e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{m_i}, \text{ak}_u\right) = \\
& \prod_{u \in [n]} e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{m_i}, (g^{1/\beta_u})^\alpha\right) = \\
& \prod_{u \in [n]} e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{m_i}, g\right)^\alpha = \\
& \left(\prod_{u \in [n]} e(g, H(\text{id}_u))\right)^\alpha \cdot \left(\prod_{u \in [n]} e\left(\prod_{i=1}^l g_i^{m_i}, g\right)\right)^\alpha = \\
& \left(\prod_{u \in [n]} e(g, H(\text{id}_u)) \cdot e\left(\prod_{i=1}^l g_i^{m_i}, g\right)\right)^{\text{mk}} \quad (4)
\end{aligned}$$

如果式 (4) 成立, 则用户能够相信聚合者确实根据用户提供的数据计算了正确的向量聚合; 否则, 用户拒绝接受聚合结果。

定理 2 (不可伪造性): 如果 eBCDH 假设和 BDDH 假设成立, 则本文提出的适用于多方聚合计算的双向认证方法是不可伪造的。

证明 下面从不可伪造性的两个方面分别证明。

1) 证明用户向量签名的不可伪造性, 假设多项式时间敌手 $\mathcal{A}_1$ 能构造出算法 $\mathcal{B}_1$, 可以以不可忽略的概率解决在 (G, G_T) 中的 eBCDH 问题。

设置: 敌手 $\mathcal{A}_1$ 获得系统中所有的公开参数 pp 和 $(\text{id}_u, \text{pk}_u)_{u \in [n]}$ 。

签名问询: 敌手 $\mathcal{A}_1$ 向挑战者 $\mathcal{C}$ 多次询问向量

$\mathbf{M}_1, \mathbf{M}_2, \dots, \mathbf{M}_q$ 的签名, 得到挑战者 $\mathcal{C}$ 运行 Sign 算法生成的签名 $\sigma_1, \sigma_2, \dots, \sigma_q$ 。

伪造签名: 由于任意用户 u 向量签名的计算结果 $\sigma_u = \left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{m_i}\right)^{\text{sk}_u}$ 能够生成看似随机的值, 敌手 $\mathcal{A}_1$ 通过观察得到的多个向量对应的签名 $\sigma_1, \sigma_2, \dots, \sigma_q$, 利用算法 $\mathcal{B}_1$ 以不可忽略的概率输出合理的向量 $\mathbf{M}_u^*$ 以及对应的签名 σ_u^* 。

如果敌手 $\mathcal{A}_1$ 能够取得游戏胜利, 则 Verify $(\text{id}_u, \text{pk}_u, \mathbf{M}_u^*, \sigma_u^*) = 1$ 成立, 即算法 $\mathcal{B}_1$ 能够以不可忽略的概率构造出签名 σ_u^* , 使等式 $e(\sigma_u^*, g) = e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{m_i}, \text{pk}_u\right)$ 成立, 那么敌手 $\mathcal{A}_1$ 一定能根据 σ_u 、 $\text{id}_u = g^{1/\beta_u}$ 和 $\text{pk}_u = g^{\beta_u}$ 得到 β_u , 即以不可忽略的概率解决 eBCDH 问题, 但这与 eBCDH 问题的困难性假设矛盾。所以, 用户向量签名是不可伪造的。

2) 证明向量聚合结果证明的不可伪造性, 假设多项式时间敌手 $\mathcal{A}_2$ 能构造出算法 $\mathcal{B}_2$, 可以以不可忽略的概率解决在 (G, G_T) 中的 BDDH 问题。

设置: 敌手 $\mathcal{A}_2$ 获得系统中所有的公开参数 pp 和 $(\text{id}_u, \text{pk}_u)_{u \in [n]}$ 。

聚合密钥问询: 敌手 $\mathcal{A}_2$ 向挑战者 $\mathcal{C}$ 多次询问身份为 $\text{id}_1, \dots, \text{id}_q$ 的用户的聚合密钥, 得到挑战者 $\mathcal{C}$ 运行 RGen 算法生成的聚合密钥 $\text{ak}_1, \dots, \text{ak}_q$ 。

聚合结果证明问询: 敌手 $\mathcal{A}_2$ 向挑战者 $\mathcal{C}$ 多次询问不同组 $(\mathbf{M}_1^t, \mathbf{M}_2^t, \dots, \mathbf{M}_n^t)_{t \in \{1, \dots, q\}}$ 的聚合结果 $\mathbf{M}$ 的证明, 得到挑战者 $\mathcal{C}$ 依次运行 Sign 和 ASign 算法生成的聚合结果证明 $\sigma_1, \dots, \sigma_q$ 。

伪造聚合结果证明: 敌手 $\mathcal{A}_2$ 根据已知的公开参数 pp 和 $(\text{id}_u, \text{pk}_u)_{u \in [n]}$, 通过观察得到的多个聚合密钥 $\text{ak}_1, \dots, \text{ak}_q$ 和多组不同向量聚合结果证明 $\sigma_1, \dots, \sigma_q$, 利用算法 $\mathcal{B}_2$ 以不可忽略的概率输出一个合理的聚合结果 $\mathbf{M}^*$ 及对应的证明 σ^* 。

如果敌手 $\mathcal{A}_2$ 能够取得游戏的胜利, 那么 $\text{AVerify}(\mathbf{M}^*, \sigma^*, (\text{id}_u)_{u \in [n]}, (g_i)_{i \in [l]}, g, \text{mk}) = 1$ 成立, 即算法 $\mathcal{B}_2$ 能够以不可忽略的概率构造出签名 σ^* , 使 $\sigma^* = \left(\prod_{u \in [n]} e(g, H(\text{id}_u)) \cdot e\left(\prod_{i=1}^l g_i^{m_i}, g\right)\right)^{\text{mk}}$ 成立, 那么敌手 $\mathcal{A}_2$ 能够根据

$\text{id}_1, \dots, \text{id}_q$ 和对应的 $\text{ak}_1, \dots, \text{ak}_q$ 得到 α ，从而可以根据已知的公共参数 pp 计算 $\left(\prod_{u \in [n]} e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{m_i^*}, g\right)\right)^\alpha$ ，即以不可忽略的概率解决 BDDH 问题，但这与 BDDH 问题的困难性假设矛盾。所以，向量聚合结果证明是不可伪造的。

因此，本文提出的适用于多方聚合计算的双向认证方法是不可伪造的。

5.1.2 可验证联邦聚合方法的安全性

本节从聚合结果的正确性和可验证性以及用户模型的来源可信性、数据完整性和隐私保护几个方面具体分析第4.2节提出的基于同态代理重认证的可验证联邦聚合方法的安全性。

(1) 聚合结果的正确性：在本文提出的基于同态代理重认证的联邦聚合方法中，如果每个实体都诚实地执行协议，就可以获得正确的聚合梯度来更新模型。

在本文方法中，如果每个实体诚实地执行协议，则只有当 $\sum_{u \in U_3} X_u = \sum_{u \in U_3} Y_u - \sum_{u \in U_3} P_u + \sum_{u \in U_3, v \in U_2 \cup U_3} P_{v,u}$ 成立时，才能获得正确的聚合梯度，证明该等式成立的过程如下。

$$\begin{aligned} & \sum_{u \in U_3} Y_u - \sum_{u \in U_3} P_u + \sum_{u \in U_3, v \in U_2 \cup U_3} P_{v,u} = \\ & \sum_{u \in U_3} \left(X_u + P_u + \sum_{v \in U_2} P_{u,v} \right) - \\ & \sum_{u \in U_3} P_u + \sum_{u \in U_3, v \in U_2 \cup U_3} P_{v,u} = \\ & \sum_{u \in U_3} X_u + \sum_{u \in U_3} P_u + \sum_{u \in U_3, v \in U_2 \cup U_3} P_{v,u} - \\ & \sum_{u \in U_3} P_u + \sum_{u \in U_3, v \in U_2 \cup U_3} P_{v,u} = \sum_{u \in U_3} X_u \end{aligned} \quad (5)$$

因此，如果方法中的每个实体都诚实地执行协议，就可以获得正确的聚合梯度来更新模型。

(2) 聚合结果的可验证性：在本文提出的基于同态代理重认证的联邦学习聚合方法中，每个参与训练的用户都可以独立验证聚合结果，并且通过验证可以以压倒性的概率检测出伪造的聚合结果。

假设在计算聚合结果—生成证明阶段，恶意服务器将 $\sum_{u \in U_3} X_u$ 替换成 $\sum_{u \in U_3} X_u^*$ 来伪造聚合结果。同时，由于聚合结果证明的计算结果能够生

成看似随机的值，聚合服务器通过伪造出对应的聚合结果证明 σ^* 来试图通过用户对聚合结果的验证。在本文方法中，用户在本地保存验证密钥 mk ，然后仅将计算得到的聚合密钥 $\text{ak}_u = (\text{id}_u)^{\text{mk}}$ 作为公开信息发送给聚合服务器，基于在 (G, G_T) 中的 BDDH 困难问题，聚合服务器无法根据 ak_u 推测出 mk 的有关信息。因此，当用户对 $(\sigma^*, \sum_{u \in U_3} X_u^*)$ 进行验证时，根据伪造的全局模型 $\sum_{u \in U_3} X_u^*$ 和伪造的签名 σ^* 无法使等式 $\sigma^* = \left(\prod_{u \in U_3} e(g, H(\text{id}_u)) \cdot e\left(\prod_{i=1}^l g_i^{x_i^*}, g\right)\right)^{\text{mk}}$ 成立。聚合服务器伪造出一对合理的全局模型聚合结果以及证明能通过用户的验证在计算上是不可行的。也就是说，聚合服务器伪造出的证明 σ^* 不能以不可忽略的概率通过用户的验证，除非服务器获得了全部用户私钥和验证密钥 mk 。若聚合服务器能够以不可忽略的概率通过验证者的验证，那么他就必定能解决 BDDH 问题，但这与困难问题假设矛盾。如果聚合服务器想要聚合结果能通过用户的验证，那么就必须按照要求对所有参与训练的用户梯度进行正确的聚合计算。

因此，本文提出的基于同态代理重认证的可验证联邦聚合方法的聚合结果是可验证的。

(3) 用户模型的来源可信性和数据完整性：聚合服务器在进行模型参数聚合之前需要先验证所有的模型向量是否来自该用户，以及该模型向量是否具有数据完整性。

在加密—签名阶段，每个用户对其模型向量签名 $\sigma_u^y = \left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{y_u}\right)^{A_u^{\text{sk}}}$ 。在验证用户签名阶段，服务器根据算法 $\text{Verify}(\text{id}_u, A_u^{\text{pk}}, Y_u, \sigma_u^y)$ ，通过判断 $e(\sigma_u^y, g) = e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{y_u}, A_u^{\text{pk}}\right)$ 等式是否成立，来验证每个用户的签名 σ_u^y 。

由于模型向量签名 σ_u^y 是由用户 u 利用其私钥 A_u^{sk} 通过安全的 Sign 算法对 Y_u 签名得到的。基于4.1节双向认证方法的正确性，只有完整的模型向量 Y_u 才能使等式 $e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{y_u}, A_u^{\text{pk}}\right) = e(\sigma_u^y, g)$ 成立，保证了该用户向量 Y_u 的数据完整

性。同时,在判断等式 $e\left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{y_u}\right)$, $A_u^{\text{PK}} = e(\sigma_u^y, g)$ 是否成立时,需要用到该用户的公钥 A_u^{PK} ,基于 4.1 节的双向认证方法的不可伪造性,攻击者无法伪造出合理的向量 $\mathbf{Y}_u$ 和签名 σ_u^y 通过验证,保证了该用户模型向量 $\mathbf{Y}_u$ 的来源可信性。

(4) 用户模型的隐私保护:本文方法中的隐私保护指的是保护所有参与训练的用户模型梯度参数在每个阶段中都不被泄露。

如果聚合服务器想试图根据加密向量 $\mathbf{Y}_u$ 得到原始用户向量 $\mathbf{X}_u$,则需要得到 $\sum_{v \in U_2} P_{u,v}$ 和 $\mathbf{P}_u$ 。但 $S_{u,v}$ 是通过用密钥协商 $\text{KA.agree}(S_u^{\text{SK}}, S_v^{\text{PK}})$ 得到的, b_u 是用户 u 在本地随机生成的,对聚合服务器是保密的,因此聚合服务器无法通过 $\mathbf{P}_{u,v} = \Delta_{u,v} \cdot \text{PRG}(S_{u,v})$ 和 $\mathbf{P}_u = \text{PRG}(b_u)$ 恢复出任意用户的加密掩码 $\sum_{v \in U_2} P_{u,v}$ 和 $\mathbf{P}_u$ 。此外,基于 (G, G_T) 中的 eBCDH 困难问题,聚合服务器无法根据向量签名 $\sigma_u^x = \left(H(\text{id}_u) \cdot \prod_{i=1}^l g_i^{x_u}\right)^{A_u^{\text{SK}}}$ 得到原始用户梯度向量 $\mathbf{X}_u$ 的有关信息。

当用户在中途掉线时,服务器则会认为这些用户处于离线状态,在聚合结果阶段要求其他在线用户上传这些用户私钥的秘密共享以去掉聚合梯度中多余的掩码。如果这时通信恢复,这些用户又成功上传了加密梯度,聚合服务器利用足够多掉线用户的秘密共享 $\{(S_{v,u}^{\text{SK}})\}_{v \in U_2 \setminus U_3}$ 恢复出来 S_u^{SK} ,从而计算出第 1 层掩码的密钥种子 $S_{u,v} \leftarrow \text{KA.agree}(S_u^{\text{SK}}, S_v^{\text{PK}})$,但聚合服务器只能接收到所有在线用户的秘密分享 $\{(b_{v,u})\}_{u \in U_3}$,无法恢复出掉线用户第 2 层掩码的密钥种子 b_u 。因此,由于第 2 层掩码 $\mathbf{P}_u$ 的存在,在去掩码时聚合服务器根据收集的 secret 共享也无法恢复出原始用户梯度向量 $\mathbf{X}_u$ 。因此,所有参与训练的用户模型梯度参数在每个阶段中都不会泄露给聚合服务器。

5.2 性能评估

5.2.1 实验设置

本文通过在配备 R9-5900HX CPU (4.6 GHz) 和 32 GB 内存的 Ubuntu18.04.6 系统上模拟用户和聚合服务器,使用 Python 3.6.9 编写程序,来

进行仿真实验。本文方法不影响模型训练质量,重点关注安全聚合的验证部分,通过 Numpy 工具包生成模拟数据集。签名验证算法中的双线性对有关操作和运算均基于 Pypbc 库实现,采用 a 类曲线,参照 PBC Library Manual 0.5.14,安全参数设置为 qbits=512 和 rbits=160。方法中的对称加密算法使用 Cryptodome.Cipher 库中的 AES 函数实现,并用 Cryptodome.Random 库中的 get_random_bytes 函数生成 16 bit 随机数作为对称加密的密钥和参数;哈希函数算法使用 Cryptodome.Hash 库中的 SHA256 函数来实现;Diffie-Hellman 密钥协商算法基于 Diffiehellman 库实现;t-out-of-N Shamir 秘密分享算法使用 Secretsharing 库中的 SecretSharer 函数实现;随机数生成器算法使用 RandomState 函数实现。

5.2.2 计算开销

通过比较时间的消耗来评估当用户总数变化时,不同方法在每个阶段的计算开销,如表 1 所示。在初始化阶段,VerifyNet^[7]、VeriFL^[8]和本文方法去掉了文献[5]提出的安全聚合 (secure aggregation, SA) 方法中的一致性检测部分,节省了计算开销。在密钥分享阶段,3 个方法计算开销差异不大。在加密-签名阶段,为了支持聚合梯度的验证,在 VerifyNet、VeriFL 和本文方法中,都需要用户计算附加信息,但本文方法相比 VerifyNet 和 VeriFL,仅比没有验证功能的 SA 增加少量计算开销。在 VerifyNet、VeriFL 和本文方法的聚合阶段中,聚合服务器为了证明自己的聚合梯度的正确性,还需要根据用户发送的附加信息生成证明。与 SA 相比可以看出,为了实现对用户梯度向量和聚合结果的验证,本文方法增加的验证计算开销在合理的承受范围内,且相比 VerifyNet 和 VeriFL 方法更节省计算开销。在验证聚合结果阶段,在本文方法中只需使用计算量较轻的双线性对和模指数运算,计算开销要远小于 VerifyNet 和 VeriFL,随着用户数从 100 增加到 1 000,计算开销仅从 1 000 ms 增加到 8 000 ms,而 VerifyNet 和 VeriFL 中验证聚合结果阶段的计算开销大约占总时间的一半。

5.2.3 通信开销

验证过程中的通信开销主要体现在:①加密-签名阶段用户向服务器上传模型梯度签名的

表1 计算开销比较
Table 1 Comparison of computation cost

方法	用户数	计算开销/ms						总时间
		初始化	密钥分享	加密一签名	验证用户签名	梯度聚合一生成证明	验证聚合结果	
SA	100	108	397	268	—	291	—	1 064
	500	607	4 149	3 680	—	33 470	—	41 907
	1 000	1 461	15 861	14 783	—	45 507	—	77 612
VerifyNet	100	6	417	4 538	—	1 273	5 108	11 342
	500	33	4 052	22 047	—	6 140	24 186	50 318
	1 000	69	15 650	41 862	—	11 507	45 940	115 055
VeriFL	100	13	406	3 541	—	876	3 590	8 426
	500	67	4 144	17 304	—	4 087	16 865	42 467
	1 000	147	14 988	35 019	—	7 855	33 164	91 173
本文方法	100	5	586	2 796	2 799	359	1 372	7 918
	500	37	4 195	12 842	6 571	3 745	4 739	32 129
	1 000	61	15 726	25 170	9 849	14 087	8 044	72 937

通信开销；②聚合一生成证明阶段服务器返回给用户的全局模型聚合结果证明的通信开销。假设无用户中途退出，即需要验证的本地梯度数量与用户数量相同。当梯度向量的维度设置为100时，本文方法与SA、VerifyNet和VeriFL在加密一签名阶段和梯度聚合阶段的通信开销比较如表2所示。

表2 通信开销比较
Table 2 Comparison of communication overhead

方法	通信开销/KB	
	加密一签名阶段	聚合一生成证明阶段
SA	56.88	56.88
VerifyNet	105.68	87.21
VeriFL	79.86	68.96
本文方法	63.95	59.84

由表2可以看出，本验证方法并没有给基本的安全聚合协议引入太多额外的通信开销，且比VerifyNet和VeriFL方法表现得更好。根据SA的实验数据可以推断用于传输梯度的通信开销大致为56.88 KB，在整体通信开销中所占比例最大，验证协议带来的额外开销是可以接受的。

5.2.4 其他方面

为了便于比较模型梯度向量的维度变化对方

法性能的影响，控制其他条件不变，当用户数为500、用户退出率为0时，用所耗费的时间来表示计算开销。模型维度变化时，签名开销比较如图3所示，验证开销比较如图4所示。当模型维度从100增加到500时，本文方法签名的计算开销仅从2 944 ms增加至16 568 ms，验证的计算开销仅从4 739 ms增加至22 987 ms。虽然签名和验证时的计算开销都随着维度呈线性增长，但与方法VerifyNet和VeriFL相比开销变化并不明显。

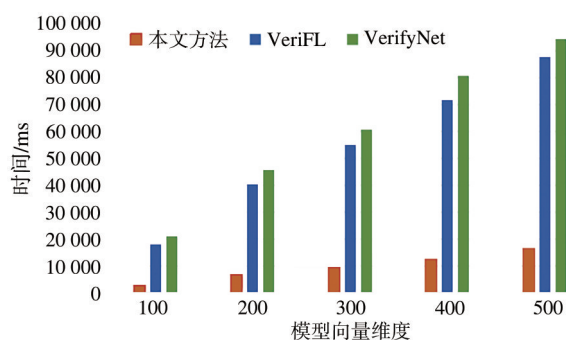


图3 模型维度变化时的签名开销比较
Figure 3 Comparison of signature cost when model dimension changes

当用户退出率变化时，本文方法在用户退出率变化时，计算开销比较如表3所示。可以看出，随着用户退出率的变化，用户在签名和验证

时的计算开销并没有明显变化。本验证方法的性能不受用户退出的影响,用户只在本地生成模型梯度的签名,不受其他中途退出的用户影响,只有当聚合成功时服务器才会将聚合结果和证明一同返回给用户验证结果。

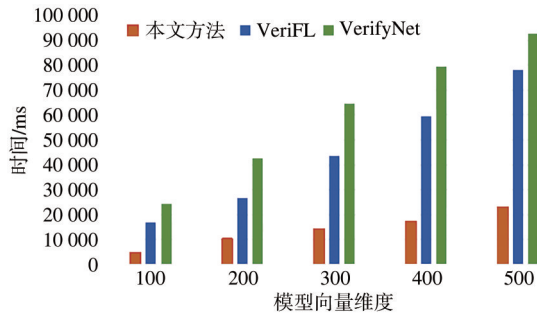


图4 模型维度变化时的验证开销比较
Figure 4 Comparison of verification cost when model dimensions changes

表3 用户退出率变化时的计算开销比较
Table 3 Comparison of computation overhead when user exit rate changes

退出率	计算开销/ms	
	加密—签名阶段	验证聚合结果阶段
0%	12 842	4 739
10%	13 024	4 698
20%	12 907	4 804

5.3 功能对比

各方法功能对比如表4所示,从全局模型果验证、用户模型验证、隐私保护和支持用户退出4个方面将本文方法与现有的可验证联邦学习方法进行对比。文献[5]基于双掩码技术解决了联邦学习中用户退出时的隐私保护问题,但不能实现全局模型聚合结果的验证。VerifyNet、VeriFL基于同态哈希的性质实现了全局模型聚

合结果的验证,并利用双掩码技术支持中途退出用户的模型隐私保护。Fu等^[13]结合单掩码和拉格朗日插值技术,提出了具有隐私保护的可验证高效联邦学习方法(VFL),不仅实现了用户梯度的隐私保护,还能实现全局模型的隐私保护,但当有用户在训练过程中频繁掉线或退出时,方法性能大幅下降。本文方法相比同类方法,实现的功能更加全面,通过结合同态代理重认证和双掩码技术实现了全局模型聚合结果的正确性验证,还能实现用户模型梯度的来源可信性和完整性验证,同时在验证过程中还支持用户梯度的隐私保护。

6 结束语

本文首次引入同态代理重认证,提出了具有隐私保护、高效可信的联邦聚合方法。方法利用同态代理重认证在多用户数据聚合场景下提供具有鲁棒性的可验证安全保证,在验证全局模型聚合结果正确性的同时,还能实现对用户上传模型的来源可信性和完整性验证,且在验证过程中还支持用户梯度的隐私保护。最后,通过形式化的分析验证了方法的安全性,并通过仿真实验的性能评估,验证了方法是可行且高效的。

参考文献:

- [1] MCMAHAN H B, MOORE E, RAMAGE D, et al. Communication-efficient learning of deep networks from decentralized data[J]. Artificial Intelligence and Statistics. 2017: 1273-1282.
- [2] YANG Q, LIU Y, CHEN T J, et al. Federated machine learning: concept and applications[J]. ACM Transactions on Intelligent Systems and Technology, 2019, 10(2): 12.

表4 各方法功能对比
Table 4 Comparison of scheme function

方法	全局模型验证	用户模型验证	隐私保护	支持用户退出
SA	×	×	√	√
VerifyNet	√	×	√	√
VeriFL	√	×	√	√
VFL ^[13]	√	×	√	×
本文方法	√	√	√	√

- [3] LI T, SAHU A K, TALWALKAR A, et al. Federated learning: challenges, methods, and future directions[J]. IEEE Signal Processing Magazine, 2020, 37(3): 50-60.
- [4] 钱文君, 沈晴霓, 吴鹏飞, 等. 大数据计算环境下的隐私保护技术研究进展[J]. 计算机学报, 2022, 45(4): 669-701.
- QIAN W J, SHEN Q N, WU P F, et al. Research progress on privacy-preserving techniques in big data computing environment[J]. Chinese Journal of Computers, 2022, 45(4): 669-701.
- [5] BONAWITZ K, IVANOV V, KREUTER B, et al. Practical secure aggregation for privacy-preserving machine learning[C]//Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM press, 2017: 1175-1191.
- [6] 周传鑫, 孙奕, 汪德刚, 等. 联邦学习研究综述[J]. 网络与信息安全学报, 2021, 7(5): 77-92.
- ZHOU C X, SUN Y, WANG D G, et al. Survey of federated learning research[J]. Chinese Journal of Network and Information Security, 2021, 7(5): 77-92.
- [7] XU G W, LI H W, LIU S, et al. VerifyNet: secure and verifiable federated learning[J]. IEEE Transactions on Information Forensics and Security. 2019(15): 911-926.
- [8] GUO X J, LIU Z L, LI J, et al. VeriFL: communication-efficient and fast verifiable aggregation for federated learning [J]. IEEE Transactions on Information Forensics and Security. 2021, 1809(16): 1736-1751.
- [9] FUNG C, YOON C J M, Beschastnikh I. The limitations of federated learning in Sybil settings[C]//Proceedings of the 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020). 2020: 301-316.
- [10] REN Y L, LI Y R, FENG G R, et al. Privacy-enhanced and verification-traceable aggregation for federated learning[J]. IEEE Internet of Things Journal, 2022, 9(24): 24933-24948.
- [11] HAHN C, KIM H, KIM M, et al. VerSA: verifiable secure aggregation for cross-device federated learning[J]. IEEE Transactions on Dependable and Secure Computing. 2023, 20(1): 36-52.
- [12] ZHANG Z Z, WU L B, HE D B, et al. G-VCFL: grouped verifiable chained privacy-preserving federated learning[J]. IEEE Transactions on Network and Service Management, 2022, 19(4): 4219-4231.
- [13] FU A M, ZHANG X L, XIONG N X, et al. VFL: a verifiable federated learning with privacy-preserving for big data in industrial IoT[J]. IEEE Transactions on Industrial Informatics, 2022, 18(5): 3316-3326.
- [14] GENNARO R, GENTRY C, PARNO B. Non-interactive verifiable computing: outsourcing computation to untrusted workers[C]//Proceedings of the Annual Cryptology Conference. Berlin: Springer, 2010: 465-482.
- [15] DEMIREL D, SCHABHÜSER L, BUCHMANN J. Homomorphic authenticators[M]. Cham: Privately and Publicly Verifiable Computing Techniques, 2017.
- [16] SONG M K, WANG Z B, ZHANG Z F, et al. Analyzing user-level privacy attack against federated learning[J]. IEEE Journal on Selected Areas in Communications. 2020, 38(10): 2430-2444.
- [17] ZHANG J L, CHEN B, CHENG X, et al. PoisonGAN: generative poisoning attacks against federated learning in edge computing systems[J]. IEEE Internet of Things Journal, 2021, 8(5): 3310-3322.
- [18] BLUM M, MICALI S. How to generate cryptographically strong sequences of pseudo random bits[C]//Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (SFCS). Piscataway: IEEE Press, 1982: 112-117.
- [19] DIFFIE W, HELLMAN M. New directions in cryptography[J]. IEEE Transactions on Information Theory, 1976, 22(6): 644-654.
- [20] LI H W, LIU D X, DAI Y S, et al. Enabling efficient multi-keyword ranked search over encrypted mobile cloud data through blind storage[J]. IEEE Transactions on Emerging Topics in Computing, 2015, 3(1): 127-138.
- [21] SHAMIR A. How to share a secret[J]. Communications of the ACM, 1979, 22(11): 612-613.
- [22] DERLER D, RAMACHER S, SLAMANIG D. Homomorphic proxy re-authenticators and applications to verifiable multi-user data aggregation[C]//Proceedings of the International Conference on Financial Cryptography and Data Security. Cham: Springer, 2017: 124-142.
- [23] BONEH D, FREEMAN D, KATZ J, et al. Signing a linear subspace: signature schemes for network coding[C]//Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography: PKC'09. New York: ACM press, 2009: 68-87.
- [24] BLAZE M, BLEUMER G, STRAUSS M. Divertible protocols and atomic proxy cryptography[M]. Berlin: Berlin Heidelberg, 1998.

[作者简介]



杨帆（1996-），女，天津人，信息工程大学硕士生，主要研究方向为网络与信息安全、可验证计算。



陈性元（1963-），男，安徽无为市人，信息工程大学教授、博士生导师，主要研究方向为网络与信息安全。



孙奕（1979-），女，河南郑州人，信息工程大学教授、博士生导师，主要研究方向为网络与信息安全、数据安全交换。



高琦（1998-），男，湖北襄阳人，信息工程大学硕士生，主要研究方向为联邦学习、隐私保护。