支持高频数据更新的 OKVS-PIR 标签隐私求交方案

薛婧婷[1),2),3)] 李文毅[1)] 李发根[2)] 张文政[3)] 张晓均[1)]

[1)](西南石油大学计算机与软件学院 成都 610500)

[2)](电子科技大学计算机科学与工程学院 成都 611731)

[3)](中国电子科技集团公司第三十研究所保密通信全国重点实验室 成都 610041)

摘 要 标签隐私集合求交(Labeled Private Set Intersection,LPSI)作为 PSI 的重要扩展,不仅延续了传统 PSI 保护集合元素隐私的核心能力,更进一步实现了与元素关联的标签信息(如医疗协作中的患者诊断结果)的安全共享。这一特性使得 LPSI 在跨机构联合分析、安全数据匹配等实际场景中具有关键应用价值。然而,现有 LPSI 方案在应对大规模、高动态数据环境时仍面临显著挑战:一方面,基于全同态加密的方案(如 APSI,CCS'21)尽管提供了强大的安全性保障,但其预处理阶段具 $O(n^2)$ 的复杂度,难以适应数据频繁更新的需求;另一方面,基于不经意键值存储(Oblivious Key-Value Store,OKVS)与批量隐私信息检索(Batch Private Information Retrieval,BatchPIR)的高效混合架构(如 UCPSI,USENIX'24)虽在计算与通信效率上取得重要进展,却缺乏对交集元素对应标签的生成与安全传输机制的原生支持,无法直接满足 LPSI 的核心功能要求。为应对上述挑战,本文提出一种支持动态更新的标签隐私集合求交方案$\mathcal{DLPSI}$,旨在实现数据高频更新与标签实时共享的高效安全统一。本文的核心贡献在于通过系统性的协议重构与技术创新,解决了动态 LPSI 场景下的效率与功能瓶颈。首先重构 OKVS 键值对结构,实现数据-标签的原子化安全传输。安全机制上,$\mathcal{DLPSI}$运用不经意伪随机函数来保护参与方集合内容的隐私性;并创新性地设计基于 HMAC 的盲化与验证机制,有效防止在标签更新与传输过程中可能引发的信息泄露,确保即使在多次动态更新操作下,敌手也无法推断出额外的关联信息,并在半诚实敌手模型下证明方案安全。性能表现上,提出二维哈希表分组匹配策略,将大规模集合的交集计算分解为可并行执行的子任务,显著提升计算吞吐量。在通信优化方面,结合随机带状矩阵 OKVS 构造与高效的 BatchPIR 技术,大幅降低在线阶段的通信负载,使方案能够扩展到百万量级的数据规模。C++实现表明:与 LPSI 方案 PEPSI(USENIX'24)相比,$\mathcal{DLPSI}$在各种数据集规模下的总运行时间减少 90%~96%。相较于 APSI,$\mathcal{DLPSI}$在预处理时间上减少 12%~61%;且在 $n_x=2^{20}$ 规模下,$\mathcal{DLPSI}$的总运行时间比 APSI 减少 19%~50%,充分证明其在处理大规模动态数据时的可扩展性与实用性。本文提出的方案$\mathcal{DLPSI}$已开源在 https://github.com/LiCryptoX/ulpsi。

关键词 标签隐私集合求交;标签共享模型;不经意键值存储;批量隐私信息检索;半诚实敌手模型

中图法分类号 TP309 **DOI 号** 10.11897/SP.J.1016.2026.01153

OKVS-PIR-Based Labeled Private Set Intersection Optimized for High-Frequency Data Updates

XUE Jing-Ting[1),2),3)] LI Wen-Yi[1)] LI Fa-Gen[2)] ZHANG Wen-Zheng[3)] ZHANG Xiao-Jun[1)]

[1)](School of Computer Science and Software Engineering, Southwest Petroleum University, Chengdu 610500)

[2)](School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731)

[3)](National Key Laboratory of Security Communication, Institute of Southwestern Communication, Chengdu 610041)

收稿日期:2025-07-30;在线发布日期:2026-01-07。本课题得到国家自然科学基金(青年基金项目 No. 61902327、面上项目 No. 62272090)、四川省自然科学基金(青年项目 No. 2023NSFSC1398、面上项目 2025ZNSFSC0495)、通信安全重点实验室科技基金(No. 61421030107012102)资助。**薛婧婷**,博士,副教授,中国计算机学会(CCF)会员,主要研究领域为应用密码学、云数据安全审计、区块链技术。E-mail:jtxue@swpu.edu.cn。**李文毅**,硕士研究生,主要研究领域为应用密码学、安全多方计算。**李发根**(**通信作者**),博士,教授,博士生导师,主要研究领域为密码学、信息安全。E-mail:fagenli@uestc.edu.cn。**张文政**,研究员,主要研究领域为密码技术、网络与信息安全。**张晓均**,博士,副教授,中国计算机学会(CCF)会员,主要研究领域为应用密码学、云计算安全、智能电网安全隐私。

Abstract Labeled Private Set Intersection (LPSI), as a significant extension of PSI, not only preserves the core capability of traditional PSI in protecting the privacy of set elements but also enables the secure sharing of label information associated with those elements (e.g., patient diagnostic results in medical collaborations). This feature makes LPSI critically valuable in real-world scenarios such as cross-institutional joint analysis and secure data matching. However, existing LPSI schemes still face substantial challenges in large-scale and highly dynamic data environments. On one hand, schemes based on Fully Homomorphic Encryption (FHE), such as APSI (CCS'21), provide robust security guarantees but suffer from $O(n^2)$ complexity in the preprocessing phase, making it difficult to adapt to frequent data updates. On the other hand, efficient hybrid architectures based on Oblivious Key-Value Store (OKVS) and Batch Private Information Retrieval (BatchPIR), such as UCPSI (USENIX'24), have achieved significant progress in computation and communication efficiency but lack native support for the generation and secure transmission of labels corresponding to intersection elements, failing to meet the core functional requirements of LPSI. To address these challenges, this paper proposes $\mathcal{DLPSI}$, a Dynamic Labeled Private Set Intersection scheme designed to achieve a high-efficiency unification of frequent data updates and real-time label sharing. The core contribution of this work lies in the systematic protocol refactoring and technical innovations that resolve efficiency and functional bottlenecks in dynamic LPSI scenarios. First, we refactor the OKVS key-value structure to achieve atomic and secure transmission of data-label pairs. For the security mechanism, $\mathcal{DLPSI}$ employs Oblivious Pseudorandom Functions (OPRF) to protect the privacy of participants' set contents. Furthermore, we innovatively design an HMAC-based blinding and verification mechanism to effectively prevent information leakage during label updates and transmission. This ensures that even under multiple dynamic update operations, an adversary cannot infer additional associated information. The security of the scheme is proved under the semi-honest adversary model. In terms of performance, we propose a two-dimensional hash table grouping strategy to decompose large-scale intersection computations into subtasks that can be executed in parallel, significantly improving computational throughput. Regarding communication optimization, by combining a random band matrix OKVS construction with efficient BatchPIR technology, the online communication load is substantially reduced, allowing the scheme to scale to million-level data sizes. C++ implementation results demonstrate that, compared to the LPSI scheme PEPSI (USENIX'24), $\mathcal{DLPSI}$ reduces total execution time by 90%～96% across various dataset scales. Compared to APSI, $\mathcal{DLPSI}$ reduces preprocessing time by 12%～61%; at a scale of $n_x=2^{20}$, the total execution time of $\mathcal{DLPSI}$ is 19%～50% less than that of APSI, fully demonstrating its scalability and practicality in processing large-scale dynamic data.

Keywords labeled private set intersection; label sharing framework; oblivious key-value store; batch private information retrieval; semi-honest adversary model

1 引 言

随着大数据技术的普及与应用,数据协调与价值挖掘逐步成为各行业数字化转型的核心驱动力。以医疗健康领域为例,权威医疗机构(如国家级疾控中心、大型三甲医院)已建立起完善的医疗数据智能分析平台,能够基于多源异构临床数据为各类疾病生成精准的诊断标签(如基因分型、个性化治疗方案等)。这些标签的生成不仅依赖于高精度医疗设备,更需要多学科专家团队的协同研判,对提升诊疗精准度、公共事件响应速度具有决定性作用。然而,医疗资源分布通常存在典型的"倒三角"特征:基层医疗机构(如社区医院、临时方舱医院)受限于硬件设

施薄弱和人才短缺,在疾病诊断标签生成能力方面与权威机构存在显著差距。因此,在现代医疗信息化建设过程中,如何实现权威医疗标签向基层的安全高效共享,同时确保患者隐私不被泄露,已成为智慧医疗发展亟待解决的关键问题。

标签隐私集合求交(Labeled Private Set Intersection,LPSI)技术作为非平衡隐私集合求交(unbalanced PSI,uPSI)的前沿方向,为上述问题提供了创新性解决方案。该技术通过密码学协议实现:权威机构作为服务端维护带标签的医疗数据库;基层机构作为客户端提交查询请求,获得精确匹配的医疗标签而不泄露其他信息。这种技术模式在多个医疗场景中展现出显著的应用前景:在传染病防控方面,疾控中心实时更新病例数据库并为每个病例标注传播风险等级,基层检测点通过 LPSI 协议快速获取阳性病例的风险等级,从而实施精准防控;在分级诊疗场景中,三甲医院维护的疑难病例数据库包含详细的诊疗建议标签,基层医院通过 LPSI 协议获取匹配病例的治疗方案,显著提升诊断准确性和效率。这种"数据不动标签动"的 LPSI 共享模式,既能保障数据安全,又能实现优质资源下沉。

尽管如此,现有主流的 uPSI 方案(如 CCS'18,CCS'21,USENIX'24,对应[1-3])在医疗场景下的适用性仍存在显著局限。这些方案大多基于全同态加密(Fully Homomorphic Encryption,FHE)技术构建,通过服务端的预计算(主要是多项式插值运算)来减轻客户端的计算负担。虽然这能实现通信复杂度与客户端数据集规模的线性关系,使其在非平衡计算场景中具有应用优势,但其平方级 $O(n_x^2)$ 的预处理时间复杂度在数据频繁更新的医疗场景中将带来难以承受的资源消耗。以传染病监测为例,疾控中心每日需处理约 5 万至 10 万条新增病例(峰值数据),且疾病标签(如新冠病毒变异株分类)需每日实时更新以同步最新疫情研究进展。现有 uPSI 方案[1,3]基于静态数据假设,完全无法支持标签动态更新;LPSI 方案[2]虽支持动态性,但标签更新计算开销随更新频率呈平方级增长,导致在高频更新场景下适用性失效。

为突破 FHE-PSI 的计算效率限制,学界转向探索新的计算范式。Hao 等人(USENIX'24)[4]创新性地提出融合批量隐私信息检索(Batch Private Information Retrieval,BatchPIR)与不经意键值存储(Oblivious Key-Value Store,OKVS)的电路 PSI 架构。将传统方案中复杂的插值多项式运算转化为高效的线性方程组求解,不仅将预处理时间复杂度优化至线性阶 $O(n_x)$,更通过 BatchPIR 机制显著提升通信效率。然而,该框架在医疗标签共享场景存在固有局限性:其原始设计聚焦传统电路 PSI 场景,完全未考虑标签的生成与传输机制,导致无法直接适配上述场景。因此,针对"标签实时共享"的关键需求,亟须研究新的安全高效 LPSI 方案。

本文提出支持服务端大规模数据高频更新的 LPSI 方案$\mathcal{DLPSI}$,贡献如下:

(1)提出$\mathcal{DLPSI}$。提出面向大规模数据高频更新的 LPSI 方案。原创性构造原子化存储数据-标签对的数据结构,用于随机带状矩阵 OKVS(Random Band matrix OKVS,RB-OKVS)编解码。设计基于 RB-OKVS 与 CWC&-FHE-BatchPIR(基于恒权码和全同态加密的 BatchPIR)的交集计算范式,计算性能优于 FHE-LPSI。

(2)安全提升。重构 OKVS 编码中键-值对数据结构(详见 2.2 技术演进),引入不经意伪随机函数(Oblivious Pseudorandom Function,OPRF)保障服务端数据隐私;提出 HMAC 盲化机制,解决标签更新引入的信息泄露问题。

(3)性能优化。设计基于二维哈希表的分组匹配机制,实现并行化交集计算,大幅提升计算性能;在非平衡设定 $n_x \gg n_y$ 下,本文提出的交集计算范式,相比于 OKVS 传统的黑盒使用方式(如 USENIX'23[5]中),将通信复杂度从 $O(n_x)$ 降至 $O(n_y \sqrt[h]{n_x})$。

(4)安全性证明和性能评估。采用模拟证明范式和混合论证框架,形式化证明$\mathcal{DLPSI}$在半诚实敌手模型下的安全性。使用 C++编程语言系统地实现$\mathcal{DLPSI}$。通过创新的动态数据处理机制,首次实现标签安全共享与高性能的兼容,从根本上解决了动态医疗场景中"标签实时更新"这一核心适用性瓶颈。实验数据表明,在 COVID-19 数据集①上,$\mathcal{DLPSI}$的更新开销比文献[2]降低 19%～50%,显著提升实时响应能力。

2　研究动机

2.1　问题定义

在数字化时代,敏感数据的安全共享与高效利用至关重要。LPSI 作为一项关键的隐私保护技术,允许参与方在不泄露完整数据集的前提下,安全地

① https://www.kaggle.com/datasets/meirnizri/covid19-dataset。

计算交集并获取相关标签。本研究旨在显著提升LPSI的性能与安全，尤其在大规模、高动态场景下。

以“数据不动标签动”的医疗场景为例反映LPSI处理海量、动态敏感数据时面临的普遍技术瓶颈。服务端(如国家健康医疗大数据中心)：标签提供方，维护大规模异构医疗数据集$\{x_i\}_{i\in[n_x]}$，并由专业团队标注结构化标签$\{L_i\}_{i\in[n_x]}$。需支持百万级日处理量的实时预处理和动态标签更新。客户端(如地方医院)：标签需求方，持有小规模查询集$Y=\{y_j\}_{j\in[n_y]}$。通过$\mathcal{DLPSI}$安全获取匹配标签，以支持公共卫生管理、临床决策等应用。即在非平衡设定($n_y \ll n_x$)下的目标：在严格保护客户端查询隐私和服务端数据隐私的前提下，使客户端能安全高效地从服务端数据集$X=\{(x_i,L_i)\}_{i\in[n_x]}$中获取与其查询集$Y$匹配的标签，即实现

$$\mathbb{SET}_{\mathcal{DLPSI}}=\{(x_i,L_i)\in X \mid \exists y_j\in Y: y_j=x_i\} \tag{1}$$

当前大规模、动态LPSI面临以下核心挑战。这些挑战不仅限于医疗领域，在其他敏感数据应用中也普遍存在：

(1)大规模数据集处理：服务端数据集庞大($n_x\approx 2^{20}$)。多数通用LPSI方案因过高的计算(如$O(n_x^2)$)或通信开销(如$O(n_x)$)而难以适用，造成性能瓶颈。

(2)高频动态更新支持：医疗数据更新频繁。现有LPSI方案对服务端数据及标签的周期性更新支持不足，或在更新时引入额外高昂开销及潜在信息泄露风险。

(3)场景适应性与优化不足：通用LPSI方案缺乏针对特定场景数据结构(如医疗标签)和隐私敏感性(如异步交互、资源受限设备)的深度定制，限制了其实际应用的性能、安全性和实用性。

本研究的核心问题：如何设计并实现一个安全高效、支持服务端大规模数据高频更新的LPSI方案？该方案需在严格保障双向隐私(即客户端查询隐私和服务端数据及更新隐私)的前提下，实现标签的安全高效共享。

2.2 技术演进

本小节展示$\mathcal{DLPSI}$的三阶段技术演进路径，原创性地通过OKVS数据结构重构和密码学协议优化，旨在实现安全性与效率的显著提升。

(1)基础协议(存在隐私泄露风险)：采用标准OKVS编码协议，对数据集$X=\{(x_i,L_i)\}_{i\in[n_x]}$进行编码，计算

$$\boldsymbol{S}=OKVS.Encode(x_i,\alpha\|L_i) \tag{2}$$

其中，α为λ位随机数。收到编码结果$\boldsymbol{S}$后，客户端通过解码操作(以$2^{-\lambda}$的错误概率)验证数据交集并提取标签。该协议存在三个关键缺陷：固定前缀α会破坏OKVS的不经意性；$O(n_x)$的通信复杂度效率低下；需要$O(n_xn_y)$的逐项比对计算。

(2)安全增强协议(支持静态标签)：针对基础协议的缺陷，提出三重优化：

①隐私保护增强：引入OPRF技术，计算并划分$OPRF(x_i)=x_i' \| x_i''$，通过伪随机性确保数据隐私。将OKVS编码改进为

$$\boldsymbol{S}=OKVS.Encode(x_i',(\alpha\|L_i)\oplus x_i'') \tag{3}$$

②计算性能提升：设计基于二维哈希表的分组匹配机制，用桶索引$\widetilde{m}$的哈希值$H_3(\widetilde{m})$替代随机数α，实现并行化交集计算。

③通信效率优化：融合RB-OKVS与CWC&FHE-BatchPIR，将通信复杂度从$O(n_x)$降至$O(n_y\sqrt[h]{n_x})$。

(3)动态标签协议$\mathcal{DLPSI}$(支持标签安全更新)：为解决标签更新导致的信息泄露问题，引入基于随机数r的HMAC盲化机制①，将OKVS编码进一步改进为

$$\boldsymbol{S}=OKVS.Encode(x_i',r\|((H_3(m)\|L_i)\oplus HMAC(x_i'',r))) \tag{4}$$

通过动态更新的r，确保即使标签L_i发生变更(如L_1更新为L_2)，*Client*也无法推导出$L_1\oplus L_2$等敏感信息，支持医疗场景下的标签动态更新需求。

3 相关工作

PSI协议起源于Meadows[6]基于Diffie-Hellman密钥交换的开创性工作。DH-PSI协议族[6-8]因其实现简单、通信高效，在小规模数据集上表现出色，但其模指数运算导致计算复杂度呈指数增长。为优化性能，后续研究转向基于多项式插值[9-13]的方法，通过适度增加计算开销来提升通信效率。近年来，基于不经意传输扩展(OTE)[14-17]的PSI协议受到关注，主要依赖对称加密，仅需少量公钥加密。Garimella等人[18]提出的OKVS数据结构统一了现有PSI方案[19-23]的数据范式，并与VOLE-OTE技

① HMAC是一种安全的基于哈希函数的消息认证码算法，通过结合密钥与哈希函数实现消息完整性验证与身份认证，其计算效率高且易于实现，无论选用何种标准哈希函数(如SHA-256)，均能实现快速处理。

术结合,实现了高性能 PSI[5,24-25]。Gong 等人[26]基于 Paillier 加密和高阶剩余类判定性(Decisional Composite Residuosity,DCR)问题构建了 PSI。Li 等人[27]则利用 OKVS 和不经意密钥共享伪随机函数(Oblivious Key-Sharing Pseudorandom Function,OKS-PRF)设计 PSI 并扩展到多方场景。然而,这些方案均针对平衡数据集设计,其线性通信开销在非平衡场景下会导致客户端负载过重。

当前 uPSI 协议[1-3,28-30]主要采用 FHE 技术。其核心思想是服务端预计算多项式,客户端通过 FHE 密文与服务端交互来获取交集。Mahdavi 等人[3]创新性地结合 FHE 和恒权码(Constant Weight Code,CWC),通过任务并行显著降低计算时间,但性能高度依赖硬件并行度。Hao 等人[4]提出的不经意键值检索技术,突破性地整合 OKVS 与 Batch-PIR,避免了耗时的 FHE 多项式运算,特别适合大规模动态数据场景。Kales 等人[31]融合布谷鸟过滤器、混淆电路和 OTE 技术设计了 uPSI 协议。Sun 等人[32]通过设计高效可验证不可预测函数的不经意评估算法,在恶意敌手模型下实现了高效 uPSI。

2018 年,Chen 等人[1]首次提出 LPSI 概念,允许客户端安全获取服务端数据的关联标签。2021 年,Cong 等人[2]引入 Paterson-Stockmeyer 算法,有效降低了计算和通信开销。2024 年,Bienstock 等人[33]基于 BatchPIR 设计 LPSI 方案,通过 OPRF 盲化数据并加密标签,支持客户端通过关键词 BatchPIR 获取标签。2025 年,Liu 等人[34]采用分布式点函数实现性能更优的 LPSI,但引入辅助服务器可能带来新的安全隐患。同年,本团队提出了面向协同数据清洗的 PSI 新变体[35],可在非平衡场景下高效检测“脏”数据,拓展了 PSI 应用边界。但该方案与本文关注的 LPSI 场景不同:(a) 服务端与客户端均拥有数据及标签,侧重定位数据一致但标签不一致的“脏”数据;(b) 采用 FHE 技术范式构建核心计算框架,仍存在 FHE 固有的高计算复杂度问题,难以满足数据高频更新需求。因此,无法直接应用于当前 LPSI 场景。

4　准备工作

4.1　技术背景

(1)布谷鸟哈希。布谷鸟哈希[36]使用 $h>1$ 个哈希函数 $H_1,H_2,\cdots,H_h$ 构造一个紧密的哈希表。当以映射的方式写入数据时,将 e 插入到哈希表下标位置为 $H_i(e)$的桶中,其中 $H_i(e)$是从 $H_1,H_2,\cdots,H_h$ 中随机选择的。如果该桶中已有其他数据 e',则将 e'踢出该桶并写入 e,然后将 e'插入到哈希表下标位置为 $H_j(e')$的桶中,其中 $j\in[h]\backslash i$。循环以上过程直到将最后一个被踢出的数据插入空桶中。

(2)置换哈希。置换哈希[37]通过减小哈希表中存储元素的长度,降低存储空间的占用。例如,将数据 x 映射到表长为 m 的哈希表中时,划分 $x=x_1\|x_2$(其中 x_2 的位长为 $\log_2 m$),只需要将 x_1 存储在序号为 $H(x_1)\oplus x_2$ 的位置。这样,在不损坏数据的情况下,每个元素将少占用 $\log_2 m$ 尺寸的存储空间。

(3)基于 Diffie-Hellman 密钥交换的 OPRF。DH-OPRF[38]允许客户端秘密地获取 $OPRF_k(y)$,其中 y 是接收方的输入,k 是服务端的 DH-OPRF 密钥。形式化地,已知群 Z_p^*,安全哈希函数 $H_1:\{0,1\}^*\to Z_p^*$ 和 $H_2:Z_p^*\to\{0,1\}^{2\lambda}$,其中 p 为大素数且 $|p|=\lambda$。具体如下:

①客户端选择随机数 $\beta\in Z_p^*$,计算 $H_1(y)^\beta$ 并将其发送给服务端;

②服务端选择随机数 $k\in Z_p^*$ 作为 DH-OPRF 密钥,计算 $H_1(y)^{\beta k}$ 并将其发送给客户端;

③客户端计算得 $OPRF_k(y)=H_2(H_1(y)^k)$。

(4)不经意键值存储。OKVS[18]是一种存储键和值映射关系的数据结构,由编码和解码算法构成。形式化地,已知 K 和 V 分别是键和值的集合,具体如下:

①$Encode(I)\to\{\boldsymbol{S},\perp\}$:输入键值对 $I=\{(k_1,v_1),(k_2,v_2),\cdots,(k_n,v_n)\}\in(K\times V)^n$,输出编码结果 $\boldsymbol{S}\in V^t$ 或错误指示符 $\perp$;

②$Decode(\boldsymbol{S},k)\to v$:输入编码结果 $\boldsymbol{S}\in V^t$ 和键 $k\in K$,输出值 $v\in V$。

OKVS 满足下述性质:

①正确性。对 $\forall I\in(K\times V)^n$,有:(a)$Pr[Encode(I)=\perp]\leqslant 2^{-\lambda}$;(b)$Decode(\boldsymbol{S},k)=v$,其中 $\boldsymbol{S}\leftarrow Encode(I)$且 $\boldsymbol{S}\neq\perp$。

②不经意性。以不同的集合 $K_1=\{k_1,k_2,\cdots,k_n\}$,$K_2=\{k'_1,k'_2,\cdots,k'_n\}$作为编码算法的输入,均匀随机地选择$\{(v_1,v_2,\cdots,v_n)\mid v_i\leftarrow V\}$,两个分布$\{\boldsymbol{S}\mid\boldsymbol{S}\leftarrow Encode(\{(k_1,v_1),(k_2,v_2),\cdots,(k_n,v_n)\})$,$\boldsymbol{S}\neq\perp\}$,$\{\boldsymbol{S}'\mid\boldsymbol{S}'\leftarrow Encode(\{(k'_1,v_1),(k'_2,v_2),\cdots,(k'_n,v_n)\})$,$\boldsymbol{S}'\neq\perp\}$计算不可区分。

③稀疏性。$\boldsymbol{S}$ 能划分 $\boldsymbol{S}=\boldsymbol{S}_0\|\boldsymbol{S}_1$,其中 $|\boldsymbol{S}_0|$增长速度严格快于 $|\boldsymbol{S}_1|$。同时,对 $\forall k\in K$,$Decode(\boldsymbol{S},k)=\langle sparse(k)\|dense(k),\boldsymbol{S}_0\|\boldsymbol{S}_1\rangle$,其中

$sparse(k)$输出$|\boldsymbol{S}_0|$长度的稀疏向量，$dense(k)$输出$|\boldsymbol{S}_1|$长度的紧密向量。

④双重不经意性。对任意不同键的集合$K=\{k_1,k_2,\cdots,k_n\}$，均匀随机地选择$\{(v_1,v_2,\cdots,v_n)|v_i\leftarrow V\}$。$Encode(\{(k_1,v_1),(k_2,v_2),\cdots,(k_n,v_n)\})$与$\widetilde{\boldsymbol{v}}$统计不可区分，其中$\widetilde{\boldsymbol{v}}\leftarrow V^t$。

RB-OKVS实例化构造[5]的细节如下所示。

Parameters：安全参数λ，键值对$I=\{(k_i,v_i)\}_{i\in[n]}$，安全哈希函数$H_4:\{0,1\}^*\rightarrow[t-w+1]$，$H_5:\{0,1\}^*\rightarrow\{0,1\}^w$，其中$t$是OKVS编码长度，$w$是随机带长度。

Encode(I)：

(a) Sample。定义函数$row(k)=\{0\}^{H_4(k)-1}\|H_5(k)\|\{0\}^{t-w-H_4(k)+1}\in\{0,1\}^t$生成随机带矩阵的行向量，定义随机带矩阵$\boldsymbol{M}=\begin{bmatrix}row(k_1)\\ \cdots\\ row(k_n)\end{bmatrix}\in\{0,1\}^{n\times t}$。

(b) Solve system-of-equations。按每一行非零元的位置重新排序矩阵$\boldsymbol{M}$的行，用高斯消元法和回代求解线性方程组，计算向量$\boldsymbol{S}$满足$\boldsymbol{M}\cdot\boldsymbol{S}=[v_1,v_2,\cdots,v_n]^T$，其中向量$\boldsymbol{S}$的自由变量设置为随机数。返回$\boldsymbol{S}$。

Decode$(\boldsymbol{S},k)$：

返回$<H_5(k),\boldsymbol{S}[H_4(k),H_4(k)+w-1]>$。

(5)批量隐私信息检索。BatchPIR[39-41]允许客户端从服务端检索多个数据项的同时保证查询内容的隐私性。形式化地，假设服务端存储数据$PL=\{pl_1,pl_2,\cdots,pl_n\}$，客户端检索其中索引为$\bar{I}=\{i_1,i_2,\cdots,i_L\}$的$L$个数据。具体如下：

①$Query(\bar{I})\rightarrow(qu,st)$：客户端输入$\bar{I}$，计算查询消息$qu$和隐私状态符$st$，并将$qu$发送给服务端；

②$Answer(PL,qu)\rightarrow resp$：服务端输入$PL$和$qu$，计算响应消息$resp$，并将其发送给客户端；

③$Extract(st,resp)\rightarrow\overline{PL}$：客户端输入$st$和$resp$，计算检索结果

$$\overline{PL}=\{pl_{i_1},pl_{i_2},\cdots,pl_{i_L}\} \tag{5}$$

BatchPIR满足下述性质：

①正确性。对于任意数据集PL和一组互异的检索索引$\bar{I}$，$Extract(st,Answer(PL,qu))=\overline{PL}$，其中$(qu,st)\leftarrow Query(\bar{I})$。

②客户端查询隐私性。对于所有概率多项式时间敌手$\mathcal{A}$及任意互异的批量查询索引$\bar{I}_1,\bar{I}_2$且$|\bar{I}_1|=|\bar{I}_2|$，

$$Pr[\mathcal{A}(qu)=1\mid(qu,st)\leftarrow Query(\bar{I}_1)]-Pr[\mathcal{A}(qu)=1\mid(qu,st)\leftarrow Query(\bar{I}_2)]\leqslant negl(\lambda) \tag{6}$$

FHE&CWC-BatchPIR[41]实例化方案中涉及下述密码学原语：

恒权码是一种每个码字具有固定汉明权重的二进制分组码，表示为$CWC(l,h)$，其中l为码长，h为汉明权重。可用于快速判断两个数据是否相等，具体如下：

$$\forall x,y\in CWC(l,h),\prod_{y[j]=1}x[j]=\begin{cases}1, & \text{if } x=y\\ 0, & \text{otherwise}\end{cases} \tag{7}$$

概率批处理码(Probabilistic Batch Code，PBC)是一种允许以小概率(e.g，2^{-40})失败换取更低计算与通信开销的编码方案，通过将数据库分布存储于多个桶，支持并发查询并行执行，从而分摊负载、提升性能。具体如下：

①$Encode(DB)\rightarrow DB'$：输入数据库$DB$，输出数据库编码结构$DB'$；

②$GenSchedule(I)\rightarrow I'$：输入查询集合$I$，输出查询编码结构$I'$；

③$Decode(DB',I')\rightarrow element$：输入$DB'$，$I'$，输出查询结果$element$。

全同态加密是一种支持在密文上做任意计算的加密技术，其包含5个同态操作：

$$\begin{aligned}SIMDEnc(x)&\rightarrow[\![x]\!]\\ SIMDDec([\![x]\!])&\rightarrow x\\ SIMDAdd([\![x]\!],[\![y]\!])&\rightarrow[\![x+y]\!]\\ SIMDPmul([\![x]\!],y)&\rightarrow[\![x\times y]\!]\\ SIMDMul([\![x]\!],[\![y]\!])&\rightarrow[\![x\times y]\!]\end{aligned} \tag{8}$$

其中，$[\![x]\!]$表示明文x加密后的密文。

$\mathcal{DLPSI}$的技术框架为：设计OKVS编码数据结构构造基础协议，利用OPRF盲化数据增强安全性，采用哈希映射机制支持并行计算以提升处理性能，并通过BatchPIR机制显著减少通信开销，从而在安全、效率与可扩展性上实现综合优化。

4.2 符号定义

表1给出$\mathcal{DLPSI}$方案设计与性能评估部分涉及的主要符号及其含义。

4.3 安全定义

在静态半诚实敌手模型下，定义$\mathcal{DLPSI}$的安全性[42]。安全假设为，敌手在$\mathcal{DLPSI}$执行之前腐化参与方，被腐化的参与方严格按照协议的规定执行，但试图获取额外的隐私信息。

定义1(计算不可区分性). 已知两个概率系综(Probability Ensemble)$\mathcal{X}=X(n)_{n\in N}$和$\mathcal{Y}=Y(n)_{n\in N}$

表 1　符号表

方案设计部分涉及的符号及含义			
符号	含义	符号	含义
$[n]=\{1,\cdots,n\}$	元素集合	$H_1,H_2,\cdots,H_8$	安全哈希函数
λ,κ	统计安全参数、计算安全参数	t,w	OKVS 编码长度、随机带长度
$\mathcal{T},\mathcal{H}$	二维哈希表、布谷鸟哈希表	m	哈希表长度
$k,\boxed{\beta k}$	DH-OPRF 密钥、密钥辅助参数	β,r	客户端和服务端选择的随机数
ρ	映射函数	$\boldsymbol{S}$	OKVS 编码结果
P	服务端计算的键-值对集合	ZQ	客户端计算的用于获取标签的集合
n_x,n_y	服务端和客户端的数据集基数	$row(\cdot)$	生成随机带矩阵行向量的函数
$X=\{(x_i,L_i)\}_{i\in[n_x]}$	服务端的数据集	U,qu,st	BatchPIR 的检索集合、查询消息、隐私状态符
$Y=\{y_j\}_{j\in[n_y]}$	客户端的数据集	$resp$	BatchPIR 的响应消息
$\|$	位连接运算	$\oplus$	位异或运算
$\mid$	位或运算	$[\![x]\!]$	明文 x 加密后的密文
性能评估部分涉及的符号及含义			
符号	含义	符号	含义
T_{PM}	FHE 明文乘密文运算所需时间	h	CWC 的汉明权重
T_M	FHE 密文乘密文运算所需时间	q,N	FHE 的密文模数、模多项式次数
l	CWC 的长度	B	二维哈希表中每个桶的桶容量

是计算不可区分的,记为$x\approx_c y$。如果任意概率多项式时间区分器 D 不能区分随机变量 $X(n)$和 $Y(n)$的分布,即存在可忽略函数 $negl(n)$,使得公式 9 对充分大的 n 成立,即

$$\left|\underset{x\leftarrow X(n)}{Pr}[D(1^n,x)=1]-\underset{y\leftarrow Y(n)}{Pr}[D(1^n,y)=1]\right| < negl(n) \tag{9}$$

定义 2(半诚实敌手模型下确定性功能函数的安全性). 已知两方确定性功能函数$\mathcal{F}=(\mathcal{F}_1,\mathcal{F}_2)$:$\{0,1\}^*\times\{0,1\}^*\rightarrow\{0,1\}^*\times\{0,1\}^*$。对于输入 InP_1,InP_2 和安全参数 λ,定义协议$\prod$运行过程中第 i 个参与方 P_i 的视图为 $view_i^{\prod}(InP_1,InP_2,\lambda)=(InP_i,r^i,m_1^i,m_2^i,\cdots,m_d^i)$,其中 InP_i 为 P_i 的输入,r^i 为 P_i 内部随机带产生的随机串,$m_1^i,m_2^i,\cdots,m_d^i$ 为 P_i 收到的所有消息。对于静态半诚实敌手 $\mathcal{A}$,在理想世界中若存在模拟器 Sim_1,Sim_2 使得公式 10 成立,则在半诚实敌手模型下协议$\prod$能安全计算功能函数$\mathcal{F}$。

$$\begin{aligned} &Sim_1(InP_1,\mathcal{F}_1(InP_1,InP_2),\lambda)\approx {}_cview_1^{\prod}(InP_1,InP_2,\lambda),\\ &Sim_2(InP_2,\mathcal{F}_2(InP_1,InP_2),\lambda)\approx {}_cview_2^{\prod}(InP_1,InP_2,\lambda) \end{aligned} \tag{10}$$

5　方案设计

如图 1 所示,$\mathcal{DLPSI}$分为准备阶段和线上阶段。准备阶段由服务端独立执行:(1)系统初始化;(2)将数据和标签映射到二维哈希表中,执行 OPRF、OKVS 编码算法来预处理哈希表中的数据,得到编码结果 S。当客户端发起标签共享请求时,客户端和服务端交互执行线上阶段:(1)客户端将数据映射到布谷鸟哈希表中,并基于服务端共享的密钥辅助参数执行 OPRF 算法;(2)客户端使用 BatchPIR 检索 OKVS 编码结果 $\boldsymbol{S}$,然后执行 OKVS 解码算法以获得数据交集和对应标签。至此,客户端在不泄漏隐私数据的前提下获得其对应标签。

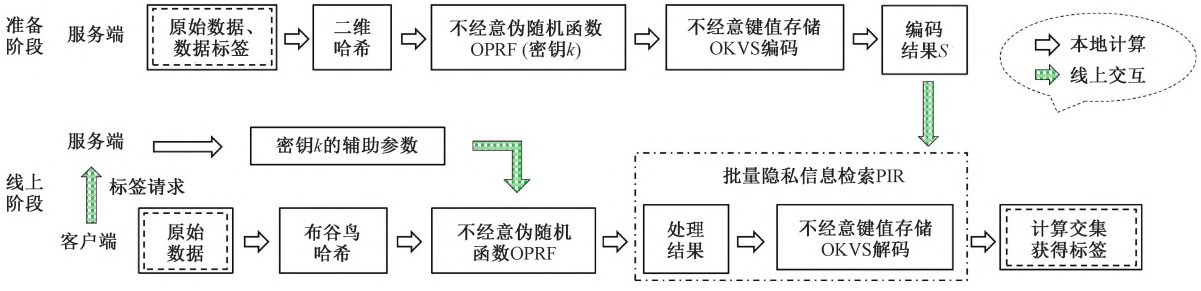


图 1　$\mathcal{DLPSI}$ 执行流程

5.1　准备阶段

服务端独立执行本阶段。生成系统参数,并预处理数据集 $X=\{(x_i,L_i)\}_{i\in[n_x]}$,其中 $x_i\in\{0,1\}^*$ 是数据,$L_i\in\{0,1\}^\ell$ 是对应的标签。

(1)初始化:基于安全参数 λ,生成系统参数 $params$。

①选择群 Z_p^*[①] 其中 p 为大素数且 $|p|=\lambda$。设置哈希表长 m。

②选择安全哈希函数 $H_1:\{0,1\}^* \to Z_p^*$,$H_2: Z_p^* \to \{0,1\}^{2\lambda}$,$H_3:[m] \to \{0,1\}^{\lambda}$,$H_4:\{0,1\}^* \to [t-w+1]$,$H_5:\{0,1\}^* \to \{0,1\}^w$,$H_6$,$H_7$,$H_8$:$\{0,1\}^* \to [m]$,其中 t 是 OKVS 编码长度,w 是随机带长度。

③定义函数 $row(\cdot)=\{0\}^{H_4(\cdot)-1} \| H_5(\cdot) \| \{0\}^{t-w-H_4(\cdot)+1} \in \{0,1\}^t$,用于生成随机带矩阵的行向量。

④公开系统参数 $params=\{Z_p^*, p, H_1, H_2, H_3, H_4, H_5, H_6, H_7, H_8, m, t, w, row(\cdot)\}$。

(2)二维哈希映射:结合置换哈希,将数据集 X 的数据映射到二维哈希表 $\mathcal{T}$ 中。

①初始化一个表长为 m 的空二维哈希表 $\mathcal{T}$,即含 m 个桶。

②对数据-标签对 $(x,L)\in X$ 和安全哈希函数索引 $g\in\{6,7,8\}$(对应 H_6,H_7,H_8),定义 $\tilde{x}=x \| g$。根据置换哈希函数规则,划分数据 $\tilde{x}=\tilde{x}_1 \| \tilde{x}_2$,其中 $|\tilde{x}_2|=\log_2 m$。

③将 $\tilde{x}_1$ 映射到桶 $\mathcal{T}_{H_g(\tilde{x}_1)\oplus\tilde{x}_2}$ 中。

(3)OPRF 与 OKVS 编码:采用 DH-OPRF 盲化 $\mathcal{T}$ 中元素,再使用 OKVS 编码盲化数据。

①选择随机数 k 作为 DH-OPRF 密钥,计算 $\mathcal{T}$ 中元素 $\mathcal{T}_{ij}$ 的 DH-OPRF 盲化值,即 $H_2(H_1(\mathcal{T}_{ij})^k)$ 并划分为 $\mathcal{T}'_{ij} \| \mathcal{T}''_{ij}$,其中 $i\in[m]$,$j\in[|\mathcal{T}_i|]$,$|\mathcal{T}'_{ij}|=|\mathcal{T}''_{ij}|=\lambda$。

②选择随机数 $r_{ij} \leftarrow \{0,1\}^{\lambda}$。定义键-值对

$$P_i:(key_{ij}, value_{ij})=\{(\mathcal{T}'_{ij}, r_{ij} \| ((H_3(i) \| L_{ij}) \oplus HMAC(\mathcal{T}''_{ij}, r_{ij})))\}_{i\in[m], j\in[|\mathcal{T}_i|]} \tag{11}$$

其中,L_{ij} 是 $\mathcal{T}_{ij}$ 对应的标签。

③计算编码结果(具体步骤见算法 1)

$$\boldsymbol{S}=OKVS.Encode(\{P_i\}_{i\in[m]}) \tag{12}$$

算法 1. OKVS.Encode

输入:$\{P_i:(key_{ij}, value_{ij})\}_{i\in[m], j\in[|\mathcal{T}_i|]}$

输出:$\boldsymbol{S}_{t\times 1}$//除了本算法,论文的其他位置将 $\boldsymbol{M}_{\Delta\times t}$,$\boldsymbol{S}_{t\times 1}$ 分别简写为 $\boldsymbol{M}$,$\boldsymbol{S}$

1. FOR $i\in[m], j\in[|\mathcal{T}_i|]$
2. $\quad row_{ij}=row(key_{ij})$
3. ENDFOR
4. 令矩阵 $\boldsymbol{M}_{\Delta\times t}=[row_{11},\cdots,row_{1|\mathcal{T}_1|},row_{21},\cdots,row_{2|\mathcal{T}_2|},\cdots,row_{m1},\cdots,row_{m|\mathcal{T}_m|}]^{\mathrm{T}}$,其中 $\Delta=\sum_{i=1}^{m}|\mathcal{T}_i|$
5. 求解线性方程组,计算向量 $\boldsymbol{S}_{t\times 1}$ 满足 $\boldsymbol{M}_{\Delta\times t}\cdot\boldsymbol{S}_{t\times 1}=[value_{11},\cdots,value_{1|\mathcal{T}_1|},value_{21},\cdots,value_{2|\mathcal{T}_2|},\cdots,value_{m1},\cdots,value_{m|\mathcal{T}_m|}]^{\mathrm{T}}$,其中 $\boldsymbol{S}_{t\times 1}$ 的自由变量设置为随机数
6. RETURN $\boldsymbol{S}_{t\times 1}$

5.2 线上阶段

客户端发起标签共享请求,与服务端交互执行本阶段。客户端拥有查询集 $Y=\{y_j\}_{j\in[n_y]}$,其中 $y_j\in\{0,1\}^*$ 是数据。

(1)布谷鸟哈希映射:由客户端执行,结合置换哈希,将查询集 Y 的数据映射到布谷鸟哈希表 $\mathcal{H}$ 中。

①初始化一个表长为 m 的空布谷鸟哈希表 $\mathcal{H}$,即含 m 个桶。

②对数据 $y\in Y$,随机选择安全哈希函数索引 $g\leftarrow\{6,7,8\}$(对应 H_6,H_7,H_8),定义 $\tilde{y}=y \| g$。根据置换哈希函数规则,划分数据 $\tilde{y}=\tilde{y}_1 \| \tilde{y}_2$,其中 $|\tilde{y}_2|=\log_2 m$。

③将 $\tilde{y}_1$ 映射到桶 $\mathcal{H}_{H_g(\tilde{y}_1)\oplus\tilde{y}_2}$ 中。若映射过程发生哈希碰撞,则踢出旧元素,为其重新选择桶完成映射。重复该过程直到所有元素均插入 $\mathcal{H}$。

④基于前序映射关系,定义映射函数 $\rho:[n_y]\to[m]$,将元素在数据集 Y 中的索引映射到哈希表 $\mathcal{H}$ 中的索引,即对 $\forall j\in[n_y]$,$\mathcal{H}_{\rho(j)}$ 中存放元素对应原数据为 y_j。

(2)DH-OPRF:根据服务端共享的 DH-OPRF 密钥辅助参数,客户端盲化 Y 以获得 Y^o。

①客户端选择随机数 β,计算 $\boxed{\beta}$ 并将其发送给服务端,其中,

$$\boxed{\beta}=\{H_1(\mathcal{H}_{\rho(j)})^{\beta}\}_{\forall j\in[n_y]} \tag{13}$$

②服务端计算密钥辅助参数

$$\boxed{\beta k}=\{H_1(\mathcal{H}_{\rho(j)})^{\beta k}\}_{\forall j\in[n_y]} \tag{14}$$

并将其发送给客户端,其中 k 是 DH-OPRF 密钥。

③客户端计算

$$Y^o=\{H_2(H_1(\mathcal{H}_{\rho(j)})^k)\}_{\forall j\in[n_y]} \tag{15}$$

并划分为 $Y^o=\{\mathcal{H}'_{\rho(j)} \| \mathcal{H}''_{\rho(j)}\}_{\forall j\in[n_y]}$,其中,$|\mathcal{H}'_{\rho(j)}|=|\mathcal{H}''_{\rho(j)}|=\lambda$。

(3)批量检索:由客户端和服务端交互执行,客户端通过 BatchPIR 从编码结果 $\boldsymbol{S}$ 中获取有效信息(如图 2 所示)。

① DDH 假设在该群难解。

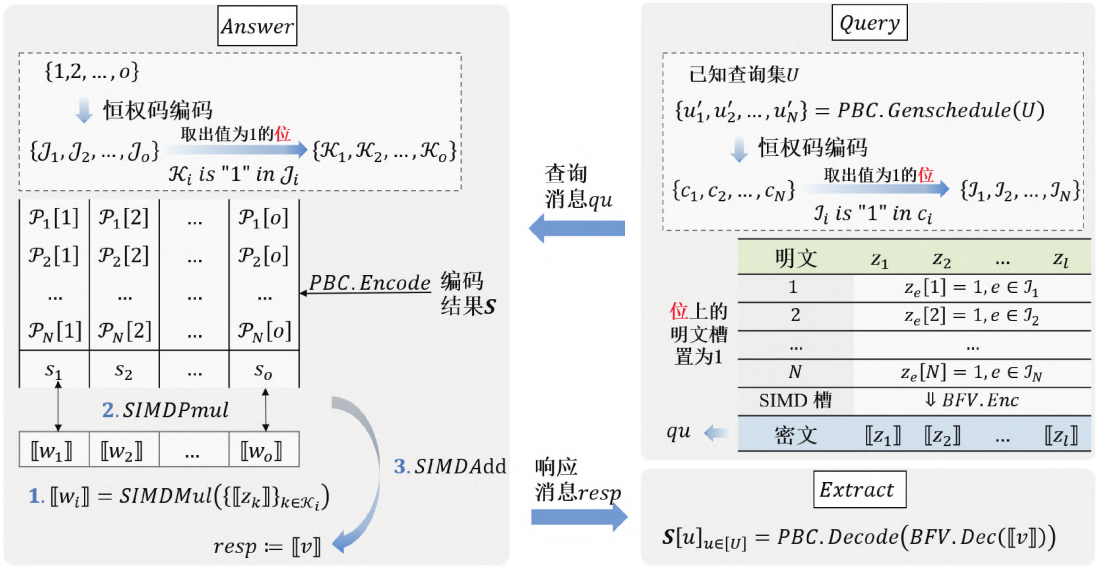


图 2　BatchPIR 执行流程

①客户端构建 OKVS 编码结果 $\boldsymbol{S}$ 中需检索的索引的集合[①]

$$U=\{u \mid (row(\mathcal{H}'_{\rho(1)}) \mid row(\mathcal{H}'_{\rho(2)}) \mid \cdots \mid row(\mathcal{H}'_{\rho(n_y)}))_u = 1\} \tag{16}$$

其中，|表示按位或运算。

②客户端向 U 中填充随机数，使得 $|U|=w\cdot n_y$，保障 BatchPIR 客户端查询隐私。计算 $(qu, st)=BatchPIR.Query(U)$ 并将查询消息 qu 发送给服务端，具体步骤见算法 2。

算法 2. BatchPIR.Query

输入：　$U=\{u_1,\cdots,u_{w\cdot n_y}\}$

输出：　$[\![z_1]\!],\cdots,[\![z_l]\!],u'_1,\cdots,u'_N$//查询消息 $qu:=\{[\![z_1]\!],\cdots,[\![z_l]\!]\}$，隐私状态符 $st:=\{u'_1,\cdots,u'_N\}$

1. $\{u'_1,\cdots,u'_N\}=PBC.GenSchedule(U)$
2. FOR $q\in[N]$
3. u'_q 编码为 $\mathcal{Q}\in CWC(l,h)$
4. $\mathcal{J}=\{i\mid\mathcal{Q}[i]=1, i\in[l]\}$//由 CWC 定义，$|\mathcal{J}|=h$
5. FOR $j\in[l]$
6. IF $j\in\mathcal{J}$ THEN
7. $z_j[q]=1$
8. ELSE
9. $z_j[q]=0$
10. ENDIF
11. ENDFOR
12. ENDFOR
13. FOR $j\in[l]$
14. $[\![z_j]\!]=SIMDEnc(z_j)$
15. ENDFOR
16. RETURN $[\![z_1]\!],\cdots,[\![z_l]\!],u'_1,\cdots,u'_N$

③服务端计算响应消息 $resp=BatchPIR.Answer(\boldsymbol{S},qu)$，并发送给客户端，具体步骤见算法 3。

算法 3. BatchPIR.Answer

输入：　$\boldsymbol{S},[\![z_1]\!],\cdots,[\![z_l]\!]$

输出：　$[\![v]\!]$//响应消息 $resp:=[\![v]\!]$

1. $o=3t/N$
2. 初始化 $\{p_1,\cdots,p_o\}$，其中 $p_j=\{\mathcal{P}_1[j],\cdots,\mathcal{P}_N[j]\}, j\in[o]$；$\{\mathcal{P}_1,\cdots,\mathcal{P}_N\}=PBC.Encode(\boldsymbol{S})$
3. FOR $j\in[o]$
4. j 编码为 $\mathcal{J}\in CWC(l,h)$
5. $\mathcal{K}=\{k\mid\mathcal{J}[k]=1, k\in[l]\}$//由 CWC 定义，$|\mathcal{K}|=h$
6. $[\![w_j]\!]=SIMDMul(\{[\![z_k]\!]\}_{k\in[\mathcal{K}]})$
7. $[\![u_j]\!]=SIMDPmul(\{[\![w_j]\!]\},p_j)$
8. ENDFOR
9. $[\![v]\!]=SIMDAdd([\![u_1]\!],\cdots,[\![u_o]\!])$
10. RETURN $[\![v]\!]$

(4) 计算交集：由客户端执行，计算数据交集和对应标签。

①计算检索结果 $\boldsymbol{S}[u]_{u\in U}=BatchPIR.Extract(st,resp)$，具体步骤见算法 4。

算法 4. BatchPIR.Extract

输入：　$u'_1,\cdots,u'_N,[\![v]\!]$

输出：　$\boldsymbol{S}[u]_{u\in[U]}$//$\boldsymbol{S}[u]$指 t 维向量 $\boldsymbol{S}$ 中第 u 维的数值

1. $\{\mathcal{P}_1[u'_1],\cdots,\mathcal{P}_N[u'_N]\}=SIMDDec([\![v]\!])$
2. $\boldsymbol{S}[u]_{u\in[U]}=PBC.Decode(\{\mathcal{P}_1[u'_1],\cdots,\mathcal{P}_N[u'_N]\})$

① $\mathcal{DLPSI}$采用“通信高效的 OKVS 解码”[43]方式。相较于传统方案中服务端需将完整的 OKVS 编码结果 S 发送给客户端的黑盒交互模式，$\mathcal{DLPSI}$仅需客户端获取 S 中特定索引位置的子集即可完成 OKVS 解码运算。该设计在保证 OKVS 解码算法正确性的前提下大幅减少通信开销。

3. RETURN $\boldsymbol{S}[u]_{u\in[U]}$

②对 $\forall j\in[n_y]$，计算解码结果 $S'_j=OKVS.Decode(\boldsymbol{S}[u]_{u\in[U]},\mathcal{H}'_{\rho(j)})$，具体步骤见算法 5。然后划分 $S'_j=r\parallel r'$，计算 $S'_j=HMAC(\mathcal{H}''_{\rho(j)},r)\oplus r'$，其中 r 是 λ 位长。最后，划分 $S''_j=z_{\rho(j)}\parallel q_{\rho(j)}$，定义集合 $ZQ=\{z_{\rho(1)}\parallel q_{\rho(1)},z_{\rho(2)}\parallel q_{\rho(2)},\cdots,z_{\rho(n_y)}\parallel q_{\rho(n_y)}\}$，其中 $z_{\rho(j)}$ 是 λ 位长。

③对 $\forall j\in[n_y]$，若 ZQ 中 $z_{\rho(j)}=H_3(\rho(j))$，则 $\mathcal{H}_{\rho(j)}$ 中元素对应的数据为交集元素，$q_{\rho(j)}$ 为对应标签。

算法 5. OKVS.Decode

输入：$\boldsymbol{S}[u]_{u\in[U]},\mathcal{H}'_{\rho(j)}$

输出：S'_j

1. $S'_j=\sum_{\{u|row(\mathcal{H}'_{\rho(j)})_u=1\}}\boldsymbol{S}[u]$
2. RETURN S'_j

6 安全分析

6.1 正确性证明

本节推导$\mathcal{DLPSI}$交集计算的正确性。为确保推导的简洁性，不展开描述置换哈希的计算过程。

Case 1：若客户端的查询数据 y 是 X 中的数据，即 $y\in\{x_1,x_2,\cdots,x_{n_x}\}$，则$\mathcal{DLPSI}$保证客户端能够判断其属于交集且获取对应标签。

此时 $\exists i\in[m],j\in[|\mathcal{T}_i|]$ 使得 $\mathcal{T}_{ij}=\mathcal{H}_i$ 且 $x=y$，则有

$$\begin{aligned}\boldsymbol{S}&=OKVS.Encode(\{(\mathcal{T}'_{ij},r_{ij}\parallel((H_3(i)\parallel L_{ij})\\&\oplus HMAC(\mathcal{T}''_{ij},r_{ij})))\}_{i\in[m],j\in[|\mathcal{T}_i|]}),\\r\parallel r'&=OKVS.Decode(\boldsymbol{S},\mathcal{H}'_i)\\&=OKVS.Decode(\boldsymbol{S},\mathcal{T}'_{ij})\\&=r_{ij}\parallel((H_3(i)\parallel L_{ij})\oplus HMAC(\mathcal{T}''_{ij},r_{ij})),\\z\parallel q&=HMAC(\mathcal{H}''_i,r)\oplus r'\\&=HMAC(\mathcal{H}''_i,r_{ij})\oplus HMAC(\mathcal{T}''_{ij},r_{ij})\\&\oplus(H_3(i)\parallel L_{ij})\\&=H_3(i)\parallel L_{ij}\end{aligned}\tag{17}$$

因此，客户端通过 $z=H_3(i)$ 能够判断数据 y 是交集元素且 q 为对应标签。

Case 2：若客户端的查询数据 y 不是 X 中的数据，即 $y\notin\{x_1,x_2,\cdots,x_{n_x}\}$，则$\mathcal{DLPSI}$保证客户端无法获得任何隐私信息。

此时存在唯一的安全哈希函数索引 $g\in\{6,7,8\}$ 使得 $\mathcal{H}_{H_g(y)}\notin\mathcal{T}_{H_g(y)}$。将 $\mathcal{H}'_{H_g(y)}||\mathcal{H}''_{H_g(y)}=DH-OPRF(\mathcal{H}_{H_g(y)})$ 代入公式 17，计算可得 $z\parallel q$ 为随机数，不包含服务端的任何隐私信息。由于哈希碰撞可能导致误报，即存在 $y\notin\{x_1,x_2,\cdots,x_{n_x}\}$ 且 $z=H_3(i)$ 的情形。但误报率为 $2^{-\lambda}$，可忽略不计。

综合 Case 1 和 Case 2 所述，可验证交集计算的正确性。

6.2 安全性证明

在静态半诚实敌手模型下，证明协议 $\Pi_{\mathcal{DLPSI}}$ 安全性。图 3 展示证明过程中涉及的理想功能函数。

> 参数设定：协议参与方服务端和客户端的数据集基数分别为n_x,n_y且$n_x\gg n_y$。
> 功能函数：服务端输入$X=\{(x_i,L_i)\}_{i\in[n_x]}$，客户端输入$Y=\{y_j\}_{j\in[n_y]}$。服务端的输出为空，客户端输出$\{(x_i,L_i)\in X|\exists y_j=x_i\}$。

图 3 理想功能函数 $\mathcal{F}_{\mathcal{DLPSI}}$

定理 1. $\mathcal{DLPSI}$构造的 OKVS 编码结构具备双重不经意性。

证明：即证[OPRF 与 OKVS 编码]构造的 P_i 中所有 $value_{ij}=r_{ij}\parallel((H_3(i)\parallel L_{ij})\oplus HMAC(\mathcal{T}''_{ij},r_{ij}))$ 均匀随机分布。$\mathcal{T}''_{ij}\leftarrow\{0,1\}^\lambda$ 作为 OPRF 输出值满足均匀随机分布特性，对于任意输入 $r_{ij}\leftarrow\{0,1\}^\lambda$ 和密钥 $\mathcal{T}''_{ij}$，HMAC 作为单向伪随机函数，其分布与真随机函数计算不可区分。同时 r_{ij} 独立均匀随机选取，且 $(H_3(i)\parallel L_{ij})\oplus HMAC(\mathcal{T}''_{ij},r_{ij})$ 保持均匀随机性。因此 $value_{ij}$ 均匀随机分布，即$\mathcal{DLPSI}$构造的 OKVS 编码结构具备双重不经意性。

定理 2. 若 BatchPIR 原语具备客户端查询隐私性且 OKVS 原语具备双重不经意性，则在半诚实敌手存在情况下，在 $\mathcal{F}_{OPRF}$ 混合模型中，协议 $\Pi_{\mathcal{DLPSI}}$ 能够安全计算功能函数 $\mathcal{F}_{\mathcal{DLPSI}}$。

证明：构造模拟器 Sim_{Client} 和 Sim_{Server}，分别用于模拟腐化客户端和服务端的视图。通过标准混合论证方法，证明模拟器生成的视图与真实协议执行的视图计算不可区分。

Case 1：敌手腐化客户端。

$Sim_{Client}(Y,(\rho,ZQ),\lambda)$ 模拟腐化客户端的敌手的视图，执行以下两步准备工作：(a) 对 $\forall j\in[n_y]$，读取哈希表 $\mathcal{H}$ 中的数据 $\mathcal{H}_{\rho(j)}$，均匀随机选取 $y_j^*=y_j^{*'}\parallel y_j^{*''}\leftarrow\{0,1\}^{2\lambda}$，其中 $|y_j^{*'}|=|y_j^{*''}|=\lambda$。调用 DH-OPRF 模拟器 $Sim_{DH-OPRF}^{Client}(\{(\mathcal{H}_{\rho(j)},y_j^*)\}_{j\in[n_y]})$，把输出加入 Sim_{Client} 的视图。(b) 均匀随机选取 $\boldsymbol{S}'\leftarrow V^t$，使得 $OKVS.Decode(\boldsymbol{S}',y_j^{*'})=r_{\rho(j)}\parallel r'_{\rho(j)}$，满足 $r'_{\rho(j)}\oplus HMAC(y_j^{*''},r_{\rho(j)})=z_{\rho(j)}\parallel q_{\rho(j)}$。然后，按照协议计算 $(qu,st)=BatchPIR.Query(U)$ 和

$resp=BatchPIR.Answer(\boldsymbol{S}',qu)$,把响应消息 $resp$ 加入 Sim_{Client} 的视图。这里采用标准的混合论证(*Hybrid Argument*)方法,定义三个混合交互记录(*Hybrid Transcript*)T_0,T_1,T_2,证明 Sim_{Client} 与真实敌手的视图计算不可区分。具体如下:

(1) $Hybrid_0$。令 T_0 表示客户端的真实视图。$Hybrid_0$ 是协议 $\prod_{\mathcal{DLPSI}}$ 真实的执行过程。诚实的服务端输入 X 与被敌手腐化的客户端交互执行协议。

(2) $Hybrid_1$。定义 T_1 与 T_0 一样,除了 DH-OPRF 被替换为 DH-OPRF 中接收方的模拟器 $Sim^{Client}_{DH-OPRF}(\{(\mathcal{H}_{\rho(j)},y_j^*)\}_{j\in[n_y]})$,其中 $y_j^*\leftarrow\{0,1\}^{2\lambda}$。模拟器 $Sim^{Client}_{DH-OPRF}$ 的安全(规约至离散对数困难问题)保证 T_1 与 T_0 是计算不可区分的。

(3) $Hybrid_2$。令 T_2 表示 Sim_{Client} 的输出。定义 T_2 与 T_1 一样,除了均匀随机选取 $\boldsymbol{S}'\leftarrow V^t$,使得 $OKVS.Decode(\boldsymbol{S}',y_j^{*'})=r_{\rho(j)}\|r'_{\rho(j)}$,满足 $r'_{\rho(j)}\oplus HMAC(y_j^{*''},r_{\rho(j)})=z_{\rho(j)}\|q_{\rho(j)}$。$Sim_{Client}$ 按照协议规定执行计算查询消息 qu 并在本地计算 $resp=BatchPIR.Answer(\boldsymbol{S}',qu)$。由于 OKVS 具有双重不经意性,$Sim_{Client}$ 选取的 $\boldsymbol{S}'$ 和真实协议执行的计算出来的 $\boldsymbol{S}$ 是计算不可区分的。因此,T_2 与 T_1 计算不可区分。

因此,T_0 与 T_2 是计算不可区分的,即

$$Sim_{Client}(Y,(\rho,ZQ),\lambda)\approx_c view\ \prod^{\mathcal{DLPSI}}_{Client}(X,Y,\lambda) \tag{18}$$

Case 2:敌手腐化服务端。

$Sim_{Server}(X,\perp,\lambda)$模拟腐化服务端的敌手的视图,执行以下两步准备工作:(a) 对 $\forall i\in[m],j\in[|\mathcal{T}_i|]$,选择随机数 $r_{ij}\leftarrow\{0,1\}^{\lambda}$,执行准备阶段的[OPRF 与 OKVS 编码],计算$\{P_i\}_{i\in[m]}$。然后选择随机数 k 作为 DH-OPRF 密钥,调用 DH-OPRF 中发送方的模拟器 $Sim^{Server}_{DH-OPRF}(k,\perp)$,并将输出加入视图。(b) 均匀随机选取集合 $U=\{u_1,u_2,\cdots,u_{w\cdot n_y}\}\leftarrow[t]^{w\cdot n_y}$,计算$(qu,st)=BatchPIR.Query(U)$,并把查询消息 qu 加入视图。

类似地,基于标准的混合论证方法,定义三个混合交互记录 T_0,T_1,T_2,证明 Sim_{Server} 与真实敌手的视图计算不可区分。具体如下:

(1) $Hybrid_0$。令 T_0 表示服务端的真实视图。$Hybrid_0$ 是协议真实的执行过程。诚实的客户端输入 Y 与被敌手腐化的服务端交互协议。

(2) $Hybrid_1$。定义 T_1 与 T_0 一样,除了 DH-OPRF 被替换为 DH-OPRF 中发送方的模拟器 $Sim^{Server}_{DH-OPRF}(k,\perp)$。模拟器 $Sim^{Server}_{DH-OPRF}$ 的安全性保证 T_1 与 T_0 计算不可区分。

(3) $Hybrid_2$。令 T_2 表示 Sim_{Client} 的输出。定义 T_2 与 T_1 一样,除了拟检索的索引集合 $U\leftarrow[t]^{w\cdot n_y}$。由于 BatchPIR 具有客户端查询隐私性,因此 T_2 与 T_1 计算不可区分。

因此,T_0 与 T_2 计算不可区分,即

$$Sim_{Server}(X,\perp,\lambda)\approx_c view\ \prod^{\mathcal{DLPSI}}_{Server}(X,Y,\lambda) \tag{19}$$

基于定义 2,综合 Case 1 和 Case 2 所述,可验证该定理的正确性。

7　性能评估

7.1　实验设计

本节通过系统性对比实验评估$\mathcal{DLPSI}$在非平衡医疗数据集场景下的性能表现与可扩展性。实验环境及配置参数如表 2 所示。特别地,$\mathcal{DLPSI}$支持任意长度数据的灵活处理,所有参数可通过配置文件动态调整,确保实验结果的全面性与可复现性。

表 2　实验环境

类别	参　数
硬件环境	Intel Core i7-9750H (2.60GHz),16GB RAM
软件环境	VMware Workstation 16 Pro,Ubuntu 20.04(分配 8 核 CPU,8GB 内存,300GB 存储)
代码实现	C++(总代码量 12,032 行,含核心算法 9,200 行,注释 1,224 行)
线程配置	$T\in\{1,4,8\}$
安全参数	$\kappa=128$(计算安全),$\lambda=40$(统计安全)
数据维度	服务端数据集 $n_x\in\{2^{20},2^{18}\}$,客户端查询集 $n_y\in\{2^{10},2^8,2^6\}$
数据格式	数据:64 字节随机字符串 标签:6 字节随机字符串

接下来,详细阐述系统实现中的四个核心算法模块及其优化策略。

(1)哈希表构造模块。采用二维哈希表$\mathcal{T}$(服务端)与布谷鸟哈希表$\mathcal{H}$(客户端)的双重数据结构设计。参照文献[28](CCS'17)标准参数配置,设置 3 个哈希映射函数,表长 $m=8192$,桶容量 $B=556$,确保$\mathcal{H}$最大负载 5535 元素且构造失败概率小于 $2^{-\lambda}$。通过置换哈希对原始医疗数据进行压缩编码,旨在大幅提升存储空间利用率。

(2)安全盲化模块。集成 Microsoft APSI① 的

① https://github.com/microsoft/APSI。

高性能 *Four*ℚ 椭圆曲线库，实现 DH-OPRF。该曲线具备高效的随机点标量乘法和哈希到曲线(hash-to-curve)算法。旨在减少数据盲化步骤的耗时。

(3)OKVS 编解码模块。为适配与 BatchPIR 协议的协同运算需求，重构 Google Research 的 OKVS 实现①。关键技术参数包括：错误率 $\varepsilon=0.05$ 的安全-效率平衡点设置；(b) 通过 AVX2 指令集实现 SIMD 并行化，提升矩阵运算速度，支持 $n_x=2^{20}$ 规模数据集的实时处理。

(4)批量查询模块。改进 PIRANA 框架②，主要优化包括：(a)启用压缩模式减少通信开销；(b)禁用 BFV-SIMDRotate 以适配 128 位无符号整型的 OKVS 编码结构，旨在提升查询吞吐量。

系统实现严格采用 128 位无符号整型作为基础数据类型，确保各模块间的数据兼容性。编译本系统要求 GCC 版本不低于 11。

7.2 特性对比与复杂度分析

表 3 系统对比$\mathcal{DLPSI}$与几项代表性 PSI 方案在关键技术特性上的差异，涵盖基于 FHE，OPRF，OKVS，BatchPIR 等密码学原语的实现方案。在现有 LPSI 方案中，$\mathcal{DLPSI}$在计算复杂度方面达到最优，预计算与总计算时间均低于现有代表性方案。

表 3 代表性 PSI 方案特性对比

方案	密码技术	通信复杂度	计算复杂度	非平衡	LPSI
Chen[1]	OPRF，FHE	$O(n_y\log_2 n_x)$	$O(n_x^2)$	√	√
Gong[26]	AHE，DCR	$O(n_x)$	$O(n_x^2\log_2 n_x)$	×	×
Cong[2]	OPRF，FHE	$O(n_y\log_2 n_x)$	$O(n_x^2)$	√	√
Bienstock[5]	OKVS	$O(n_x)$	$O(\lambda n_x)$	×	×
Mahdavi[3]	FHE，CWC	$O(n_y\sqrt[h]{n_x})$	$O(n_x\sqrt[h]{n_x})$	√	√
Hao[4]	OPRF，OKVS，FHE-BatchPIR	$O(n_y\sqrt[h]{n_x})$	$O(\lambda n_x)$	√	×
Li[27]	OKS-PRF，OKVS	$O(n_x)$	$O(\kappa n_x)$	×	×
Xue[35]	OPRF，OKVS，FHE	$O(\ell n_y\log_2 n_x)$	$O(\ell\lambda n_x)$	√	×
我们的工作$\mathcal{DLPSI}$	OPRF，OKVS，FHE&CWC-BatchPIR	$O(n_y\sqrt[h]{n_x})$	$O(\lambda n_x)$	√	√

注：(1)在非平衡设定下，服务端和客户端的数据集基数，$n_x\gg n_y$；λ，κ 分别是统计、计算安全参数；h 是汉明权重；ℓ 是标签位长；
(2)AHE：加法同态加密；DCR：高阶剩余类判定性问题；OKS-PRF：不经意密钥共享伪随机函数；
(3)文献[4]中采用 FHE-BatchPIR[40]，$\mathcal{DLPSI}$采用 FHE&CWC-BatchPIR[41]。

进一步，系统性对比$\mathcal{DLPSI}$与基准 LPSI 方案[2-3](对应 CCS'21，USENIX'24)的复杂度，涵盖计算与通信两个关键维度。

(1)计算复杂度。文献[2]采用标准插值多项式方法，其准备阶段达到 $O(n_x^2)$ 的平方复杂度；文献[3]基于 CWC 与 FHE 的混合设计，总运算量为 $(l\cdot T_{PM}+h\cdot T_M)\cdot B\cdot\lceil m/N\rceil$，其中 $l\in O(\sqrt[h]{h!\cdot n_x}+h)$[41]，故线上阶段运算量呈现 $O(n_x\cdot\sqrt[h]{n_x})$ 的亚线性增长。相较之下，$\mathcal{DLPSI}$通过采用随机带状矩阵编码技术，将准备阶段复杂度优化至线性阶 $O(\lambda\cdot n_x)$。线上阶段结合 BatchPIR 算法，运算量为 $(\lceil(3n_x)/N\rceil\cdot T_{PM}+\lceil(3n_x)/N\rceil(h-1)\cdot T_M)$，实现 $O(n_x)$ 的稳定计算性能。故整体计算复杂度 $O(\lambda\cdot n_x)$，保持线性优势。

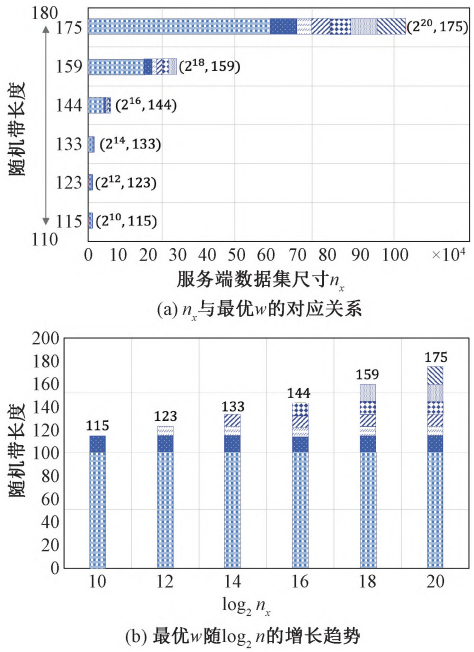
(2)通信复杂度。通信复杂度分析表明：文献[2]执行 DH-OPRF 算法(复杂度 $O(n_y)$)且需传输 FHE 密文，产生 $O(n_y\log_2 n_x)$ 的通信开销。文献[3]的密文传输机制导致 $O(n_y\cdot\sqrt[h]{n_x})$ 的通信负载。具体为客户端发送的 $\lceil m/N\rceil\cdot l$ 个密文，其中每个密文大小为 $N\cdot q$ 比特。

$\mathcal{DLPSI}$的线上阶段，客户端与服务端交互执行 DH-OPRF 算法，保持基础通信量 $O(n_y)$。同时采用压缩 BatchPIR 传输模式，开销为 $(\sqrt[h]{h!\cdot(3n_x)/N}+h)\cdot N\cdot q\cdot n_y\cdot w$。因此，总通信复杂度控制在 $O(n_y\cdot\sqrt[h]{n_x})$。

7.3 关键参数 w 设置

复杂度分析表明，随机带长度 w 是影响$\mathcal{DLPSI}$方案性能的关键参数，其取值直接影响计算与通信开销的平衡。文献[5]提出 $w=O(\lambda/\varepsilon+\log_2 n_x)$ 的理论设置方法，能够将 OKVS 的错误率控制在 $2^{-\lambda}$ 以下，但由于本方案在 OKVS.Encode 算法中需要处理 $3\cdot n_x$ 个数据点(而非传统场景的 n_x 个)，该理论结果无法直接适用。为此，设计系统的实验计算方法：针对不同规模的数据集 n_x，首先生成 $3\cdot n_x$ 个 64 字节的随机整数样本，然后通过二分搜索确定能够通过 6000 次正确性测试的最小 w 值。实验结果显示(如图 4(a))，该参数选择策略能够在

① https://github.com/google/private-membership/tree/main/research/okvs。
② https://github.com/zju-abclab/PIRANA。



(a) n_x与最优w的对应关系

(b) 最优w随$\log_2 n$的增长趋势

图 4　随机带长度 w 的参数优化分析：安全性-性能权衡

错误率与性能之间取得最优平衡，例如当 $n_x=2^{20}$ 时，$w=175$ 可使方案达到最佳运行效率。值得注意的是，w 的取值存在明显的权衡(trade-off)特性：增大 w 能进一步降低错误率(可通过增加测试次数实现)，但会线性增加计算开销。图 4(b)进一步表明，服务端数据规模 n_x 指数增长时，最优 w 值的非线性变化规律，这能为不同规模医疗数据应用场景的参数配置提供参考。

7.4　实验结果与成因分析

表 4 详细对比$\mathcal{DLPSI}$与基准方案[2-3]的性能指标。为确保实验结果的公平性与可重复性，采用以下严格测试方案：首先，将各方案的官方开源实现(对应文献[2]的 Microsoft APSI、文献[3]的 PEPSI 实现①)部署于统一测试环境(Intel Core i7-9750H，16GB 内存，Ubuntu 20.04)；其次，为控制变量，所有方案均采用相同的数据规范(64 字节数据位长，6 字节标签位长)，但由于文献[3]在低线程环境下存在显著性能瓶颈，经权衡后单独将其数据位长调整为 4 字节以完成基准测试；最后，每组实验均独立重复执行 5 次，取平均值作为最终结果。该实验设计能有效确保性能对比的客观性，为方案评估提供可靠的数据支撑。

表 4　$\mathcal{DLPSI}$与基准 LPSI 方案[2-3](对应 CCS'21，USENIX'24)在非平衡数据集下的性能对比

数据集基数		方案	准备阶段计算时间/s			线上阶段计算时间/s			通信开销/MB	
n_x	n_y		$T=1$	$T=4$	$T=8$	$T=1$	$T=4$	$T=8$	$C\to S$	$S\to C$
2^{18}	2^6	APSI[2]	22.56	7.48	6.10	1.09	0.44	0.43	1.03	2.49
		PEPSI[3]	6.28	2.90	2.52	589.63	180.15	121.64	4.57	0.33
		Ours	18.13	6.56	4.80	5.72	4.33	4.16	2.52	0.52
	2^8	APSI[2]	23.34	7.62	6.81	1.13	0.45	0.43	1.03	2.49
		PEPSI[3]	6.28	2.90	2.52	589.94	178.83	121.85	4.57	0.33
		Ours	18.19	6.13	4.78	7.15	4.24	4.21	5.75	2.08
	2^{10}	APSI[2]	23.08	8.32	6.63	1.15	0.59	0.51	1.06	2.51
		PEPSI[3]	6.20	2.91	2.55	590.67	181.27	120.79	4.57	0.33
		Ours	18.24	6.37	4.85	10.80	6.40	6.37	13.90	7.49
2^{20}	2^6	APSI[2]	185.01	58.80	42.28	10.08	3.25	2.43	1.23	3.41
		PEPSI[3]	15.86	5.70	4.25	1786.49	522.10	361.72	4.59	0.33
		Ours	71.77	24.65	19.28	22.71	15.62	15.59	4.75	0.52
	2^8	APSI[2]	186.19	57.75	41.86	10.17	3.23	2.51	1.23	3.32
		PEPSI[3]	15.90	5.57	4.34	1785.81	522.42	362.24	4.59	0.33
		Ours	71.94	24.73	19.04	25.66	15.73	15.66	10.42	2.08
	2^{10}	APSI[2]	185.13	58.11	43.13	10.24	3.27	2.70	1.26	3.34
		PEPSI[3]	15.93	5.70	4.24	1786.80	532.02	360.81	4.59	0.33
		Ours	72.03	24.89	19.14	31.59	19.60	19.19	22.29	8.07

注：(1)n_x 和 n_y 分别表示服务端和客户端的数据集基数，T 表示并行计算线程数配置；

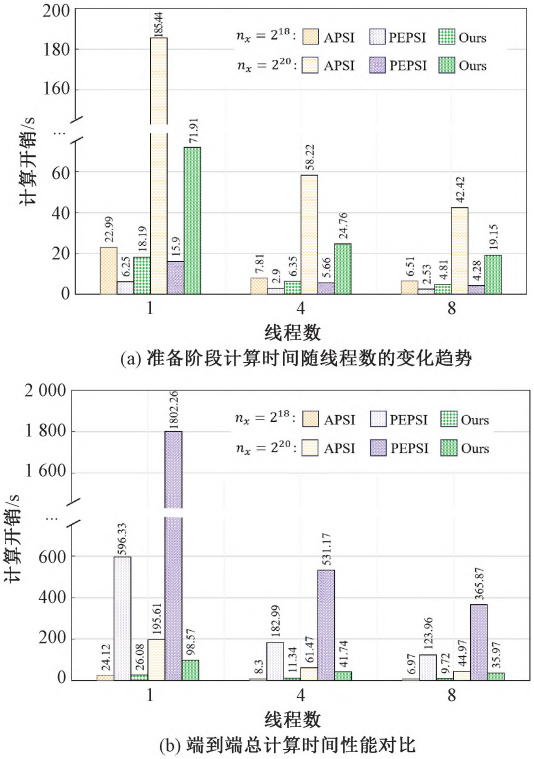
(2)计算时间指标：准备阶段专指服务端离线预处理时间，线上阶段包含服务端与客户端的协同计算总耗时；

(3)通信开销：$C\to S$ 表示客户端上行通信量，$S\to C$ 表示服务端下行通信量。

本节性能分析严格基于第 4 节建立的密码学原语理论框架，其中 OKVS 编码的线性计算复杂度优势(源自第 4 节 RB-OKVS.Encode 中高斯消元求解线性方程组的特性)与结合 BatchPIR 的通信高效 OKVS 解码机制(源自 BatchPIR 中请求/响应消息交换的通信模型)为性能分析提供直接理论依据。

① https://github.com/RasoulAM/pepsi。

(1)计算开销。通过系统性的性能对比实验,评估不同线程数配置下各方案在准备阶段计算时间和总运行时间的表现差异。实验采用固定 n_x 和线程数 T,通过取 $n_y \in \{2^6, 2^8, 2^{10}\}$ 运行时间的平均值来确保结果的代表性。得益于 OKVS 编码的线性方程组求解相较于多项式运算的复杂度优势,如图 5a 所示,$\mathcal{DLPSI}$在准备阶段计算时间上较 APSI 显著降低 12%～61%。在总运行时间方面(详见图 5b),当处理大规模数据集($n_x = 2^{20}$)时,$\mathcal{DLPSI}$的稀疏矩阵乘法相比 APSI 的多项式求值展现出 19%～50%的性能优势;然而在较小规模($n_x = 2^{18}$)下,由于多项式次数降低,$\mathcal{DLPSI}$较 APSI 增加 8%～40%开销。值得注意的是,$\mathcal{DLPSI}$在所有测试场景下均大幅领先 PEPSI 90%～96%。在此背景下,PEPSI 表现出的相对优势在于其线程扩展性:随线程数增加,性能提升更为显著。这主要归因于:(a) PEPSI 高度依赖多线程加速(32 线程下性能最优),而$\mathcal{DLPSI}$和 APSI 的核心运算模块(OPRF 除外)并行化受限;(b) PEPSI 的 CWC 编码运算虽然可高度并行化,但在低线程配置(1/4/8 线程)时性能瓶颈明显,导致其在客户端计算资源有限的场景中效率受限。实验结果表明,$\mathcal{DLPSI}$特别适合服务端数据规模大($n_x \geqslant 2^{20}$)且更新频繁、客户端资源受限的医疗应用场景,能提供最优的计算效率。

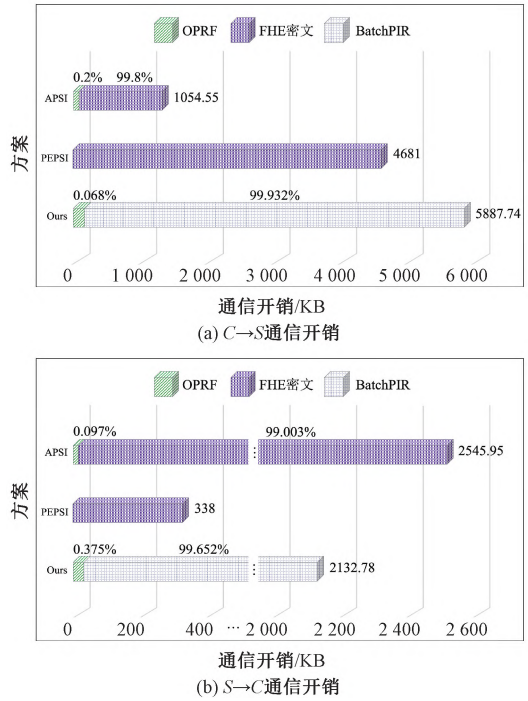


(a) 准备阶段计算时间随线程数的变化趋势

(b) 端到端总计算时间性能对比

图 5 多线程环境下 LPSI 的计算性能对比

(2)通信开销。根据表 4 的数据分析,$\mathcal{DLPSI}$在通信开销方面呈现出与数据集规模相关的增长特性:当(n_x, n_y)=(2^{18}, $2^6/2^8/2^{10}$)时,协议总通信开销较 APSI 与 PEPSI 分别平均增加 2.04 倍和 1.19 倍;而在更大规模 $n_x = 2^{20}$ 时,这一差距进一步扩大至 2.49 倍和 2.26 倍。根据图 6 的数据对比,通过对(n_x, n_y)=(2^{18}, 2^8)的通信开销分析可知,基于 FHE 的 APSI 和 PEPSI 方案的主要通信负载来源于密文传输,而$\mathcal{DLPSI}$则主要消耗于 BatchPIR 协议的请求/响应消息交换。值得注意的是,虽然在小规模数据集下两类通信开销相近,但随着数据规模扩大,BatchPIR 的通信开销呈现超线性增长趋势(相较于 FHE 密文的线性增长),这构成本方案的主要通信瓶颈。然而,这种设计权衡具备显著的计算效率优势:准备阶段大规模数据预处理时间大幅降低,适配数据高频更新场景。在医疗健康领域(如日均需处理百万级患者数据的疾控中心),这种“以通信换计算”的范式具有显著实用价值,其带来的实时性提升远超通信开销的增加,能为大规模医疗标签安全共享提供可行的技术解决方案。



(a) $C \rightarrow S$通信开销

(b) $S \rightarrow C$通信开销

图 6 $n_x = 2^{18}$, $n_y = 2^8$ 设定下 LPSI 通信开销构成与对比

8 总结与展望

本文以医疗健康领域标签共享需求为背景,针对非平衡数据集下的 LPSI 问题,提出融合 RB-OKVS 与 CWC&FHE-BatchPIR 技术的$\mathcal{DLPSI}$。通过重

构 RB-OKVS 编码结构实现数据标签对的原子化传输，同时借助 CWC&FHE-BatchPIR 技术获取必要的 OKVS 编码结果以减少通信开销。安全性分析表明，该方案在半诚实敌手模型下满足隐私保护要求，性能评估验证其计算与通信开销均衡，可有效支撑医疗场景下的标签安全共享。

需注意的是，本方案在医疗场景应用中存在一定局限：医疗数据多呈现范围性特征（如诊断指标区间、疾病分型标准），而传统 PSI 协议依赖数据完全相等实现匹配，直接应用易导致数据匹配失效。

在此前工作[35]（TIFS'25）和本文工作的基础上，后续研究将进一步优化：针对医疗范围性数据，通过哈希映射预处理转化为确定性数据，或扩展为模糊隐私集合交集（Fuzzy PSI）范式以实现相似度阈值匹配；同时优化通信效率并探索基于格密码的后量子安全扩展，结合模块化设计支持 LPSI 与电路 PSI 的灵活转换，以更好适配医疗标签共享场景需求。

参　考　文　献

[1] Chen H, Huang Z C, Laine K, Rindal P. Labeled psi from fully homomorphic encryption with malicious security//Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. Toronto, Canada, 2018: 1223-1237

[2] Cong K, Moreno R C, Gama M B D, Dai W, Iliashenko I, Laine K, Rosenberg M. Labeled psi from homomorphic encryption with reduced computation and communication//Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. Virtual, Republic of Korea, 2021: 1135-1150

[3] Mahdavi R A, Lukas N, Ebrahimianghazani F, Humphries T, Kacsmar B, Premkumar J, Li X, Oya S, Amjadian E, Kerschbaum F. Pepsi: practically efficient private set intersection in the unbalanced setting//Proceedings of the 33rd USENIX Security Symposium. Philadelphia, USA, 2024: 6453-6470

[4] Hao M, Liu W, Peng L, Li H, Zhang C, Chen H, Zhang T. Unbalanced circuit-psi from oblivious key-value retrieval//Proceedings of the 33rd USENIX Security Symposium. Philadelphia, USA, 2024: 6435-6451

[5] Bienstock A, Patel S, Seo J Y, Yeo K. Near-optimal oblivious key-value stores for efficient psi, psu and volume-hiding multimaps//Proceedings of the 32nd USENIX Security Symposium. Anaheim, USA, 2023: 301-318

[6] Meadows C. A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party//Proceedings of the 1986 IEEE Symposium on Security and Privacy. Oakland, USA, 1986: 134-137

[7] Huberman B A, Franklin M, Hogg T. Enhancing privacy and trust in electronic communities//Proceedings of the 1st ACM SIGSAC Conference on Electronic Commerce. Denver, USA, 1999: 78-86

[8] Rosulek M, Trieu N. Compact and malicious private set intersection for small sets//Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. Virtual, Republic of Korea, 2021: 1166-1181

[9] Freedman M J, Nissim K, Pinkas B. Efficient private matching and set intersection//Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques. Interlaken, Switzerland, 2004: 1-19

[10] Ghosh S, Nilges T. An algebraic approach to maliciously secure private set intersection//Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques. Darmstadt, Germany, 2019: 154-185

[11] Abadi A, Murdoch S J, Zacharias T. Polynomial representation is tricky: maliciously secure private set intersection revisited//Proceedings of the 26th European Symposium on Research in Computer Security. Darmstadt, Germany, 2021: 721-742

[12] Pinkas B, Rosulek M, Trieu N, Yanai A. Spot-light: lightweight private set intersection from sparse ot extension//Proceedings of the Annual International Cryptology Conference. Santa Barbara, USA, 2019: 401-431

[13] Abadi A, Terzis S, Metere R, Dong C. Efficient delegated private set intersection on outsourced private datasets. IEEE Transactions on Dependable and Secure Computing, 2017, 16(4): 608-624

[14] Kolesnikov V, Kumaresan R, Rosulek M, Trieu N. Efficient batched oblivious prf with applications to private set intersection//Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. Vienna, Austria, 2016: 818-829

[15] Pinkas B, Schneider T, Zohner M. Scalable private set intersection based on ot extension. ACM Transactions on Privacy and Security, 2018, 21(2): 1-35

[16] Chase M, Miao P. Private set intersection in the internet setting from lightweight oblivious PRF//Proceedings of the Annual International Cryptology Conference. Santa Barbara, USA, 2020: 34-63

[17] Rindal P, Rosulek M. Malicious-secure private set intersection via dual execution//Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. Dallas, USA, 2017: 1229-1242

[18] Garimella G, Pinkas B, Rosulek M, Trieu N, Yanai A. Oblivious key-value stores and amplification for private set intersection//Proceedings of the Annual International Cryptology Conference. Virtual, 2021: 395-425

[19] Cho C, Dachman-Soled D, Jarecki S. Efficient concurrent

covert computation of string equality and set intersection//Proceedings of the Cryptographers' Track at the RSA Conference. San Francisco, USA, 2016: 164-179

[20] Dong C, Chen L, Wen Z. When private set intersection meets big data: an efficient and scalable protocol//Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security. Berlin, Germany, 2013: 789-800

[21] Pinkas B, Rosulek M, Trieu N, Yanai A. Psi from paxos: fast, malicious private set intersection//Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques. Zagreb, Croatia, 2020: 739-767

[22] Pinkas B, Schneider T, Tkachenko O, Yanai A. Efficient circuit-based psi with linear communication//Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques. Darmstadt, Germany, 2019: 122-153

[23] Benny P, Thomas S, Michael Z. Faster private set intersection based on ot extension//Proceedings of the 23rd USENIX Security Symposium. San Diego, USA, 2014: 797-812

[24] Rindal P, Schoppmann P. Vole-psi: fast oprf and circuit-psi from vector-ole//Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques. Zagreb, Croatia, 2021: 901-930

[25] Raghuraman S, Rindal P. Blazing fast psi from improved okvs and subfield vole//Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. Los Angeles, USA, 2022: 2505-2517

[26] Gong Lin-Ming, Wang Dao-Shun, Liu Mo-Meng, Gao Quan-Li, Shao Lian-He, Wang Ming-Ming. PSI computation based on no matching errors. Chinese Journal of Computers, 2020, 43(9): 1769-1790 (in Chinese)
(巩林明, 王道顺, 刘沫萌, 高全力, 邵连合, 王明明. 基于无匹配差错的 PSI 计算. 计算机学报, 2020, 43(9): 1769-1790)

[27] Li Gong-Li, Liu Wei-Chen, Zheng Dong. Research on efficient and scalable private set intersection cardinality schemes. Journal on Communications, 2025, 46(5): 272-282 (in Chinese)
(李功丽, 刘威辰, 郑东. 高效可扩展的隐私集合交集基数方案研究. 通信学报, 2025, 46(5): 272-282)

[28] Chen H, Laine K, Rindal P. Fast private set intersection from homomorphic encryption//Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. Dallas, USA, 2017: 1243-1255

[29] Wu M, Yuen T H. Efficient unbalanced private set intersection cardinality and user-friendly privacy-preserving contact tracing//Proceedings of the 32nd USENIX Security Symposium. Anaheim, USA, 2023: 283-300

[30] Chielle E, Maniatakos M. Recurrent private set intersection for unbalanced databases with cuckoo hashing and leveled fhe//Proceedings of the Annual Network and Distributed System Security Symposium. San Diego, USA, 2025

[31] Kales D, Rechberger C, Schneider T, Senker M, Weinert C. Mobile private contact discovery at scale//Proceedings of the 28th USENIX Security Symposium. Santa Clara, USA, 2019: 1447-1464

[32] Sun Y, Katz J, Raykova M, Schoppmann P, Wang X. Actively secure private set intersection in the client-server setting//Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security. Salt Lake City, USA, 2024: 1478-1492

[33] Bienstock A, Patel S, Seo J Y, Yeo K. Batch pir and labeled psi with oblivious ciphertext compression//Proceedings of the 33rd USENIX Security Symposium. Philadelphia, USA, 2024: 5949-5966

[34] Liu Q, Guo X, Yang K, Yu Y. Labeled private set intersection from distributed point function. IEEE Transactions on Information Forensics and Security, 2025, 20: 2970-2983

[35] Xue J T, Li W Y, Li F G, Zhang W Z, Zhou Y, Zhang X J. Efficient collaborative data cleaning using private set intersection and encoding for unbalanced datasets. IEEE Transactions on Information Forensics and Security, 2025, 20: 8715-8729

[36] Pagh R, Rodler FF. Cuckoo hashing. Journal of Algorithms, 2004, 51(2): 122-144

[37] Arbitman Y, Naor M, Segev G. Backyard cuckoo hashing: constant worst-case operations with a succinct representation//Proceedings of the Annual symposium on foundations of computer science. Las Vegas, USA, 2010: 787-796

[38] Jarecki S, Liu X. Fast secure computation of set intersection//Proceedings of the Annual International Conference on Security and Cryptography for Networks. Amalfi, Italy, 2010: 418-435

[39] Angel S, Chen H, Laine K, Setty S. Pir with compressed queries and amortized query processing//Proceedings of the 39th IEEE Symposium on Security and Privacy. San Francisco, USA, 2018: 962-979

[40] Mughees M H, Ren L. Vectorized batch private information retrieval//Proceedings of the 44th IEEE Symposium on Security and Privacy. San Francisco, USA, 2023: 437-452

[41] Liu J, Li J, Wu D, Ren K. Pirana: faster multi-query pir via constant-weight codes//Proceedings of the 45th IEEE Symposium on Security and Privacy. San Francisco, USA, 2024: 4315-4330

[42] Goldreich O. The foundations of cryptography-volume 2: Basic application. Cambridge, USA: Cambridge University Press, 2004

[43] Zhang C, Chen Y, Liu W, Peng L, Hao M, Wang A, Wang X. Unbalanced private set union with reduced computation and communication//Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security. Salt Lake City, USA, 2024: 1434-1447

XUE Jing-Ting, Ph. D., associate professor. Her primary research interests include applied cryptography, cloud data security auditing, and blockchain technology.

LI Wen-Yi, master candidate. His primary research interests include applied cryptography and secure multi-party computation.

LI Fa-Gen, Ph. D., professor and Ph. D. supervisor. His primary research interests include cryptography and information security.

ZHANG Wen-Zheng, research fellow. His primary research interests include cryptographic techniques, and network and information security.

ZHANG Xiao-Jun, Ph. D., associate professor. His primary research interests include applied cryptography, cloud computing security, and smart grid security and privacy.

Background

This work focuses on Labeled Private Set Intersection (LPSI), a specific problem within secure multi-party computation, a subfield of privacy-preserving computation. Currently, prevailing LPSI schemes globally typically employ either fully homomorphic encryption paradigms or circuit PSI architectures. However, these approaches often encounter computational efficiency bottlenecks during the preprocessing phase or fail to adequately account for secure label generation and transmission, rendering them unsuitable for large-scale dynamic data scenarios with daily updates in the millions. To address these challenges, this paper introduces the first dynamic LPSI scheme enabling high-frequency data updates and real-time label sharing. Its core innovation involves re-engineering the Oblivious Key-Value Store (OKVS) data structure to balance security and performance, thereby achieving atomic secure data-label transmission. By leveraging random band matrix OKVS and batch private information retrieval techniques, our scheme reduces communication complexity from the optimal $O(n_x)$ to $O(n_y \sqrt[h]{n_x})$. C++ experimental results demonstrate a 90%-96% reduction in total runtime compared to PEPSI (USENIX'24) and a 19%-50% reduction compared to APSI (CCS'21) for datasets of size $n_x = 2^{20}$.

This work was supported by the National Natural Science Foundation of China (Grant Nos. 62272090, 61902327), the Natural Science Foundation of Sichuan Province (Grant Nos. 2023NSFSC1398, 2025ZNSFSC0495), and the Science and Technology Fund of the Communication Security Laboratory of China (Grant No. 61421030107012102). Notably, the National Natural Science Foundation of China General Program, "Design and Analysis of Reverse Firewalls for Identity-Based Cryptography" (Grant No. 62272090), aims to construct a new generation of active defense cryptographic systems. This is achieved by innovatively integrating reverse firewall technology into identity-based cryptographic systems, thereby providing inherent security protection for digital infrastructure and fundamentally enhancing network security defense capabilities. Our research group has published several results in this project direction, such as TIFS'25, FGCS'25, IOTJ'25, TCC'24. The LPSI research presented in this paper serves as a crucial privacy-enhancing component within the overarching project, "Design and Analysis of Reverse Firewalls for Identity-Based Cryptography." It specifically addresses the issue of private data label leakage that arises when reverse firewalls are deployed in secure multi-party computation scenarios. By integrating oblivious techniques with robust label protection mechanisms, our work offers a verifiable private intersection computation scheme for the project, filling a critical technical gap in existing reverse firewalls concerning fine-grained data control and further strengthening the security of identity-based cryptographic systems against insider threats.