

面向IEX方案存储开销优化的方法

张象博, 马昌社, 苏雨晨

(华南师范大学计算机学院, 广东 广州 510631)

摘要: 可搜索对称加密允许在加密数据中进行高效搜索, IEX是首个实现次线性搜索复杂度且支持布尔查询的方案, 但该方案的服务器存储开销较大。为减少服务器存储开销, 提出了一种基于三角矩阵的数据组织方法, 该方法可以减少约50%的两两关键字的文档交集数据量。基于上述数据组织方法, 进一步提出了IEX改进方案IEX-RS。实验结果表明, 与IEX方案相比, IEX-RS方案在保持相同搜索性能的同时, 可减少49%的服务器存储开销。

关键词: 可搜索对称加密; 布尔查询; 结构化加密

中图分类号: TP393

文献标志码: A

DOI: 10.11959/j.issn.2096-109x.2025052

Method for reducing the storage overhead of the IEX scheme

ZHANG Xiangbo, MA Changshe, SU Yuchen

School of Computer Science, South China Normal University, Guangzhou 510631, China

Abstract: Searchable symmetric encryption was recognized as an efficient method for searching within encrypted data. The IEX scheme was the first to achieve sublinear search complexity while supporting Boolean queries; however, it was found to incur high server storage overhead. To reduce the storage cost on the server side, a data organization method based on triangular matrices was proposed. This method was shown to reduce the data volume of pairwise keyword-document intersections by approximately 50%. Based on the aforementioned data organization method, an improved scheme named IEX-RS was further proposed. Experimental results demonstrate that the IEX-RS scheme significantly reduces server storage overhead by 49%, while search performance equivalent to that of the original IEX scheme is maintained.

Keywords: searchable symmetric encryption, Boolean search, structured encryption

0 引言

近年来, 云计算技术及应用迅速发展, 用户倾向于将数据外包给云服务器, 以减少本地的资源开销。然而, 云服务器通常被视为不完全可信。为保障数据隐私安全, 用户往往会先对数据进行

加密处理再上传至云端。传统的加密手段虽能有效保证数据安全, 却使用户几乎丧失了对加密数据的搜索能力。若要实现搜索功能, 最简单的解决方式是在查询时将文件全部下载至本地再执行查询操作。但这种方法不仅需要用户在本地进行

收稿日期: 2025-02-01; 修回日期: 2025-06-19

通信作者: 马昌社, chsma@163.com

引用格式: 张象博, 马昌社, 苏雨晨. 面向IEX方案存储开销优化的方法[J]. 网络与信息安全学报, 2025, 11(5): 149-161.

Citation Format: ZHANG X B, MA C S, SU Y C. Method for reducing the storage overhead of the IEX scheme[J]. Chinese Journal of Network and Information Security, 2025, 11(5): 149-161.

大量解密工作,还会大幅增加不必要的通信开销。为解决这一问题,可搜索加密 (searchable encryption, SE) 技术应运而生,其核心目标是在保证数据安全的前提下,以可控的部分信息泄露为代价,换取用户对加密数据的搜索能力。经过多年的发展,当前可搜索加密的研究方向聚焦于在性能、功能与安全性之间寻求更优的平衡。

可搜索对称加密^[1] (searchable symmetric encryption, SSE) 常见的应用场景为单关键字搜索。然而,为满足更广泛的应用需求,SSE 方案的功能需要进一步扩展。为此,后续研究围绕复杂查询需求展开技术迭代,相继提出了支持布尔查询、范围查询、模糊查询等多样化检索模式的 SSE 方案。Chase 等^[2]提出了结构化加密 (structured encryption, STE) 概念。STE 将 SSE 的工作机制推广至任意的结构化数据模型,为布尔查询类 SSE 方案的发展奠定了理论基础。Cash 等^[3]在 SSE 框架下,基于 STE 和 Diffie-Hellman 类操作构建了交叉索引结构,设计出支持联合查询的 OXT (oblivious cross-tags) 方案。该方案实现了次线性时间复杂度的联合查询操作,并可将联合查询拓展为特定形式的布尔查询。然而,OXT 方案在处理非联合查询时效率较低。此外,由于 OXT 依赖 Diffie-Hellman 类操作,因此引入了较大的计算开销。

为解决 OXT 方案在非联合查询场景下的效率瓶颈,Kamara 等^[4]提出非交互式可搜索加密方案 IEX。该方案在维持次线性搜索复杂度的同时,可扩展支持布尔查询。与 OXT 相比,IEX 通过规避 Diffie-Hellman 类操作,显著减少了加密过程的计算开销。

IEX 方案的存储结构包含一个加密多映射和一个加密字典。其中,加密多映射的功能类似于 OXT 中的 TSet,用于存储单个关键字及其对应的加密文档标识符。加密字典则用于存储预计算的关键字对之间的文档标识符交集结果,旨在高效支持析取 (OR) 查询。然而,存储所有可能的关键字对的文档交集会导致服务器的存储开销急剧增长。因此,IEX 方案本质上是以增加存储资源开销为代价,换取析取查询效率的提升。

本文聚焦于解决 IEX 方案中服务器存储开销

过大的问题。分析发现,该问题的核心症结在于其加密字典在存储关键字对的文档标识符交集时存储了大量冗余数据。以关键字集合 $\{w_1, w_2, w_3\}$ 为例,加密字典不仅存储 $w_1 \wedge w_2$ 的文档交集,还会重复存储 $w_2 \wedge w_1$ 的文档交集。为减少 IEX 方案的服务器存储开销,本文提出基于三角矩阵的数据组织方法,在此基础上设计了优化服务器存储开销的 SSE 方案 IEX-RS。与 IEX 相比,该方案在保持相同搜索性能的同时,减少了约 50% 的服务器存储开销。

1 相关工作

Song 等^[1]首次提出了可搜索加密的概念,基于流密码构造了可搜索对称加密方案,允许在加密数据上进行高效的关键字搜索。Curtmola 等^[5]首次给出了可搜索对称加密的安全定义,并基于倒排索引设计了一个高效的 SSE 方案。然而,早期的静态 SSE 方案均不支持加密数据的动态更新。为实现加密数据的动态更新,动态可搜索对称加密 (dynamic searchable symmetric encryption, DSSE) 技术应运而生。Cash 等^[6]设计了一个可扩展的 DSSE 方案,能够在 10^9 数量级键值对的大型数据库中执行高效查询。但该方案在数据更新过程中会泄露部分关键字信息。为解决这一安全隐患,Bost^[7]首次提出前向安全的形式化定义,并基于陷门置换理论构造了满足前向安全的 DSSE 方案。

Bost 等^[8]在前向安全理论上进一步拓展安全模型,提出包含 3 种不同强度的后向安全形式化定义。该团队基于增量穿刺加密技术设计了 Janus 方案,首次实现同时满足前后向安全的 DSSE 系统。

早期的布尔查询方案大多数仅支持联合查询。Golle 等^[9]提出了满足组件间通信 (inter-component communication, ICC) 安全模型的联合查询方案,其通信开销和服务器存储开销均与文件数量呈线性关系。Ballard 等^[10]基于双线性对设计了一个联合查询方案,但该方案的服务器计算开销较大。Cash 等^[3]提出 OXT 方案,首次实现次线性搜索复杂度的联合查询,并支持将联合查询拓展为布尔查询形式。然而,该方案在执行

析取查询时效率较低。文献[11]和文献[12]基于混淆电路^[13]和布隆过滤器技术,提出了支持布尔查询、范围查询等查询功能的方案。Kamara等^[4]提出非交互式SSE方案IEX,该方案实现了次线性复杂度的析取查询,并可扩展支持布尔查询。当前主流的布尔查询SSE方案呈现多元技术路径,多数方案基于OXT与IEX架构进行改进优化,少数方案基于树结构^[14-15]与扩展位图索引^[16]实现。

在布尔查询SSE方案研究领域,基于OXT方案的改进探索持续深入。文献[17-18]揭示攻击者能够利用SSE方案的部分泄露信息恢复查询关键字。文献[19]指出文件注入攻击对SSE方案构成严重威胁,攻击者利用关键字对结果模式(keyword pair result pattern, KPRP)泄露可准确恢复查询关键字。为解决联合查询中的KPRP泄露问题,Lai等^[20]提出基于隐藏向量加密和布隆过滤器技术的联合查询方案HXT(hidden cross-tags),成功实现KPRP隐藏。不过,该方案额外增加一轮客户端与服务器通信,致使服务器存储开销大幅攀升。

随着文献[7]明确前向安全正式的定义,文献[8]提出后向安全正式的定义,布尔查询方案开始向满足前后向安全方向发展。Patranabis等^[21]在OXT基础上构建DSSE方案ODXT(oblivious dynamic cross-tags),不仅满足前后向安全,还能扩展至大型结构化数据库。然而,该方案存在惰性删除缺陷(执行删除操作时仅在关系表中添加新条目,并未实际删除数据)。Ma等^[22]提出子集成员查询密码学原语,并基于此设计了PHXT(practical hidden cross-tags)方案^[22]。PHXT解决了KPRP泄露问题,与HXT方案相比,其存储开销和通信开销均有减少。Zuo等^[23]通过调用非交互式删除的前后向安全方案Aura^[24],提出SSE方案解决了ODXT的惰性删除问题,同时保证了TSet和XSet的前向安全性。与ODXT相比,该方案通信开销和计算开销有所增加。Jutla等^[25]提出JXT(join cross-tags)方案,该方案支持对连接后的加密表进行联合查询,与OXT实现完全兼容。Yuan等^[26]基于HXT提出满足前后向安全的动态联合查询方案HDXT(hidden dynamic

cross-tags)。HDXT保留了隐藏KPRP的特性,搜索复杂度为次线性,但是增加了客户端存储开销。Bag等^[27]提出的TWINSSE(two-in-one-SSE)方案,支持联合与析取查询,并可拓展至一般布尔查询。Bag等^[28]提出满足前后向安全的联合查询方案,将应用场景拓展到多用户多读单写环境。Yuan等^[29]设计了Result-hiding Filter,并基于此提出了支持布尔查询的方案HBS(hidden Boolean search)。该方案能够隐藏KPRP,具有存储开销较少、假阳性概率可忽略不计的优点。

在基于IEX的布尔查询SSE方案研究领域,一系列创新性成果不断涌现。Patel等^[30]提出支持布尔查询的STE方案CNFFilter,该方案有效规避了OXT方案中单关键字结果集泄露的风险,安全性较IEX更高。郭斯栩等^[31]提出一种基于多关键字top- k 布尔可搜索加密方案TBSE(top- k Boolean searchable encryption scheme based on multiple keywords)。该方案可在执行布尔查询的同时实现文件的top- k 排序,执行布尔查询的效率高于IEX方案,并支持索引的动态更新。Ferreira等^[32]将布尔查询应用场景拓展至多用户领域,提出支持多用户执行任意布尔查询的DSSE方案BISEN(Boolean isolated searchable symmetric encryption)。杨紫依等^[33]引入转换密钥加密的密码学原语,基于ConjFilter提出可解决交叉模式泄露问题的联合查询方案TK-Filter(transformable key filter)。

除基于OXT、IEX的改进方案外,基于树结构和扩展位图构造索引的方案也为布尔查询研究开辟了新方向。文献[14]提出的IBTree(indistinguishable binary tree)与文献[15]提出的VBTree(virtual binary tree),均基于树结构构建索引。二者虽然性能优良,但存在信息泄露问题。Zuo等^[16]引入扩展位图索引概念,并在此基础上设计了一个满足前后向安全的DSSE方案。该方案不仅支持联合查询,且客户端存储开销为常数级别。

上述方案均须在功能性、性能与安全性之间进行权衡。以IEX方案为例,其通过增加服务器存储开销,成功实现析取查询效率的显著提升,

达到次线性搜索复杂度,但也因此带来服务器存储开销过大的弊端。类似地, ConjFilter 方案^[30]在基于 IEX 的改进中,同样选择以增加服务器存储开销为代价,实现联合查询效率的优化。二者是以空间换时间的典型案例。针对 IEX 方案存储开销过大问题,本文提出析取查询方案 IEX-RS。该方案在维持与 IEX 方案相同搜索效率的前提下,大幅降低了服务器存储开销。

2 预备知识

2.1 符号说明

本节将对本文使用的符号进行说明。 $x \stackrel{s}{\leftarrow} X$ 表示从集合 X 中均匀随机采样得到样本 x 。 $\{0,1\}^n$ 表示所有长度为 n 的二进制字符串。假设 a 和 b 为正整数,并且 $a < b$, $[a]$ 表示正整数集合 $\{1, \dots, a\}$, $[a, b]$ 表示正整数集合 $\{a, \dots, b\}$ 。 $R \leftarrow A$ 表示运行算法 A , 输出结果 R 。给定长度为 n 的向量或序列 $\mathbf{v}$, 用 $\mathbf{v}[i]$ 或者 v_i 表示其中的第 i 个分量, 用 $|\mathbf{v}|$ 表示向量或序列的长度。 $\text{poly}(\cdot)$ 表示一个能在多项式时间内执行的函数。对于函数 $\text{negl}(\lambda): \mathbb{N} \rightarrow \mathbb{R}$, 若对于任意的正多项式 $\text{poly}(\cdot)$, 都存在一个正整数 N , 对于所有的 $\lambda > n$, 都有 $\text{negl}(\lambda) < 1/\text{poly}(\cdot)$, 则称 $\text{negl}(\lambda)$ 为可忽略函数。

对于集合 A 和 B , $A \setminus B$ 表示做集合的差集运算。字典 DX 由 n 个标签/值对 $\{(l_i, \mathbf{v}_i)\}_{i \leq n}$ 组成, $\text{DX}[l_i]$ 表示获取标签 l_i 对应的值, 返回值 $\mathbf{v}_i$ 。多映射 MM 由 n 个标签/元组对 $\{(l_i, \mathbf{v}_i)\}_{i \leq n}$ 组成, $\text{MM}[l_i]$ 表示通过标签 l_i 获取对应值元组。假设明文数据库为 DB , 对应数据集索引用倒排索引 $(w_i, \text{DB}(w_i))_{i=1}^{|W|}$ 表示。 id_i 表示第 i 个文档的标识符, W 表示数据库中的所有关键字集合, $|W|$ 表示数据库中关键字数量, $\text{DB}(w_i)$ 表示包含关键字 w_i 的文档标识符集合, $\text{DB}(w_i) \cap \text{DB}(w_j)$ 和 $\text{DB}(w_i \wedge w_j)$ 表示关键字 w_i 与关键字 w_j 的文档交集。为简化后续描述, 本文将字典 DX 和多映射 MM 中的标签 l 统一用关键字 w 表示, 值 $\mathbf{v}$ 用文档标识符 id 表示, 即以关键字作为键。

2.2 IEX 方案

IEX 方案是首个实现次线性搜索复杂度且支

持布尔查询的可搜索加密方案。它在 STE 的基础上, 利用多映射和字典两种数据结构构建索引。在执行析取查询时, IEX 将求集合并集的操作转换为求集合交集的操作。在安全体系构建方面, IEX 方案基于给定的安全参数 λ , 以及字典加密方案 $\Sigma_{\text{DX}} = (\text{Setup}, \text{Token}, \text{Get})$ 、加密多映射方案 $\Sigma_{\text{MM}} = (\text{Setup}, \text{Token}, \text{Get})$ 、SKE 方案 (Enc, Dec) , 构建起完整的安全防护体系。IEX 主要由系统初始化算法 Setup 、令牌生成算法 Token 和搜索协议 Search 三大模块组成。IEX 方案的实现流程如下。

1) 执行系统初始化算法: $(K, \text{EDB}) \leftarrow \text{IEX.Setup}(\lambda, \text{DB})$ 。此算法主要在客户端执行。客户端输入明文数据库 DB 和安全参数 λ , 服务器无输入。该算法构建的加密数据库 EDB 由加密多映射 EMM_g 和加密字典 EDX 组成, 系统初始化算法输出加密数据库 EDB 和密钥集合 K , 其中 K 保存在客户端、 EDB 存储于服务器。

2) 执行令牌生成算法: $\text{tk} \leftarrow \text{IEX.Token}(K, \Phi)$ 。此算法主要在客户端执行。客户端输入查询 $\Phi = (w_1 \vee \dots \vee w_q)$ 和密钥集合 K , 输出令牌集合 $\text{tk} = \{\text{tk}_1, \dots, \text{tk}_{q-1}, \text{gtk}_q\}$ 。其中, 每个子令牌 $\text{tk}_i = (\text{gtk}_i, \text{dtk}_i, \text{ltk}_{i+1}, \dots, \text{ltk}_q)$ 分为两部分, gtk_i 用于查询单关键字 w_i , 剩余令牌 $(\text{dtk}_i, \text{ltk}_{i+1}, \dots, \text{ltk}_q)$ 用于查询关键字 w_i 与 w_j 的文档交集, 其中 $j \in [i+1, q]$ 。对于 Φ 中最后一个关键字 w_q , 仅生成用于单关键字查询的令牌 gtk_q , 因为该关键字无须进行交集查询。

3) 执行搜索协议: $R \leftarrow \text{IEX.Search}(\text{EDB}, \text{tk})$ 。该协议在客户端和服务器交互执行。客户端输入查询令牌集合 tk , 服务器输入加密数据库 EDB 。对于令牌集合 tk 中的子令牌 $\text{tk}_i = (\text{gtk}_i, \text{dtk}_i, \text{ltk}_{i+1}, \dots, \text{ltk}_q)$, 服务器先利用 gtk_i 检索加密多映射 EMM_g , 得到关键字 w_i 对应的加密文档集合, 即 $\text{DB}(w_i)$ 。随后, 利用其余令牌检索加密字典 EDX , 得到文档交集 $\text{DB}(w_i) \cap \text{DB}(w_j)$, 其中 $j \in [i+1, q]$ 。之后对 $\text{DB}(w_i)$ 依次执行过滤操作, 即进行差集运算 $\text{DB}(w_i) \setminus (\text{DB}(w_i) \cap \text{DB}(w_j))$, 直到所有文档交集都过滤完毕, 得到过滤后的结果集合 R_i 。最后, 对每个子令牌 tk_i 得到的结果

集合 R_i 执行并集运算，将得到的查询结果 R 返回给客户端。

3 基于三角矩阵的数据组织方法

本节首先分析 IEX 方案的系统初始化算法。IEX 初始化实例如图 1 所示。图 1 中最左侧为倒排索引表，表示每个关键字 w 及其对应的文档标识符 id ，调用 IEX.Setup 算法后得到全局多映射 MM_g 和字典 DX 。DX 需要使用字典加密方案 $\Sigma_{DX}.Setup$ 进行加密，生成加密字典 EDX ， MM_g 需要使用加密多映射方案 $\Sigma_{MM}.Setup$ 进行加密，得到加密全局多映射 EMM_g 。图 1 中为简化展示，省略了这两个加密步骤。在 MM_g 和 EMM_w 中，文档标识符 id 实际对应加密文档标识符 tag ，但为直观起见，图中统一以明文形式 id 表示。全局多映射 MM_g 将关键字 w 映射到其对应 id 元组，而 DX 将关键字 w_i 映射到对应的加密多映射 EMM_{w_i} 。在加密多映射 EMM_{w_i} 中， $EMM_{w_i}[w_j]$ 用于存储 $DB(w_i) \cap DB(w_j)$ 的加密文档标识符集合。为便于展示， EMM_w 在图 1 中以明文表示。由于其存储了两个关键字的文档交集，因此其键用 $DB(w_i \wedge w_j)$ 表示。

从图 1 中可以观察到，DX 存在重复存储两个关键字文档交集的情况。例如，在多映射 EMM_{w_1} 中已经存储 $w_1 \wedge w_2$ 的交集 id ，而在多映射 EMM_{w_2} 中又额外存储了 $w_2 \wedge w_1$ 的交集 id ，造成了冗余。进一步分析可知，冗余数据主要来源于两个方面：一是 DX 中 EMM_w 存储了两两不同

关键字文档交集；二是 MM_g 存储内容可以看作两个相同关键字的文档交集。

针对 IEX 方案的冗余数据问题，本文提出基于三角矩阵的数据组织优化方法。具体而言，对于一个包含 n 个关键字的集合 W ，可将 IEX 构造的 DX 抽象成一个 $n \times n$ 的矩阵 M 。表 1 对优化前后的字典存储情况进行了对比，其中 IEX-RS 采用了三角矩阵数据组织方法。矩阵中的第一列与第一行代表关键字，矩阵 M 中每个元素 $M[i, j]$ 存储关键字 w_i 和关键字 w_j 的文档交集。表 1 中， $\perp$ 表示不存储当前两个关键字的文档交集。由于 IEX-RS 仅存储不同关键字的文档交集，因此对角线元素均为 $\perp$ 。从表 1 可以清晰地观察到，IEX 方案中存在显著的数据冗余现象：在下三角矩阵中，除对角线外的每个元素 $M[i, j]$ ，在上三角矩阵中都能找到对应的重复数据 $M[j, i]$ 。

虽然基于三角矩阵的数据组织方法有效削减了冗余数据，但其在查询适配方面存在明显局限。由于客户端无法预知 DX 中关键字的顺序，因此难以直接生成正确的查询令牌满足查询需求。举例来说，假设查询 Φ 中包含关键字 w_1 和 w_2 ，在查询表达式中 w_1 位于 w_2 之前，但在三角矩阵存储结构中 w_2 排在 w_1 之前，那么在矩阵的多映射 EMM_{w_1} 中仅存储 $w_2 \wedge w_1$ 的文档交集，而 $w_1 \wedge w_2$ 的文档交集将被忽略。此时若客户端基于 w_1 生成查询令牌，服务器在查询 EMM_{w_1} 后将返回空结果，因为文档交集实际存储在 EMM_{w_2} 中，只有基于关键字 w_2 生成的查询令牌，才能

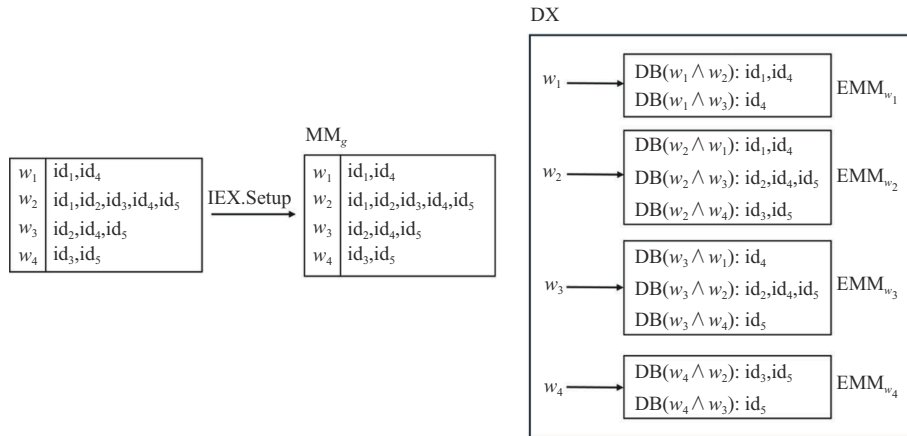


图1 IEX多映射构造图

Figure 1 IEX multimap construction diagram

表1 DX优化示意
Table 1 DX optimization diagram

关键字	IEX					
	w_1	w_2	w_3	w_4	$\dots$	w_n
w_1	$\perp$	$DB(w_1 \wedge w_2)$	$DB(w_1 \wedge w_3)$	$DB(w_1 \wedge w_4)$	$\dots$	$DB(w_1 \wedge w_n)$
w_2	$DB(w_2 \wedge w_1)$	$\perp$	$DB(w_2 \wedge w_3)$	$DB(w_2 \wedge w_4)$	$\dots$	$DB(w_2 \wedge w_n)$
w_3	$DB(w_3 \wedge w_1)$	$DB(w_3 \wedge w_2)$	$\perp$	$DB(w_3 \wedge w_4)$	$\dots$	$DB(w_3 \wedge w_n)$
w_4	$DB(w_4 \wedge w_1)$	$DB(w_4 \wedge w_2)$	$DB(w_4 \wedge w_3)$	$\perp$	$\dots$	$DB(w_4 \wedge w_n)$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
w_n	$DB(w_n \wedge w_1)$	$DB(w_n \wedge w_2)$	$DB(w_n \wedge w_3)$	$DB(w_n \wedge w_4)$	$\dots$	$\perp$

关键字	IEX-RS					
	w_1	w_2	w_3	w_4	$\dots$	w_n
w_1	$\perp$	$DB(w_1 \wedge w_2)$	$DB(w_1 \wedge w_3)$	$DB(w_1 \wedge w_4)$	$\dots$	$DB(w_1 \wedge w_n)$
w_2	$\perp$	$\perp$	$DB(w_2 \wedge w_3)$	$DB(w_2 \wedge w_4)$	$\dots$	$DB(w_2 \wedge w_n)$
w_3	$\perp$	$\perp$	$\perp$	$DB(w_3 \wedge w_4)$	$\dots$	$DB(w_3 \wedge w_n)$
w_4	$\perp$	$\perp$	$\perp$	$\perp$	$\dots$	$DB(w_4 \wedge w_n)$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
w_n	$\perp$	$\perp$	$\perp$	$\perp$	$\dots$	$\perp$

成功查询到 $w_2 \wedge w_1$ 的文档交集。为解决这一问题,一种可行的暴力方法是让客户端枚举所有可能的关键字对组合,分别生成对应的多映射 EMM_w 查询令牌,随后服务器逐一尝试使用这些令牌查询 EMM_w 。若某次查询结果为空,则丢弃该令牌,继续使用下一个令牌进行查询,直至遍历所有令牌。但该策略会使服务器执行大量无效查询操作,导致查询效率大幅降低。

此时的问题在于即便使用上三角矩阵存储使关键字处于有序状态,客户端若没有掌握排序规则,仍无法生成有效的查询令牌。一种简单的解决方法是客户端本地直接缓存经过排序处理的关键字序列。具体来说,在系统初始化阶段,客户端执行特定算法:首先对数据库的关键字进行排序,得到有序的关键字序列 $\mathcal{W} = \{w_1, w_2, \dots, w_q\}$;随后,依据序列 $\mathcal{W}$ 的关键字顺序生成 EMM_{w_i} ,该集合记录同时包含关键字 w_i 和 w_j 的加密文档标识符 tag 集合(其中 $i < j$),其余构造过程与 IEX 方案保持一致。初始化完成后,客户端将该关键字序列 $\mathcal{W}$ 缓存在本地。在生成查询令牌时,客户端先依据本地缓存的关键字序列 $\mathcal{W}$ 对输入的待查询关键字进行排序,再基于重排后顺序生成对应的查询令牌。若将此时服务器存储的 DX 用矩阵表示,

其对应表 1 中优化后的结构。此方法通过客户端主动维护关键字排序信息,既有效消除了矩阵左下角的冗余数据,又确保客户端能够生成与服务器存储结构匹配的正确查询令牌。然而,该方案的实施也带来了新的挑战,即客户端须额外分配存储空间用于缓存关键字序列,从而增加了客户端的存储开销。

在实际应用中,客户端的存储空间通常比较有限,即便仅缓存关键字序列,当关键字数量庞大时,也会产生显著的存储开销。针对客户端存储受限情况,本文提出一种更优的改进方案:客户端不直接保存排序后的关键字序列,而是让客户端掌握关键字排序方法。经实验验证,服务器采用三角矩阵组织数据产生的存储开销与关键字排序方式并无关联,不同的排序方法仅改变矩阵中文档交集的存储位置,不影响整体数据量。考虑到客户端在生成查询令牌时,仅持有密钥集合和待查询关键字,因此排序方法须与关键字自身属性紧密相关。为此,本文选定字典序排序作为统一策略。在系统初始化阶段,客户端按照字典序规则对关键字集合进行排序,并依此构建字典 DX;在令牌生成环节,客户端接收查询关键字后,同样先执行字典序排序操作,再生成对应的

查询令牌，从而保障令牌的有效性。本文将该改进方案命名为 IEX-RS。从存储结构来看，IEX-RS 方案仅保留上三角矩阵数据，相比 IEX 原始方案，成功消除了下三角部分的冗余内容。理论上，该方法可使字典 DX 的存储开销减少 50%。

4 IEX-RS 方案

IEX-RS 方案由系统初始化算法 Setup、令牌生成算法 Token 和搜索协议 Search 组成。给定字典加密方案 $\Sigma_{\text{DX}} = (\text{Setup}, \text{Token}, \text{Get})$ 、加密多映射方案 $\Sigma_{\text{MM}} = (\text{Setup}, \text{Token}, \text{Get})$ 、SKE 方案 (Enc, Dec) 和伪随机函数 F ，IEX-RS 方案具体描述如下。

4.1 方案描述

(1) 系统初始化算法 Setup 执行流程

系统初始化算法 Setup 由客户端执行，具体流程如算法 1 所示。客户端输入安全参数 λ 和明文数据库 DB，服务器无输入；输出密钥集合 K 和加密数据库 EDB。首先，从密钥空间 $\{0,1\}^\lambda$ 随机均匀抽样生成密钥 K_1 ，并初始化字典 DX 和全局多映射 MM_g 。在加密数据库前，客户端对关键字集合 $\mathcal{W}$ 按字典序进行排序，得到有序关键字序列。之后，对于每个关键字 $w_i \in \mathcal{W}$ ，客户端对 w_i 对应的所有文档标识符 id 进行加密，生成对应的密文 $\text{tag}_{w_i, \text{id}}$ ，并将其存储到全局多映射 $\text{MM}_g[w_i]$ 中。然后，针对每个关键字 $w_i \in \mathcal{W}$ ，客户端初始化多映射 MM_{w_i} ，对关键字 w_i 和关键字 w_j （在关键字序列 $\mathcal{W}$ 中， w_j 位于 w_i 后）所有的交集文档 id，使用满足 IND-CPA 安全的对称加密算法进行加密，并将生成的密文 $\text{tag}_{w_i, \text{id}}$ 存储到对应的多映射 $\text{MM}_{w_i}[w_j]$ 中。随后，客户端调用多映射初始化算法 $\Sigma_{\text{MM}}.\text{Setup}$ 对多映射 MM_{w_i} 进行加密。最后，客户端调用多映射初始化算法 $\Sigma_{\text{MM}}.\text{Setup}$ 加密全局多映射 MM_g 得到 EMM_g ，使用 $\Sigma_{\text{DX}}.\text{Setup}$ 对字典 DX 进行加密，得到加密字典 EDX。初始化算法运行结束后，客户端存储密钥集合 K ，服务器存储加密数据库 EDB。

算法 1 Setup 算法

输入 λ, DB

输出 K, EDB

1 $K_1 \xleftarrow{\$} \{0,1\}^\lambda$

```

2  初始化字典 DX 和全局多映射  $\text{MM}_g$ 
3       $\mathcal{W} \leftarrow \text{sort}(\mathcal{W})$ 
4  For  $i = 1$  to  $|\mathcal{W}|$  do
5      For  $\text{id} \in \text{DB}(w_i)$  do
6           $\text{tag}_{w_i, \text{id}} \leftarrow \text{Enc}(F(K_1, w_i), \text{id})$ 
7           $\text{MM}_g[w_i] \leftarrow \text{tag}_{w_i, \text{id}}$ 
8      end for
9      初始化多映射  $\text{MM}_{w_i}$ 
10     For  $j = i + 1$  to  $|\mathcal{W}|$  do
11         For  $\text{id} \in \text{DB}(w_i) \cap \text{DB}(w_j)$  do
12              $\text{tag}_{w_i, \text{id}} \leftarrow \text{Enc}(F(K_1, w_i),$ 
13                  $\text{id})$ 
14              $\text{MM}_{w_i}[w_j] \leftarrow \text{tag}_{w_i, \text{id}}$ 
15         end for
16     end for
17      $(K_{w_i}, \text{EMM}_{w_i}) \leftarrow \Sigma_{\text{MM}}.\text{Setup}(1^\lambda,$ 
18          $\text{MM}_{w_i})$ 
19      $\text{DX}[w_i] \leftarrow \text{EMM}_{w_i}$ 
20 end for
21  $(K_g, \text{EMM}_g) \leftarrow \Sigma_{\text{MM}}.\text{Setup}(1^\lambda, \text{MM}_g)$ 
22  $(K_d, \text{EDX}) \leftarrow \Sigma_{\text{DX}}.\text{Setup}(1^\lambda, \text{DX})$ 
23  $K \leftarrow \{K_1, K_g, K_d, \{K_w\}_{w \in \mathcal{W}}\}$ 
24  $\text{EDB} \leftarrow \{\text{EDX}, \text{EMM}_g\}$ 

```

(2) 令牌生成算法 Token 执行流程

令牌生成算法 Token 由客户端执行，具体流程如算法 2 所示。客户端输入密钥集合 K 和查询范式 $\Phi = (w_1 \vee \dots \vee w_q)$ ，服务器无输入；输出令牌集合 tok_Φ 。首先，客户端解析查询语句，得到关键字序列 $\mathcal{W}$ ，并对 Φ 中所有查询关键字进行字典排序，得到有序序列 $\mathcal{W}$ 。然后，客户端根据 $\mathcal{W}$ 中的每个关键字 w_i ，调用 $\Sigma_{\text{MM}}.\text{Token}$ 生成令牌 gtk_i ，用于检索加密全局多映射 EMM_g ，得到 w_i 的加密文档集合；调用 $\Sigma_{\text{DX}}.\text{Token}$ 生成令牌 dtk_i ，用于检索加密字典 EDX 中的相关条目；针对位于 w_i 之后的剩余关键字 w_j ，调用 $\Sigma_{\text{MM}}.\text{Token}$ 生成令牌 ltk_j ，用于查询加密多映射 EMM_{w_i} 中 w_i 和 w_j 的文档交集。对于 $\mathcal{W}$ 中的每个关键字 w_i ($i \in [1, q-1]$)，都会生

成子令牌集合 $\text{tok}_i = (\text{gtk}_i, \text{dtk}_i, \text{ltk}_{i+1}, \dots, \text{ltk}_q)$ 。对于序列 $\mathcal{W}$ 中最后的关键词 w_q ，仅生成用于查询全局多映射 EMM_g 的令牌 gtk_q 。最后，客户端将生成的所有令牌集合组装成总令牌集合 $\text{tok}_\phi = (\text{tok}_1, \dots, \text{tok}_{q-1}, \text{gtk}_q)$ ，并发送给服务器。

算法2 Token算法

输入 $K, \Phi = (w_1 \vee \dots \vee w_q)$

输出 tok_ϕ

```

1       $\{K_1, K_g, K_d, \{K_w\}_{w \in W}\} \leftarrow K$ 
2       $W \leftarrow \{w_1, \dots, w_q\}$ 
3       $\mathcal{W} \leftarrow \text{sort}(W)$ 
4      for  $i = 1$  to  $|\mathcal{W}| - 1$  do
5           $\text{gtk}_i \leftarrow \Sigma_{\text{MM}}.\text{Token}(K_g, w_i)$ 
6           $\text{dtk}_i \leftarrow \Sigma_{\text{DX}}.\text{Token}(K_d, w_i)$ 
7          for  $j = i + 1$  to  $|\mathcal{W}|$  do
8               $\text{ltk}_j \leftarrow \Sigma_{\text{MM}}.\text{Token}(K_w, w_j)$ 
9          end for
10          $\text{tok}_i \leftarrow (\text{gtk}_i, \text{dtk}_i, \text{ltk}_{i+1}, \dots, \text{ltk}_q)$ 
11     end for
12      $\text{gtk}_q \leftarrow \Sigma_{\text{MM}}.\text{Token}(K_g, w_q)$ 
13      $\text{tok}_\phi \leftarrow (\text{tok}_1, \dots, \text{tok}_{q-1}, \text{gtk}_q)$ 

```

(3) 搜索协议 Search 执行流程

搜索协议 Search 在客户端和服务器交互执行，具体流程如算法3所示。服务器输入加密数据库 EDB，客户端输入令牌集合 tok_ϕ ；输出查询 Φ 的加密结果集 R ，其中 R 为加密文档标识符 tag 的集合。首先，对于每个子令牌 tok_i ，服务器使用其中的 gtk_i 作为输入，调用 $\Sigma_{\text{MM}}.\text{Get}$ 算法检索加密全局多映射 EMM_g ，获取与关键词 w_i 对应的密文集合 R_i ，对应 $\text{DB}(w_i)$ 。然后，服务器使用 dtk_i 检索加密字典 EDX，获取 w_i 对应的加密多映射 EMM_{w_i} ，结合子令牌中的 ltk_j 检索 EMM_{w_i} ，得到 w_i 与其他关键词 w_j 进行交集运算后的结果集 $R' [R' = \text{DB}(w_i) \cap \text{DB}(w_j)]$ 。之后，将结果集 R' 中的元素从 R 中剔除，即执行集合差集操作。上述过程对所有子令牌 tok_i 重复执行，得到每个关键词 w_i 处理后的结果集 R_i 。对于查询的最后一个

关键词 w_q ，服务器使用子令牌 gtk_q 检索多映射 EMM_g ，得到集合 R_q 。最终，服务器对所有结果集 R_i 求并集，得到完整的加密结果集 R （形式化描述为 $R = \bigcup_{i=1}^q R_i$ ），并将 R 返回给客户端。

算法3 Search 算法

输入 $\text{tok}_\phi = (\text{tok}_1, \dots, \text{tok}_{q-1}, \text{gtk}_q), \text{EDB}$

输出 R

```

1       $R \leftarrow \emptyset$ 
2      For  $i = 1$  to  $q - 1$  do
3           $\text{tok}_i \leftarrow (\text{gtk}_i, \text{dtk}_i, \text{ltk}_{i+1}, \dots, \text{ltk}_q)$ 
4           $R_i \leftarrow \Sigma_{\text{MM}}.\text{Get}(\text{EMM}_g, \text{gtk}_i)$ 
5           $\text{EMM}_{w_i} \leftarrow \Sigma_{\text{DX}}.\text{Get}(\text{EDX}, \text{dtk}_i)$ 
6          for  $j = i + 1$  to  $q$  do
7               $R' \leftarrow \Sigma_{\text{MM}}.\text{Get}(\text{EMM}_{w_i}, \text{ltk}_j)$ 
8               $R_i \leftarrow R_i \setminus R'$ 
9          end for
10     end for
11      $R_q \leftarrow \Sigma_{\text{MM}}.\text{Get}(\text{EMM}_g, \text{gtk}_q)$ 
12      $R \leftarrow R_1 \cup R_2 \cup \dots \cup R_q$ 

```

4.2 安全性分析

在 IEX-RS 方案中，泄露函数的形式化定义为 $\mathcal{L}_{\text{IEX-RS}} = (\mathcal{L}_{\text{Setup}}, \mathcal{L}_{\text{Search}})$ 。

(1) 设置泄露函数 $\mathcal{L}_{\text{Setup}}$

该函数主要用于生成加密数据库的泄露信息，其形式化定义为 $\mathcal{L}_{\text{Setup}} = (\mathcal{L}_{\text{Setup}}^{\text{mm}}(\text{MM}_g), \mathcal{L}_{\text{Setup}}^{\text{dx}}(\text{DX}))$ 。式中， $\mathcal{L}_{\text{Setup}}^{\text{mm}}(\text{MM}_g)$ 表示全局多映射的设置泄露， $\mathcal{L}_{\text{Setup}}^{\text{dx}}(\text{DX})$ 表示字典的设置泄露。通过初始化算法生成的加密数据库泄露信息具体涵盖全局多映射 MM_g 包含的键值对数量、字典 DX 包含的键值对数量和多映射 MM_w 包含的键值对数量。在 IEX-RS 的初始化算法中，新增了对排序操作的优化。该排序操作属于内部确定性处理，仅改变数据的内部组织形式，不会引入额外的信息泄露。因此，IEX-RS 的设置泄露函数定义与 IEX 方案完全一致。

(2) 查询泄露 $\mathcal{L}_{\text{Search}}$ 函数

查询泄露函数 $\mathcal{L}_{\text{Search}}$ 的形式化定义为 $\mathcal{L}_{\text{Search}} = (\# \text{labels}, \mathcal{L}_{\text{Search}}^{\text{dx}}(\text{DX}, w_i), \mathcal{L}_{\text{Setup}}^{\text{mm}}(\text{MM}_i), \text{TagRP}, \mathcal{L}_{\text{Search}}^{\text{mm}}(\text{MM}_i, w_i))$ 。

其中, #labels 表示单个析取查询对应的标签(关键字)数量; $\mathcal{L}_{\text{Search}}^{\text{dx}}(\text{DX}, w_i)$ 表示依据 w_i 生成令牌搜索字典 DX 时造成的泄露; $\mathcal{L}_{\text{Setup}}^{\text{mm}}(\text{MM}_i)$ 表示多映射 MM_i 的设置泄露; $\mathcal{L}_{\text{Search}}^{\text{mm}}(\text{MM}_i, w_i)$ 表示依据 w_i 生成令牌搜索多映射时造成的泄露, 其形式化定义为:

$$\mathcal{L}_{\text{Search}}^{\text{mm}}(\text{MM}_i, w_i) = (\mathcal{L}_{\text{Search}}^{\text{mm}}(\text{MM}_g, w_i), \dots, \mathcal{L}_{\text{Search}}^{\text{mm}}(\text{MM}_i, w_i))_{i \in [q-1]}, \mathcal{L}_{\text{Search}}^{\text{mm}}(\text{MM}_g, w_q))$$

多映射搜索泄露信息具体涵盖各个多映射的搜索模式、访问模式(即之前的查询历史中是否出现相同的查询和 tag), 同时还包含查询多映射的响应结果大小。TagRP 表示查询过程中得到的 tag 重复模式。它为一个序列数组, 其中每个条目记录了 tag 首次出现时对应的查询索引。若两个查询包含相同的关键字 w , 则它们会生成部分子令牌相同的令牌, 进而导致对多映射的查询返回相同的 tag 值。通过分析 IEX-RS 方案的查询泄露特性, 发现其与 IEX 方案也是一致的。

综上所述, IEX-RS 和 IEX 两个方案在设置泄露和查询泄露上均保持一致, 故两个方案的安全性一致。

下面给出 IEX-RS 方案的安全性定理。

定理 1 令 F 为安全的伪随机函数, Σ_{DX} 为 $(\mathcal{L}_{\text{Setup}}^{\text{dx}}, \mathcal{L}_{\text{Search}}^{\text{dx}})$ 自适应安全的字典加密方案, Σ_{MM} 为 $(\mathcal{L}_{\text{Setup}}^{\text{mm}}, \mathcal{L}_{\text{Search}}^{\text{mm}})$ 自适应安全的多映射加密方案, SKE 方案 Σ 满足 IND-CPA 安全, IEX-RS 方案泄露函数定义为 $\mathcal{L}_{\text{IEX-RS}} = (\mathcal{L}_{\text{Setup}}, \mathcal{L}_{\text{Search}})$ 。若泄露函数包含以下泄露信息:

$$\begin{aligned} \mathcal{L}_{\text{Setup}} &= (\mathcal{L}_{\text{Setup}}^{\text{mm}}(\text{MM}_g), \mathcal{L}_{\text{Setup}}^{\text{dx}}(\text{DX})) \\ \mathcal{L}_{\text{Search}} &= (\text{\#labels}, \mathcal{L}_{\text{Search}}^{\text{dx}}(\text{DX}, w_i), \\ &\quad \mathcal{L}_{\text{Setup}}^{\text{mm}}(\text{MM}_i), \text{TagRP}, \mathcal{L}_{\text{Search}}^{\text{mm}}(\text{MM}_i, w_i)) \end{aligned}$$

则 IEX-RS 方案满足 $\mathcal{L}_{\text{IEX-RS}}$ 自适应安全。

接下来构造一系列游戏来证明定理 1。

证明 本文主要采用混合论证技术证明定理 1。构造真实游戏 G_0 , 再构造游戏 G_1 、 G_2 和 G_3 , 其中 G_3 为理想游戏。在构造的一系列游戏中, 相邻游戏之间均不可区分。依据不可区分性的传递性, 可以证明真实游戏 $\text{Real}_{\mathcal{A}}^{\text{IEX-RS}}$ 输出 1 的概率

与理想游戏 $\text{Ideal}_{\mathcal{A}, S}^{\text{IEX-RS}}$ 输出 1 的概率之间的差异是可忽略不计的, 由此可证明 IEX-RS 方案满足 $\mathcal{L}_{\text{IEX-RS}}$ 自适应安全。

构造游戏 G_0 : $G_0 = \text{Real}_{\mathcal{A}}^{\text{IEX-RS}}(\lambda)$, 服务器正确执行 IEX-RS 方案, 输入明文数据库 DB 和查询序列 DQ。

构造游戏 G_1 : 游戏 G_1 与游戏 G_0 基本一致, 除了游戏 G_1 使用多映射 MM 的仿真器 S_{MM} 模拟生成多映射。在使用仿真器 S_{MM} 时, 输入设置泄露 $\mathcal{L}_{\text{Setup}}^{\text{mm}}(\text{MM}_g)$, 输出 EMM_g 。由于 S_{MM} 满足自适应语义安全, 在敌手 $\mathcal{A}$ 的视图中, 游戏 G_1 通过仿真器 S_{MM} 生成的多映射与游戏 G_0 正确执行得到的多映射在计算上是不可区分的。故敌手无法区分处于游戏 G_1 还是游戏 G_0 中, 即 G_1 和 G_0 是不可区分的。

构造游戏 G_2 : 游戏 G_2 与 G_1 基本一致, 除了游戏 G_2 使用字典 DX 的仿真器 S_{DX} 模拟生成字典。在使用仿真器 S_{DX} 时, 输入设置泄露 $\mathcal{L}_{\text{Setup}}^{\text{dx}}(\text{DX})$, 输出 EDX。由于 S_{DX} 满足自适应语义安全, 在敌手 $\mathcal{A}$ 的视图中, 游戏 G_2 通过仿真器 S_{DX} 生成的字典与游戏 G_1 正确执行得到的字典在计算上是不可区分的。因此敌手无法区分处于游戏 G_2 还是游戏 G_1 中, 即 G_2 和 G_1 是不可区分的。

构造游戏 G_3 : $G_3 = \text{Ideal}_{\mathcal{A}, S}^{\text{IEX-RS}}(\lambda)$, 设定一个仿真器 S , 仿真器输入为 IEX-RS 方案的设置泄露 $\mathcal{L}_{\text{Setup}}$ 、查询泄露 $\mathcal{L}_{\text{Search}}$ 和多项式数量查询范式 $\Phi = (\Phi_1, \dots, \Phi_n)$, 仿真器输出查询令牌 tok。其中查询 Φ_i 泄露的形式化定义为:

$$\begin{aligned} \mathcal{L}_{\text{Search}}(\text{EDB}, \Phi_i) &= \mathcal{L}_{\text{Search}}^{\text{dx}}(\text{DX}, w_i), \mathcal{L}_{\text{Setup}}^{\text{mm}}(\text{MM}_i), \\ &\quad \mathcal{L}_{\text{Search}}^{\text{mm}}(\text{MM}_g, w_i), \mathcal{L}_{\text{Search}}^{\text{mm}}(\text{MM}_i, w_{i+1}), \dots, \\ &\quad \mathcal{L}_{\text{Search}}^{\text{mm}}(\text{MM}_i, w_q), \mathcal{L}_{\text{Search}}^{\text{mm}}(\text{MM}_g, w_q), \\ &\quad \text{TagRP}(\text{EDB}, \Phi_i) \end{aligned}$$

仿真器 S 算法如算法 4 所示。使用仿真器 S_{DX} 时, 输入字典查询泄露 $\mathcal{L}_{\text{Search}}^{\text{dx}}(\text{DX}, w_j)$; 使用仿真器 S_{MM} 时, 输入多映射查询泄露 $\mathcal{L}_{\text{Search}}^{\text{mm}}(\text{MM}_i, w_i)$ 。结合两个仿真器的输入, 得到最终的查询令牌。

算法 4 仿真器 S

输入 $\mathcal{L}_{\text{Setup}}, \mathcal{L}_{\text{Search}}, \Phi = (\Phi_1, \dots, \Phi_n)$

输出 EMM_g, EDX, tok

```

1       $EDX \leftarrow S_{DX}(\mathcal{L}_{Setup}^{dx}(DX))$ 
2       $EMM_g \leftarrow S_{MM}(\mathcal{L}_{Setup}^{mm}(MM_g))$ 
3      For  $i = 1$  to  $n$  do
4          for  $j = i$  to  $q - 1$  do
5               $gtk_j \leftarrow S_{MM}(\mathcal{L}_{Search}^{mm}(MM_g, w_j),$ 
                 $(tag)_{id \in DB(w_j)})$ 
6               $dtk_j \leftarrow S_{DX}(\mathcal{L}_{Search}^{dx}(DX, w_j), EMM_j)$ 
7              for  $k = j + 1$  to  $n$  do
8                   $ltk_j \leftarrow S_{MM}(\mathcal{L}_{Search}^{mm}(MM_j, w_k),$ 
                     $(tag)_{id \in DB(w_j) \cap DB(w_k)})$ 
9              end for
10              $tok_j \leftarrow (gtk_j, dtk_j, ltk_{j+1}, \dots, ltk_q)$ 
11         end for
12          $gtk_q \leftarrow S_{MM}(\mathcal{L}_{Search}^{mm}(MM_g, w_q),$ 
                 $(tag)_{id \in DB(w_q)})$ 
13          $tok \leftarrow (tok_1, \dots, tok_{q-1}, gtk_q)$ 
14     end for

```

在仿真器 S 中, 令牌中的 gtk 、 dtk 和 ltk 均由仿真器输入对应的查询泄露得到。仿真器 S 只输入上述定义的泄露函数, 并不涉及任何真实的数据。由于仿真器 S_{DX} 和仿真器 S_{MM} 满足自适应安全, 因此在敌手 $\mathcal{A}$ 的视图中, 游戏 G_3 中通过仿真器 S_{DX} 和仿真器 S_{MM} 生成的查询令牌与游戏 G_2 中生成的令牌在计算上是不可区分的。由此可证明敌手无法区分处于游戏 G_3 还是游戏 G_2 中, 即 G_3 和 G_2 是不可区分的。

结合以上游戏, 可推断出仿真器 S 生成的视图和游戏 G_0 生成的视图是不可区分的, 则以下不等式成立。

$$\left| \Pr[\text{Real}_{\mathcal{A}}^{\text{IEX-RS}}(\lambda) = 1] - \Pr[\text{Ideal}_{\mathcal{A}, S}^{\text{IEX-RS}}(\lambda) = 1] \right| \leq \text{negl}(\lambda)$$

5 实验

本文从初始化开销、令牌生成开销和搜索效率 3 个方面, 对 IEX 方案和 IEX-RS 方案开展性

能对比实验。本文实验基于 Java 语言的 Clusion 框架实现 IEX 方案和 IEX-RS 方案, 使用 Bouncy Castle 密码库提供加密功能。实验在单一设备上运行客户端和服务端, 设备配置为 Intel(R) Xeon (R) CPU E5-2643 v4 @ 3.40 GHz、128 GB 内存和 500 GB 硬盘, 搭载 Ubuntu 22.04.3 LTS 系统。本文的伪随机函数 (pseudo-random function, PRF) 采用 HMAC-SHA256 实现, 对称加密方案选用 AES (advanced encryption standard) 算法的 CTR (counter) 模式。所有方案的文档标识符长度统一设置为 32 位。在实验过程中, 将 IEX 方案的过滤参数 p 设为 0。

实验选用安然数据集, 考虑到加密后的存储空间需求, 实验选取不同规模的子集 (10^3 、 10^4 和 10^5 数量级的键值对数据集) 进行测试。基础的多映射实现采用文献[6]提出的 2Lev 方案, 该方案支持响应结果隐藏功能。

5.1 初始化开销

为比较 IEX 方案和 IEX-RS 方案的初始化开销, 实验选取了 10^3 、 10^4 和 10^5 数量级 (分别包含 278、2 264 和 12 435 个关键字) 的键值对数据集进行测试。为提高实验结果的准确性, 每组数据集的实验重复进行 30 次, 并对实验结果取平均值。实验重点测量服务器存储开销及初始化时间, 其中借助 Java Object Layout 库来测量各对象的内存大小。两个方案的服务器存储开销与初始化时间对比如表 2 所示。

表 2 IEX 和 IEX-RS 的服务器存储开销和初始化时间对比
Table 2 Comparison of server storage overhead and initialization time between IEX and IEX-RS

方案	键值对数量	服务器存储开销/MB		初始化时间/s
		全局多映射 MM_g	字典 DX	
IEX	10^3	0.96	107.44	16.02
	10^4	8.91	2 427.07	347.06
	10^5	46.70	66 726.10	2 935.28
IEX-RS	10^3	0.96	53.27	14.39
	10^4	8.91	1 209.92	293.63
	10^5	46.71	33 343.57	2 016.97

针对不同规模的数据集,实验结果表明 IEX 方案和 IEX-RS 方案的全局多映射所占存储空间大小基本一致。在 10^3 级别的数据集上, IEX-RS 方案的字典所占存储空间较 IEX 方案减少 50.41%, 服务器整体存储开销较 IEX 方案减少 49.96%。在 10^4 级别的数据集上, IEX-RS 方案的字典所占存储空间较 IEX 方案减少 50.14%, 服务器整体存储开销较 IEX 方案减少 49.96%。在 10^5 级别的数据集上, IEX-RS 方案的字典所占存储空间较 IEX 方案减少 50.02%, 服务器整体存储开销较 IEX 方案减少 49.99%。

在代码中, tag 由 AES 加密生成, 加密前后长度保持不变。实验结果表明, IEX-RS 方案的字典存储空间较 IEX 方案减少近 50%, 这与理论分析完全吻合。进一步分析发现, 随着键值对数量级的增加, 全局多映射的存储开销虽然增长, 但其绝对大小仍保持在 MB 级别。相比之下, 字典的存储开销则急剧增长至 GB 级别。因此, 随着键值对数量级的提升, 字典存储空间的优化对服务器整体存储开销的影响愈发显著, 进而使得整体存储开销的减少比例逐步上升。

在初始化时间表现上, IEX-RS 方案较 IEX 方案呈现出 10.17% ~ 31.29% 的优化幅度。IEX-RS 在为每个关键字生成字典时, 会主动过滤掉排序在该关键字之前的其他关键字, 从而大幅减少需要处理的交集数据量。随着键值对数量级增长, 被过滤的无效交集数据量呈递增趋势, 使得 IEX-RS 在大规模数据集下的初始化时间优化效果更为显著。需要指出的是, 两个方案的初始化算法在理论复杂度上并无差异。IEX-RS 的优化核心并非对加密算法本身进行改进, 而是通过减少实际需要加密的键值对数量, 实现初始化效率的提升。

5.2 令牌生成开销

在令牌生成算法方面, IEX-RS 与 IEX 的核心差异在于前者在原有流程中新增了关键字排序步骤, 而其他部分完全一致。这种设计使得两个方案生成的最终令牌大小相同。不过, 由于排序操作的引入, IEX-RS 的令牌生成时间略高于 IEX。实验中, 将输入关键字的数量作为主变量来测量令牌生成时间, 因为两个方案的令牌生成时间仅

与输入的关键字相关。实验将输入的关键字数量分别设置为 50 个、100 个、150 个、200 个和 250 个。两个方案的令牌生成时间对比结果如图 2 所示。

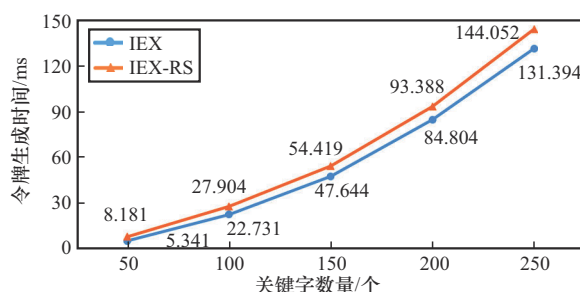


图2 IEX 和 IEX-RS 的令牌生成时间对比

Figure 2 Comparison of token generation time between IEX and IEX-RS

从图 2 的实验数据可以看出, 随着输入关键字数量的递增, IEX 与 IEX-RS 方案的令牌生成时间差距呈现逐渐扩大的趋势。这是因为 IEX-RS 在令牌生成算法中引入了额外的排序算法, 而排序算法的计算开销与输入的关键字数量呈正相关, 因此输入的关键字数量越多, IEX-RS 的排序算法计算开销越大, 导致两个方案的令牌生成时间差距越大。

5.3 搜索效率

在搜索算法设计层面, IEX 方案与 IEX-RS 方案基本一致。搜索时, 算法仅需从加密的全局多映射中提取相关数据, 并结合字典的查询结果进行过滤。得益于该高效的设计流程, 两种方案均展现出较高的搜索效率。

由于 10^3 规模的数据集中文档数量较少, 其搜索结果的波动性较大, 因此未将其纳入搜索效率测试范围。搜索效率实验在 10^4 和 10^5 规模的数据集上开展, 聚焦于测量服务器执行析取查询操作的耗时, 且测量结果不包括令牌生成时间和网络时延。为确保实验结果的准确性, 每组搜索效率实验均重复进行 500 次, 并对实验结果取平均值。两种方案在不同规模数据集上的搜索效率对比结果分别如图 3 和图 4 所示。所有测试查询均采用固定形式 $\Phi=(w_1 \vee w_2 \vee w_3 \vee w_4)$, 令牌生成时输入的关键字数量固定为 4 个。实验结果表明, IEX-RS 方案与 IEX 方案的搜索效率差距较小。这一现象的根本原因在于, 两个方案在执行

搜索协议时,本质上是对不同大小的加密数据集执行相同的搜索过滤操作。

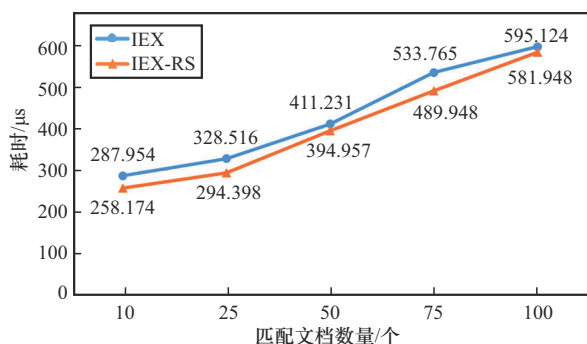


图3 10^4 级别数据集的方案搜索效率对比

Figure3 Search efficiency comparison in 10^4 size dataset

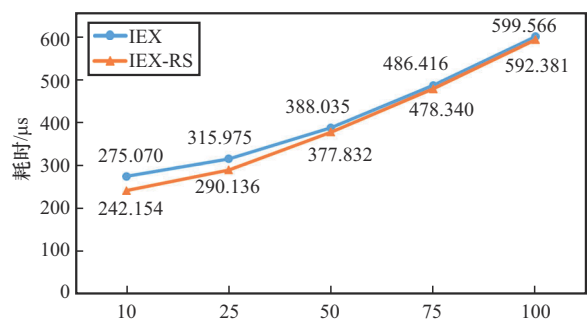


图4 10^5 级别数据集的方案搜索效率对比

Figure4 Search efficiency comparison in 10^5 size dataset

6 结束语

本文提出一种针对交集数据存储的优化方法,并基于此设计了高效布尔查询方案IEX-RS,其核心优势在于可以显著减少服务器存储开销,同时保证搜索复杂度与原方案IEX一致。实验数据表明,与IEX方案相比,IEX-RS可以减少约50%的服务器存储开销,减少10.17%~31.29%的初始化时间。IEX-RS在令牌生成阶段因新增关键字排序步骤,导致生成时间略高于IEX,但两者生成的令牌大小一致。当数据库关键字数量较多时,排序操作的计算开销主要集中在系统初始化阶段,且在实际应用中用户查询的关键字数量通常有限,因此排序引入的额外开销是可以接受的。

IEX-RS方案通过增加初始化算法中的排序操作换取服务器存储开销的显著减少,但该方案在数据库条目动态更新场景中存在局限性。为解决这一问题,动态布尔查询方案的研究成为关键方向。目前已有相关研究(如文献[32])聚焦于

动态布尔查询方案的实现,而构建同时满足前向安全与后向安全的动态布尔查询方案可以作为下一步的研究方向。

参考文献:

- [1] SONG D X, WANGER D, PERRIG A. Practical techniques for searches on encrypted data[C]// Proceeding of 2000 IEEE Symposium on Security and Privacy. Piscataway:IEEE Press, 2000: 44-55.
- [2] CHASE M, KAMARA S. Structured encryption and controlled disclosure[C]// Proceeding of Advances in Cryptology -ASIACRYPT 2010. Berlin: Springer, 2010: 577-594.
- [3] CASH D, JARECKI S, JUTLA C, et al. Highly-scalable searchable symmetric encryption with support for Boolean queries[C]// Proceeding of Advances in Cryptology - CRYPTO 2013. Berlin: Springer, 2013: 353-373.
- [4] KAMARA S, MOATAZ T. Boolean searchable symmetric encryption with worst-case sub-linear complexity[C]//Proceeding of Advances in Cryptology - EUROCRYPT 2017. Berlin: Springer, 2017: 94-124.
- [5] CURTMOLA R, GARAY J, KAMARA S, et al. Searchable symmetric encryption: improved definitions and efficient constructions[C]// Proceeding of the 13th ACM Conference on Computer and Communications Security. New York: ACM Press, 2006: 79-88.
- [6] CASH D, JAEGER J, JARECKI S, et al. Dynamic searchable encryption in very-large databases: data structures and implementation[C]// Proceeding of the Network and Distributed System Security Symposium (NDSS '14). 2014: 23-26.
- [7] BOST R. Σ o ρ o ς : forward secure searchable encryption[C]//Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM Press, 2016: 1143-1154.
- [8] BOST R, MINAUD B, OHRIMENKO O. Forward and backward private searchable encryption from constrained cryptographic primitives[C]//Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM Press, 2017: 1465-1482.
- [9] GOLLE P, STADDON J, WATERS B. Secure conjunctive keyword search over encrypted data[C]// Proceedings of Applied Cryptography and Network Security. Berlin: Springer, 2004: 31-45.
- [10] BALLARD L, KAMARA S, MONROSE F. Achieving efficient conjunctive keyword searches over encrypted data[C]//Proceedings of Information and Communications Security (ICICS 2005). Berlin: Springer, 2005: 414-426.
- [11] PAPPAS V, KRELL F, VO B, et al. Blind seer: a scalable private DBMS[C]//Proceedings of 2014 IEEE Symposium on Security and Privacy. Piscataway:IEEE Press, 2014: 359-374.
- [12] FISCH B A, VO B, KRELL F, et al. Malicious-client security in blind seer: a scalable private DBMS [C]//Proceedings of 2015 IEEE Symposium on Security and Privacy. Piscataway:IEEE Press, 2015: 395-410.
- [13] BELLARE M, HOANG V T, ROGAWAY P. Foundations of garbled circuits [C]//Proceedings of the 2012 ACM Conference on

- Computer and Communications Security. New York: ACM Press, 2012: 784-796.
- [14] LI R, LIU A X. Adaptively secure conjunctive query processing over encrypted data for cloud computing[C]//Proceedings of 2017 IEEE 33rd International Conference on Data Engineering (ICDE). Piscataway: IEEE Press, 2017: 697-708.
- [15] WU Z, LI K. VBTREE: forward secure conjunctive queries over encrypted data for cloud computing[J]. The VLDB Journal, 2019, 28 (1): 25-46.
- [16] ZUO C, SUN S F, LIU J K, et al. Forward and backward private dynamic searchable symmetric encryption for conjunctive queries[J]. Cryptology ePrint Archive, 2020: 1357.
- [17] ISLAM M S, KUZU M, KANTARCIOGLU M. Access pattern disclosure on searchable encryption: ramification, attack and mitigation [C]//Proceedings of the 2012 Network and Distributed System Security Symposium (NDSS 2012). Virginia: the Internet Society, 2012.
- [18] CASH D, GRUBBS P, PERRY J, et al. Leakage-abuse attacks against searchable encryption[C]//Proceedings of the 2015 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM Press, 2015: 668-679.
- [19] ZHANG Y P, KATZ J, PAPAMANTHOU C. All your queries are belong to us: the power of file-injection attack on searchable encryption[C]//Proceedings of the 25th USENIX Security Symposium (USENIX Security 16). Berkeley: USENIX Association, 2016: 707-720.
- [20] LAI S, PATRANABIS S, SAKZAD A, et al. Result pattern hiding searchable encryption for conjunctive queries[C]//Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18). New York: ACM Press, 2018: 745-762.
- [21] PATRANABIS S, MUKHOPADHYAY D. Forward and backward private conjunctive searchable symmetric encryption[C]//Proceedings of the 28th Annual Network and Distributed System Security Symposium (NDSS 2021). Virginia: the Internet Society, 2021: 342-345.
- [22] MA C, GU Y, LI H. Practical searchable symmetric encryption supporting conjunctive queries without keyword pair result pattern leakage[J]. IEEE Access, 2020, 8: 107510-107526.
- [23] ZUO C, LAI S, YUAN X, et al. Searchable encryption for conjunctive queries with extended forward and backward privacy[J]. Cryptology ePrint Archive, 2021: 1585.
- [24] SUN S F, STEINFELD R, LAI S, et al. Practical non-interactive searchable encryption with forward and backward privacy[C]//Proceedings of Network and Distributed System Security Symposium. Virginia: the Internet Society, 2021.
- [25] JUTLA C, PATRANABIS S. Efficient searchable symmetric encryption for join queries[C]//Proceedings of Advances in Cryptology - ASIACRYPT 2022. Berlin: Springer, 2022: 304-333.
- [26] YUAN D, ZUO C, CUI S, et al. Result-pattern-hiding conjunctive searchable symmetric encryption with forward and backward privacy [C]//Proceedings of Privacy Enhancing Technologies, 2023: 40-58.
- [27] BAG A, TALAPATRA D, RASTOGI A, et al. Two-in-one-sse: fast, scalable and storage-efficient searchable symmetric encryption for conjunctive and disjunctive boolean queries[C]//Proceedings of Privacy Enhancing Technologies. 2023: 115-139.
- [28] BAG A, PATRANABIS S, MUKHOPADHYAY D. Tokenised multi-client provisioning for dynamic searchable encryption with forward and backward privacy[C]//Proceedings of the 19th ACM Asia Conference on Computer and Communications Security. New York: ACM Press, 2024: 1691-1707.
- [29] YUAN D, CUI S, RUSSELLO G. Result pattern hiding Boolean searchable encryption: achieving negligible false positive rates in low storage overhead[J]. Cryptology ePrint Archive, 2024.
- [30] PATEL S, PERSIANO G, SEO J Y, et al. Efficient Boolean search over encrypted data with reduced leakage[C]//Proceedings of Advances in Cryptology - ASIACRYPT 2021. Berlin: Springer, 2021: 577-607.
- [31] 郭斯栩, 何申, 栗栗, 等. 基于多关键字的 Top-k 布尔可搜索加密方案[J]. 计算机研究与发展, 2022, 59(8): 1841-1852.
- GUO S X, HE S, SU L, et al. Top-k Boolean searchable encryption scheme based on multiple keywords[J]. Journal of Computer Research and Development, 2022, 59(8): 1841-1852.
- [32] FERREIRA B, PORTELA B, OLIVEIRA T, et al. Boolean searchable symmetric encryption with filters on trusted hardware[J]. IEEE Transactions on Dependable and Secure Computing, 2022, 19(2): 1307-1319.
- [33] 杨紫依, 马昌社, 谢南江. 支持联合查询和交叉模式隐藏的结构加密[J]. 网络与信息安全学报, 2024, 10(3): 175-187.
- YANG Z Y, MA C S, XIE N J. Cross pattern hiding structured encryption for conjunctive queries[J]. Chinese Journal of Network and Information Security, 2024, 10(3): 175-187.

[作者简介]



张象博 (2000-), 男, 江西吉安人, 华南师范大学硕士生, 主要研究方向为可搜索加密。



马昌社 (1974-), 男, 湖南邵阳人, 博士, 华南师范大学教授, 主要研究方向为加密数据隐私计算、应用密码技术。



苏雨晨 (2000-), 女, 湖南益阳人, 华南师范大学硕士生, 主要研究方向为可搜索加密、加密数据库。