Forward and Backward Private Conjunctive Searchable Symmetric Encryption 阅读笔记

李忆诺

2024 年 4 月 9 日

1 基本信息

1.1 论文来源

Patranabis S, Mukhopadhyay D. Forward and backward private conjunctive searchable symmetric encryption[J]. Cryptology ePrint Archive, 2020.

1.2 概述

本文提出了一种能同时满足前向或后向隐私的，用于连接关键词搜索的动态 SSE 方案：**遗忘动态交叉标签 (ODXT)**。它支持扩展到任意结构的大型数据库，且通过有效地支持大型数据库的快速更新和连接关键词查询，它实现了性能和安全性的现实平衡。

2 论文要点

2.1 背景

动态 SSE 具有两个关键的安全属性：前向隐私和后向隐私。当前的大部分工作都集中于研究**单关键字搜索**的动态 SSE 方案，但要提高可搜索加密的实用性，至少应该支持**连接关键字搜索**。大多数具有次线性复杂度的方案都不能支持更新，唯有的两个方案 Kamara 和 Moataz 又不能同时满足前向和后向隐私。因此，本文提出了一种能同时满足前向或后向隐私的，用于连接关键词搜索的动态 SSE 方案。

2.2 价值

本文提出的 ODXT 方案主要贡献如下：

- 设计了一种支持单轮通信的搜索协议。通过预先计算部分协议消息，并将其加密存储在服务器上，在搜索过程中，客户机只需提供一些辅助信息供服务器进行解密，无需进行任何进一步交互。其搜索性能相当接近于静态设置下的联合 SSE 方案所实现的最佳搜索复杂度。

- 设计了一个专门的数据结构**动态交叉标签**，支持更新和搜索，同时确保前向和后向隐私。
- 设计了一种对于“连接关键词搜索”的减少循环技术，可将“消息预计算”与“更新操作”结合起来，无需在设置时进行预处理。

2.3 问题陈述

本文的方案基于 $MITRA$ 方案进行扩展，旨在对以下部分进行改进：

- 将适用于单关键字搜索的 $MITRA$ 修改成适用于连接关键字搜索的 $MITRA_{CONJ}$;
- 减小计算开销——基于静态交叉标签引入新的语义，构造能够在每次更新操作中实时计算的动态交叉标签；
- 减少由于连接关键字搜索的扩展部分带来的信息泄露

2.4 方法

2.4.1 连接关键字搜索 $MITRA_{CONJ}$

假设客户端需要查询一个关键字集合 $q = (w_1 \wedge w_2 \wedge \dots w_n)$，则分别对每一个关键字 w_i 而言，都是进行单关键字查询，最终将查询结果取交集即为所求。

因此，对于每个关键字 w_i 仍使用 $MITRA$ 方案。除此以外，本文引入了一个键值字典 $TSet$，它存储的值任意一个 w 的每个操作的加密结果，而键由 PRF 函数生成，密钥掌握在客户端手中。键用于指定服务器上的 $TSet$ 加密数据库中，对应值的存放地址。

则该方案更新与查询步骤如下：

更新：

- 客户端申请更新 $op(id, w)$；
- 客户端本地存储的 $TSet$ 生成相应键值对；
- 客户端将相关信息发送给服务器，服务器根据键值对进行数据更新。

查询：

- 客户端发出合取查询 $q = (w_1 \wedge w_2 \wedge \dots w_n)$；
- 客户端本地存储的 $TSet$ 计算获得所有涉及 w_i 的更新操作的键；
- 服务器对每一个 w_i 实行 $MITRA$ 方案，根据键寻找对应地址的值；
- 客户端接收并解密，最终获得目标文本集合。

该方案不会泄露更新时的任何信息，满足前向隐私。

由于该方案的搜索复杂度很高，且该方案泄露了 $|Upd(w_i)|$（包括与本次查询无关的文件）和时间戳。为了减少计算开销和泄露，进行以下改进。

2.4.2 动态交叉标签 $BDXT$

本文基于静态交叉标签方案，通过扩展其语义，引入 op，使得该标签可用来表示对于给定 (id, w) 是否发生了更新操作 $(add|del)$。

为了实现该动态交叉标签，引入了新的 $XSet$，该键值字典用于给任意一个 (id, w) 对应一个键：用于表示是否进行插入、删除操作。

$XSet$ 值的计算方式为：

$$xtag_{i,j,op} = F(K_X, w_i||id_j||op)$$

因此，上述查询操作步骤改为：

查询：

- 客户端发出合取查询 $q = (w_1 \wedge w_2 \wedge \ldots w_n)$，其中假设 w_1 是 q 中出现频率最少的关键字，即 $min(|DB_{cnt}(w_i)|)$；
- 客户端本地存储的 $TSet$ 计算获得涉及 w_1 的更新操作的键；
- 服务器对 w_1 实行 $MITRA$ 方案，根据键寻找对应地址的值；
- 客户端接收并解密，获得文本集合；
- 客户端计算交叉标签对，并将其以随机排列顺序发给服务器；
- 服务器检索 $XSet$，返回满足的交叉标签对；
- 客户端根据服务器返回结果筛选，获得目标文本集合。

该方案减小了计算开销，同时客户端存储与数据集更新操作呈对数增长，服务器存储与数据集更新操作呈线性增长，通过将泄露与出现频率最低的 w_1 联系起来，该方案成功降低了泄露，其信息包括 $|DB(w_1)|$、$|Upd(w_1)|$ 和交叉标签对带来的其他信息泄露。

除此之外，由于动态标签特点，该方案不会泄露更新时的任何信息，满足前向隐私；该方案以对称的方式处理插入和删除操作，通过使用相同的 PRF 为它们生成交叉标签，确保攻击者无法区分关于关键字的插入条目和删除条目，因此满足后向隐私。

2.4.3 遗忘动态交叉标签 $ODXT$

我们希望将对 w_i 和 id_j 的交叉标签对的计算过程交给服务器，通过不显式恢复 $DB(w_1)$，而启动交叉标签的遗忘计算，避免额外一轮交互，减小计算开销与泄露。

修改 $XSet$：

在 ODXT 中，$XSet$ 值的计算方式改为：

$$xtag_{i,j,op} = g^{F_p(K_X, w_i) \cdot F_p(K_Y, id_j||op)}$$

修改 $TSet$：

在 ODXT 中，$TSet$ 额外存储一个动态致盲因素：

$$\alpha_{i,j,op} = F_p(K_Y, id_j||op) \cdot (F_p(K_Z, w_i||cnt))^{-1}$$

因此，上述查询操作步骤改为：

查询：

- 客户端发出合取查询 $q = (w_1 \wedge w_2 \wedge \ldots w_n)$，其中假设 w_1 是 q 中更新操作次数最少的关键字，即 $min(|Upd_{cnt}(w_i)|)$；
- 客户端根据本地存储的 $TSet$ 计算获得涉及 w_1 的更新操作的键；
- 客户端根据本地存储的 $XSet$ 计算 $xtoken_{i,cnt} = g^{F_p(K_X,w_i)\cdot F_p(K_Z,w_1||cnt)}$，用于协助服务器进行交叉标签对计算；
- 服务器对 w_1 实行 $MITRA$ 方案，根据键寻找对应地址的值；
- 服务器根据 $xtokenList$ 计算交叉标签对，进行文本筛选；
- 客户端接收并解密，获得目标文本集合；

该方案避免了额外一轮交互，客户端存储与数据集更新操作呈对数增长，但相比于 BDXT 减少了一半，服务器存储与数据集更新操作呈线性增长，与 BDXT 相同。通过将泄露与更新操作次数最少的 w_1 联系起来。在实际工程中，出现频率最低的关键字也可能涉及较少的更新操作，因此 BDXT 和 ODXT 对于大多数联合查询可能是相同的。

由于 $TSet$ 特性，ODXT 在更新时是无泄漏的；在搜索过程中，服务器会学习到涉及连接关键词的文档标识符的最终集合，$|Upd(w_1)|$，了解多个连接关键词查询是否有相同的 w_1，对于每个涉及 s 项的更新操作，服务器学习每个 x 项的更新操作总数及对应时间戳。

2.5 结果

2.5.1 数据集

来自维基媒体下载的 60.92GB 大小的真实数据集，包含 1600 万个文档和 4300 万个关键字。通过从原始数据集中随机插入和删除文档到空数据集来模拟更新，共执行了 108 次更新操作，其中 30% 为删除操作。

2.5.2 泄露

ODXT 方案是自适应前向和后向私有的，其泄露情况如下：

- 输出泄露：几乎所有现有的 SSE 方案，包括静态和动态方案，在单关键词和连接关键词搜索过程中都会产生输出泄露（结果模式泄露），这通常被认为是可以接受的。
- $s-term$ 泄露：即关于 w_1 的泄露，这是由于该方案的设计范式产生，但由于目前还没有已知针对静态或动态联合 SSE 方案的相关攻击，作者认为 ODXT 方案似乎也不会受到任何已知攻击；
- $x-term$ 泄露：即关于除 w_1 外的关键字的泄露，由于攻击者针对此类泄露进行攻击，必须能够计算 $|DB(w_1) \cap DB(w_i)|$，而 ODXT 不足以计算此项，不容易受到此类攻击。

2.5.3 复杂度

ODXT 需要在联合搜索期间在客户机和服务器之间进行单轮通信。客户端和服务器端的计算开销及通信开销都以 $O(n \cdot |Upd(w_1)|)$ 为尺度进行扩展。

2.5.4 将 ODXT 扩展至多客户端

由于 ODXT 消除了在联合搜索期间，客户端和服务器进行额外一轮通信的需要，这使得 ODXT 能够抵抗在多客户端模式下，服务器与恶意客户端勾结并恢复诚实客户端相关信息的攻击。由于从服务器到每个客户机的唯一消息是客户机查询相对应文件标识符的最终列表，因此不存在恶意客户机可以推断没有权限的中间信息。

因此，ODXT 可以用于受控披露，并部署在多客户机设置中。除此以外，使用相关技术可以进一步扩展 ODXT，使其不仅对服务器隐藏客户端发出的查询，对令牌颁发机构也能隐藏查询。

3 评论

3.1 局限性

$s-term$ 和 $x-term$ 泄露：本文提到，该类型泄露是由于设计范式产生，或许可通过使用 $ORAM-style$ 数据结构来防止此类泄露，或使用填充之类的容量隐藏技术。

是否能够设计一个泄露信息更少的方案，该方案只显示与最终查询相关的更新历史，且隐藏与 w_1 相关的所有信息。

3.2 扩展阅读

Forward and Backward Private Conjunctive Searchable Symmetric Encryption

Privacy-Preserving Multi-Keyword Top-k k Similarity Search Over Encrypted Data

3.3 启示

将可搜索加密方案从单关键字扩展为多关键字的连接字查询，是提高 SSE 方案的实用性的十分关键的一步。在现实过程中，更多情况下我们通过搜索若干关键字来查询所需的文本条目。因此，在保证信息安全性的情况下，如何减小开销提升查询效率便显得至关重要。本文提出的 ODXT 方案通过引入新的数据结构并改变其语义，避免额外一轮交互，基于单关键字搜索方案成功扩展到多关键字搜索。