软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2023,34(11):5424−5441 [doi: 10.13328/j.cnki.jos.006743]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于弹性秘密共享的多方门限隐私集合交集协议*

张 恩[1,2], 秦磊勇[1], 杨刃林[1], 李功丽[1,2]

[1](河南师范大学 计算机与信息工程学院, 河南 新乡 453007)

[2](智慧商务与物联网技术河南省工程实验室 (河南师范大学), 河南 新乡 453007)

通信作者: 张恩, E-mail: 121096@htu.edu.cn

摘 要: (t,n)门限隐私集合交集协议, 指N个参与者各自拥有大小为n的隐私集合, 在不泄露自身隐私信息的前提下, 如果各参与者交集数量大于门限值t, 则参与各方能够获得交集信息, 其有广泛的应用, 如指纹识别、在线拼车、相亲网站等. 然而现有门限隐私集合交集协议大多针对两方参与者进行研究, 对多方门限隐私集合交集协议的研究仍存在许多挑战, 现有的多方门限隐私集合交集协议使用全同态加密等开销较大的公钥算法, 尚没有有效实现. 针对上述问题, 结合弹性秘密共享、布隆过滤器提出两种有效的多方门限隐私集合交集协议, 并首次仿真实现了协议. 首先, 设计一种新的布隆过滤器构造方法, 将弹性秘密共享生成的份额与参与方的集合元素相对应, 通过查询布隆过滤器获取的秘密子份额能否重构出正确秘密来判断各方交集是否达到门限值, 有效防止交集基数的泄露. 设计的第 1 个协议避免使用开销较大的公钥算法, 当设置安全参数λ为 128, 集合大小为2^{14}, 门限值为$0.8n$时, 在三方场景下协议在线阶段的时间成本为 191 s. 此外, 为了能在半诚实模型下抵抗至多$N-1$个敌手合谋, 在第 1 个协议基础上结合不经意传输设计一种该协议的变体, 相同条件下, 在线阶段时间成本为 194 s. 最后通过安全证明, 证明上述协议在半诚实模型下是安全的.

关键词: 门限隐私集合交集; 抗合谋; 弹性秘密共享; 布隆过滤器; 不经意传输

中图法分类号: TP309

中文引用格式: 张恩, 秦磊勇, 杨刃林, 李功丽. 基于弹性秘密共享的多方门限隐私集合交集协议. 软件学报, 2023, 34(11): 5424–5441. http://www.jos.org.cn/1000-9825/6743.htm

英文引用格式: Zhang E, Qin LY, Yang RL, Li GL. Multi-party Threshold Private Set Intersection Protocol Based on Robust Secret Sharing. Ruan Jian Xue Bao/Journal of Software, 2023, 34(11): 5424–5441 (in Chinese). http://www.jos.org.cn/1000-9825/6743.htm

Multi-party Threshold Private Set Intersection Protocol Based on Robust Secret Sharing

ZHANG En[1,2], QIN Lei-Yong[1], YANG Ren-Lin[1], LI Gong-Li[1,2]

[1](College of Computer and Information Engineering, Henan Normal University, Xinxiang 453007, China)

[2](Engineering Lab of Intelligence Business and Internet of Things of Henan Province (Henan Normal University), Xinxiang 453007, China)

Abstract: A (t, n) threshold private set intersection protocol allows the N participants, each holding a private set of size n, to learn the intersection of their sets only if the number of the elements in their intersection is larger than or equal to the threshold t without revealing their private information. It has a wide range of applications, such as fingerprint recognition, online carpooling, and blind date websites. However, most of the existing research on threshold protocols for private set intersections focuses on two parties. The research on the threshold protocols for multi-party private set intersections is still faced with many challenges. For example, the existing threshold protocols for multi-party private set intersections use public key algorithms with large overheads, such as the fully homomorphic encryption algorithm, and such algorithms have not yet been effectively implemented. To solve the above problems, this study combines robust secret sharing and Bloom filters to propose two effective multi-party threshold private set intersection protocols and implements the protocols by simulation for the first time. Specifically, this study designs a new construction method for Bloom filters. The shares

* 基金项目: 国家自然科学基金 (62072159, 62002103, 62076089, U1804164); 河南省科技攻关计划 (212102210388)
收稿时间: 2021-12-29; 修改时间: 2022-04-03, 2022-06-22; 采用时间: 2022-07-17; jos 在线出版时间: 2023-06-16
CNKI 网络首发时间: 2023-06-19

generated by robust secret sharing are corresponded to the elements in the sets of the participants. Whether the number of the elements in the intersection of the parties reaches the threshold is determined by whether correct secrets can be reconstructed from the secret sub-shares obtained by querying the Bloom filter. In this way, the protocols effectively prevent the revealing of the intersection cardinality. The first protocol designed in this study avoids public key algorithms with large overheads. When the security parameter λ is set to 128, the set size n is set to 2^{14} and the threshold is set to $0.8n$, the time cost of the online phase of the protocols in the three-party scenario is 191 s. In addition, to resist the collusion of at most $N-1$ adversaries in the semi-honest model, this study combines oblivious transfer with the first protocol to design a variant of the first protocol. Under the same condition, the time cost of the online phase is 194 s. Finally, the security proof conducted proves that the proposed protocols are secure in the semi-honest model.

Key words: threshold private set intersection; collusion resistance; robust secret sharing; Bloom filter; oblivious transfer

隐私集合交集 (private set intersection, PSI)[1−10]允许参与方在不泄露自己隐私集合信息的情况下得到自己集合与他人集合的交集信息, 属于安全多方计算的一个特例, 有重要的研究意义与使用价值, 已经在许多领域有广泛应用, 如用于网络数据分析行业的测量广告转化率、通过社交平台搜索潜在用户、用于医疗行业的人类基因对比或者是各国犯罪嫌疑人数据库对比等. 然而, 普通的 PSI 并不适用于某些特定情况, 在隐私保护数据挖掘和机器学习中[11], 数据是在多方之间进行垂直分割的, 各方可能希望得到其数据集的交集并仅在其公共数据集足够大的情况下才开始合作. 如果他们的公共数据集太小, 在这种情况下他们没有强烈动机进行合作. 在具有保护用户隐私的拼车应用中, 多个用户只有在他们的大部分轨迹在地图上相交时才想要拼车. 在这种情况下, 用户仅在各方路线交叉轨迹较大时才会感兴趣[12].

针对这些问题, Freedman 等人[13]提出了门限隐私集合交集 (threshold private set intersection, TPSI) 的概念 (见图 1). 门限隐私集合交集是指有 N 个用户 $P_1,\ldots,P_N$, 所有用户都有各自的元素集合 $X_1,\ldots,X_N$, 在不泄露各自隐私信息的前提下, 如果各方交集数量大于或等于门限值 t, 各方获得交集信息, 经过计算之后用户得到最后的交集结果, 且不知道其他用户的非交集信息.

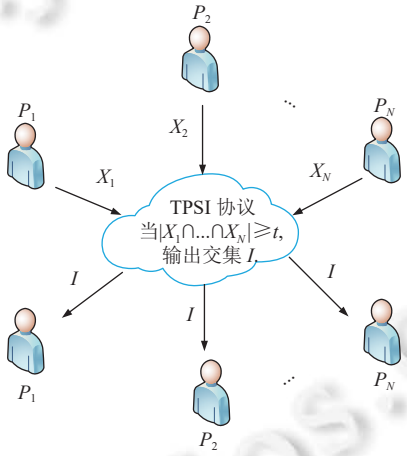

图 1 多方 TPSI 的功能示意图

目前关于 TPSI 的研究方案大多聚焦于两方 TPSI 的场景, 已有部分可实现的两方 TPSI 方案但仍存在运行效率低下且部分研究存在交集基数泄露问题. TPSI 协议相比 PSI 协议的主要区别在于 TPSI 协议需要在不泄露基数的条件下进行安全门限测试, 在安全门限测试的过程中往往需要大量昂贵的公钥计算和通信上的多轮交互, 这导致 TPSI 协议效率低下. 而多方场景下的构造要比两方场景下更加困难, 多方 TPSI 协议的实现仍存在许多挑战问题, 例如在多方门限测试过程中的计算上需要更多的公钥操作, 通信上的交互也更为复杂. 因此, 如何在不泄露交集基数的情况下实现一个有效的多方 TPSI 协议是一个困难的问题. 现有经典的多方 TPSI 协议主要聚焦于理论上的研究, 如文献 [14] 设计的协议进行门限测试主要依赖安全汉克尔矩阵奇异性检测算法, 他们的方案基于门限全同态加密, 要实现门限测试需要昂贵的计算开销. 文献 [15] 使用不经意多项式阶数测试进行门限测试, 该算法利

用门限同态加密下, 计算开销仍然是昂贵的. 同时, 他们设计的方案都使用了大量同态下的多项式插值运算, 这些运算的计算开销较大, 无法有效实现. 即使是半诚实模型下的多方 TPSI 协议, 目前也没有有效的实现.

本文构造了两个多方 TPSI 协议, 与现有的 TPSI 协议相比, 本文设计的协议具有以下优势.

(1) 提出一种新的布隆过滤器构造方法, 将弹性秘密共享生成的份额与参与方的集合元素相对应, 通过验证布隆过滤器是否能重构秘密来判断各方交集是否达到门限值, 从而有效防止交集基数的泄露.

(2) 设计了一种轻量级门限隐私集合交集协议, 避免使用开销较大的公钥算法.

(3) 在第 1 个协议的基础上, 结合不经意传输设计一种防合谋的多方门限隐私集合交集协议, 可以抵抗至多 $N-1$ 个敌手合谋.

(4) 本文首次实现了多方 TPSI 协议. 第 1 个协议在三方场景下, 当设置计算安全参数 $\lambda=128$, 集合大小为 2^{14}, 门限值为 $0.8n$ 时, 协议在线阶段的时间成本为 191 s, 第 2 个协议在相同条件下的在线阶段时间成本为 194 s, 并在半诚实模型下证明了两个协议的安全性.

本文第 1 节介绍门限隐私集合交集的研究现状. 第 2 节介绍基础知识, 包括弹性秘密共享、布隆过滤器和不经意传输等. 第 3 节介绍所构建的轻量级多方 TPSI 协议. 第 4 节介绍在第 1 个协议的基础上, 结合不经意传输设计的能抵抗至多 $N-1$ 个敌手合谋的多方 TPSI 协议. 第 5 节通过方案对比和具体实验验证所提协议的有效性. 最后总结全文.

1 相关工作

Freedman 等人[13]使用同态加密和平衡哈希设计了一种具有隐私保护的私有交集基数协议. 他们开创性地提出了 PSI 协议的变体, 允许参与方了解交集大小是否大于门限值, 当交集的基数大于或等于门限值时, 双方可以获得交集. 但是他们并没有设计具体的协议. Zhao 等人[16]提出了一种门限秘密传输方案, 该方案可用于实现门限隐私集合交集协议, 但会泄露交集大小. Ghosh 等人[17]提出了一种不经意线性函数求值的新工具, 并利用代数方法构造了一种 TPSI 协议. Hallgren 等人[12]首次将 TPSI 应用到具有隐私保护的共享拼车上. 他们基于起点、终点和轨迹设计了一种新的 TPSI 算法, 可以在不泄露自己路线的前提下了解到他们可以共享的乘车段.

但是他们的协议通信复杂度较高, 达到了 $O(n^2)$. Ghosh 等人[18]对 TPSI 的通信复杂度进行了研究, 建立了第一上下界, 认为每一个协议都必须有 $O(t)$ 的通信复杂度. 他们利用加性同态加密, 设计了一种通信复杂度为 $O(t^2)$ 的 TPSI 协议, 节省了大量通信成本. 在此之前, 所有的 TPSI 协议通信复杂度均为 $O(n)$. 他们虽然展示了如何扩展至多方设置, 但是他们仍然未能实现多方设置下的 TPSI, 此外, 他们使用了同态加密等公钥算法, 计算开销较大. Pinkas 等人[19]提出了一种新的基于电路的 PSI 协议, 其构造是一个基于二维布谷鸟哈希的变体, 可以将门的数量从 $O(n\log n)$ 减少到 $\omega(n)$. 之后 Pinkas 等人描述了可以扩展他们的 PSI-CAT 电路来解决一个 TPSI 问题, 只有当交集大小大于门限值时密钥才会被释放, 但其整体复杂度至少为 $\omega(n)$.

Zhao 等人[20]设计了两种两方的 TPSI 协议, 一种是交集大于门限值时输出交集, 另一种是交集小于门限值时输出交集. 他们利用加性同态加密与不经意的多项式计算来实现这两种 TPSI, 不会泄漏交集大小. 此外, 他们还设计了一种可外包的 TPSI 协议, 避免两个陌生参与方直接运行高度交互的 TPSI. Badrinarayanan 等人[14]对文献 [18] 进行了扩展并设计了两个多方 TPSI 协议, 第 1 个协议中, 如果各参与方的集合大小与交集大小的差异不超过 t, 则各方得到交集. 第 2 个协议中, 如果各参与方的集合并集的大小与交集大小的差异不超过 t, 则各方得到交集. 他们重点研究了协议的通信复杂度, 并表明任何协议的通信复杂度大小都必须有 $\Omega(nt)$, 但是他们使用了开销较大的全同态加密算法. Branco 等人[15]基于不经意多项式阶数测试设计了一种 TPSI 协议, 其通信复杂度为 $O\left(Nt^2\right)$, 他们的协议仍未能有效实现.

2 基础知识

本文设计的协议主要基于弹性秘密共享、布隆过滤器和不经意传输, 下面介绍相关概念和基础知识.

2.1 弹性秘密共享

秘密共享的思想最早由 Shamir[21]和 Blakley 等人[22]提出, 此后有研究人员扩展出了许多不同的秘密共享方案, 如门限秘密共享, 理性秘密共享, 弹性秘密共享等[23−32].

一个 (t,n) 门限秘密共享方案, 指的是分发者将秘密 s 拆分为 n 个相互独立的子份额, 分别发送给 n 个参与方. 当参与方在执行重构方案时, 当且仅当参与方提供子份额的数量不少于 t 个且子份额完全正确时, 秘密 s 可以被正确的重构. 但是当 t 个子份额中的任意一个有错误时 (可能由于自然损害或者敌手攻击等原因), 该秘密共享方案就无法成功重构秘密. 针对问题, 一部分研究者提出了弹性秘密共享 (robust secret sharing, *RSS*) 的概念. 在一个 (t,n) 弹性秘密共享中, 当参与方重构秘密 s 时, 只要参与方提供的正确的子份额不少于 t, 即使其中一些子份额由于自然损害或者敌手攻击等原因出现了错误, 秘密 s 仍能被正确重构. 本文用里德-所罗门解码算法[33]实例化 *RSS* 功能, *RSS* 的份额生成与秘密重构算法如图 2.

参数: 份额的数量 n, 门限值 t.

份额生成算法: 输入秘密 s, 随机选择 n 个互不相等的索引 $\{ind_1,\cdots,ind_n\}$, 选择一个 $t-1$ 次秘密多项式 $f(\cdot)$ 满足 $f(0)=s$. 计算秘密份额: $s_1=f(ind_1),\cdots,s_n=f(ind_n)$. 输出秘密份额的集合 $S=\{s_1,\cdots,s_n\}$.

秘密重构算法: 输入一个份额集合 $S'=\{s_1',\cdots,s_n'\}$. 通过里德-所罗门解码算法计算出一个多项式 $f'(\cdot)$ 满足 $f'(ind_i)=s_i'$. 如果 S' 中的正确份额(即 S 中的份额)的数量大于 t, 则: $f'(0)=f(0)=s$.

图 2 *RSS* 的份额生成与秘密重构算法

2.2 布隆过滤器与混淆布隆过滤器

布隆过滤器 (Bloom filter, *BF*) 是一种高效的数据结构 (图 3), 有非常广泛的应用[34,35], 其大小只与映射到布隆过滤器的元素数量有关, 与元素的大小无关. 布隆过滤器具有计算复杂度低、空间利用率高和查询效率高等优点.

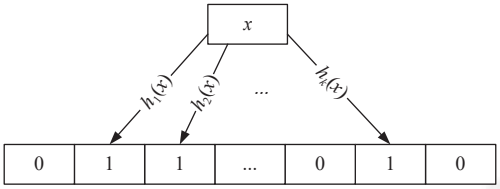

图 3 布隆过滤器构造示意图

布隆过滤器可以看作将元素编码在一个长度为 m 的数组中, 包括一组算法 (生成、构造、验证), 如下所示.

生成算法 (m,k): 输入 m,k, 其中 m 为数组大小, k 为哈希函数的个数. 算法输出 k 个分布均匀且相互独立的哈希函数 $H=\{h_1,\ldots,h_k\}$, 一个长度为 m 的全零布隆过滤器 *BF*.

构造算法 (x,H,BF): 输入 k 个哈希函数、布隆过滤器 *BF* 以及一个元素 x. $BF[h_i(x)]=1$, $i\in[k]$. 把元素 x 映射到 *BF* 中的 k 个位置中的 0 转变为 1. 最终算法输出一个布隆过滤器 *BF*.

验证算法 (x',H,BF): 输入 k 个哈希函数、布隆过滤器 *BF* 以及一个待验证元素 x'. 对于每一个 $i\in[k]$, 算法验证等式 $BF[h_i(x')]=1$ 是否成立, 如果所有等式成立, 则 $x'=x$, 否则 $x'\neq x$.

但是布隆过滤器具有假阳性, 即当元素 y 不在这个集合中, 但是 $BF[h_i(y)]$ 的值依然为 1. 该概率为 $p=1-\left(1-\frac{1}{m}\right)^{k\lambda}$, 其上界为 $\varepsilon=p^k\left(1+\mathrm{O}\left(\frac{k}{p}\right)\sqrt{\frac{\ln m-k\ln p}{m}}\right)$. 除此之外, 普通的布隆过滤器还存在泄露问题, 交集布隆过滤器中存有非交集元素的信息. 针对这个问题, Dong 等人[36]提出了混淆布隆过滤器 (garbled Bloom filter, *GBF*) 的概念 (图 4). 与布隆过滤器不同的是, 当插入一个元素 x 到 *GBF* 中时, 哈希函数映射到的位置不再是 1, 而是一个随机数, 其映射到的所有位置的值异或后的结果为 x, 即 $x=\oplus_{i=1}^{k}GBF[h_i(x)]$.

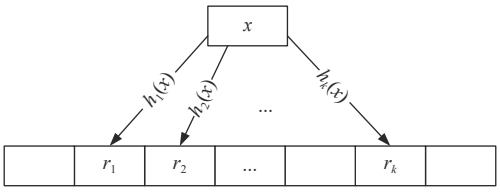

图 4 混淆布隆过滤器构造示意图

2.3 不经意传输

不经意传输 (oblivious transfer, OT)[37−40]是指发送方持有两个消息 m_0, m_1, 接收方发送一个消息比特 $b \in \{0,1\}$给 OT 函数, 如果 $b=0$, 则 OT 函数发送消息 m_0 给接收方, 如果 $b=1$, 则发送消息 m_1给接收方. 在这个过程中, 发送方不知道接收方得到了哪个消息, 接收方不知道发送方的另一个消息 m_{1-b}. OT 理想函数模型如图 5 所示.

参数: 消息的长度为 L.

输入: 接收方输入一个选择比特 $b \in \{0,1\}$, 发送方输入一对消息 $m_0, m_1 \leftarrow \{0,1\}^L$.

输出: 发送 m_b 给接收方.

图 5 不经意传输协议理想函数

2.4 安全模型

本文在半诚实模型下证明了协议的安全性[41,42], 在半诚实模型中, 协议的参与方完全遵守协议的执行规范, 但可以在协议的执行过程中根据协议的输出信息推断其他参与方的信息. 如果协议中的一方能够计算出的任何东西只能从协议的输入和输出中得到, 那么该协议就是安全的. 要求协议执行中的一方的视图仅给定其输入和输出即可模拟. 对于腐败方集合 $C=\{i_1,\ldots,i_q\} \subseteq [N] \overset{\text{def}}{=} \{1,\ldots,N\}$, 令 $f_C(X_1,\ldots,X_N)$ 为 $f_{i_1}(X_1,\ldots,X_N),\ldots,f_{i_q}(X_1,\ldots,X_N)$. 对于 $C=\{i_1,\ldots,i_q\}$的视图为 $view_C^{\pi}(\bar{X})=\left(C, view_{i_1}^{\pi}(\bar{X}),\ldots,view_{i_q}^{\pi}(\bar{X})\right)$, 其中 $\bar{X}=(X_1,\cdots,X_N)$ 为参与方的输入.

定义 1. 令 $f:(\{0,1\}^*)^N \to (\{0,1\}^*)^N$ 是一个确定性函数, 令 π 为安全计算 f 的多方协议, 如果存在概率多项式时间算法 S, 对于任意的 $C \subseteq [N]$满足:

$$\left\{S\left(C,\left(X_{i_1},\ldots,X_{i_q}\right),f_C(\bar{X})\right)\right\}_{\bar{X}\in(\{0,1\}^*)^N} \overset{c}{\equiv} view_C^{\pi}(\bar{X})_{\bar{X}\in(\{0,1\}^*)^N},$$

则协议 π 在半诚实敌手存在下是安全的.

3 一种轻量级多方 TPSI 协议

本节提出了一种轻量级的多方 TPSI 协议. 现有 N 个参与方 $P_1,\ldots,P_N$, 每个参与方都有自己的集合 $X_i=\{x_1^i,x_2^i,\ldots,x_n^i\}$, 集合中的每个元素长为 λ 比特, 设置混淆布隆过滤器与布隆过滤器的长度为 m, 门限值为 t. 所有参与方通过投币协议选择 k 个哈希函数 $h_1,\ldots,h_k:\{0,1\}^* \to [m]$ 与 N 个随机种子 $seed_1,\ldots,seed_N \in \{0,1\}^{\lambda}$. 令 $\oplus$ 表示异或, $\|$表示连接两个字符串. TPSI 协议分为离线阶段与在线阶段两个阶段. 协议的符号说明见后文表 1, 流程如后文图 6.

3.1 协议的构造

在离线阶段, P_1 根据自己的集合元素构造混淆布隆过滤器, P_1 不再把自己的元素放在混淆布隆过滤器中, 而是将弹性秘密共享生成的秘密份额分别放在自己集合元素映射到混淆布隆过滤器的位置, 见算法 1. P_1 构造好自己的混淆布隆过滤器并将其拆分成 $N-1$ 个子混淆布隆过滤器, 分别发送给 $N-1$ 个参与方 $P_2,\ldots,P_N$.

表 1 多方 TPSI 协议符号说明表

符号	说明	符号	说明
N	参与方的数量	GBF_i	P_i 的混淆布隆过滤器
n	集合的大小	GBF_i^j	P_i 发给 P_j 的混淆布隆过滤器
P_i	第 i 个参与方	BF_i	P_i 的布隆过滤器
X_i	P_i 的集合	$SGBF_i$	P_i 用 BF_i 对 GBF_i 转换产生的混淆布隆过滤器
t	门限值	$XGBF_i$	P_j 将所有的 GBF_i^j 异或得到的混淆布隆过滤器
s	秘密	$RGBF_j$	P_1 用 BF_1 和 P_j 执行OT获得的混淆布隆过滤器
ind_l	第 l 个点的索引	$RGBF$	交集混淆布隆过滤器
s_l	第 l 个份额	I	交集

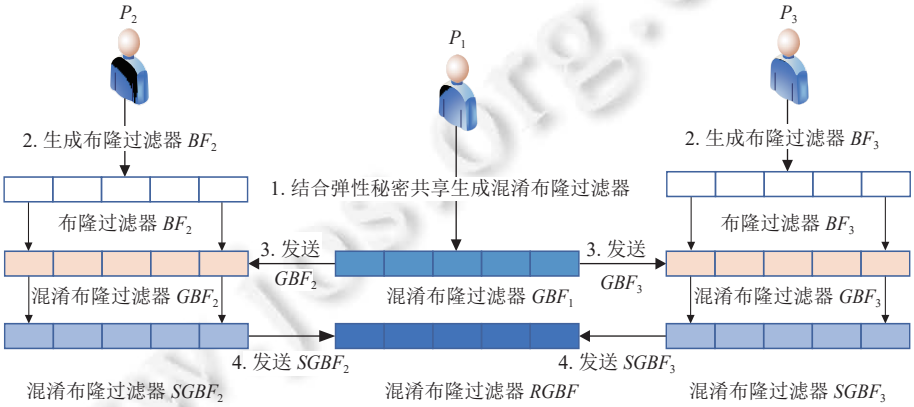

图 6 轻量级多方 TPSI 协议流程图 (以 3 个参与方为例)

算法 1. 结合弹性秘密共享的混淆布隆过滤器生成算法.

输入: 参与方 P_1 的集合 $X_1=\{x_1^1,x_2^1,\ldots,x_n^1\}$, 集合元素个数 n, 集合元素的比特长度 λ, 混淆布隆过滤器长度 m, k 个哈希函数 $h_1,\ldots,h_k$, 弹性秘密共享份额集合 $Share=\{s_1,\ldots,s_n\}$, 索引集合 $Index=\{ind_1,\ldots,ind_n\}$;

输出: 混淆布隆过滤器 GBF_1.

1. 生成一个长度为 m 的数组 GBF_1;
2. **for** $i\leftarrow 1$ to m **do**
3. $GBF_1[i]\leftarrow$ NULL;
4. **end for**
5. **for** $l\leftarrow 1$ to n **do**
6. $emptyBin\leftarrow -1$, $lastShare\leftarrow s_l||ind_l$;
7. **for** $i\leftarrow 1$ to k **do**
8. $j\leftarrow h_i(x_l)$; //得到对集合元素的哈希索引
9. **if** $GBF_1[j]==$ NULL **then**
10. **if** $emptyBin==-1$ **then**
11. $emptyBin\leftarrow j$;
12. **else**
13. $GBF_1[j]\xleftarrow{r}\{0,1\}^{2\lambda}$;

14. $lastShare \leftarrow lastShare \oplus GBF_1[j]$;
15. **end if**
16. **else**
17. $lastShare \leftarrow lastShare \oplus GBF_1[j]$;
18. **end if**
19. **end for**
20. $GBF_1[emptyBin] \leftarrow lastShare$;
21. **end for**
22. **for** $i \leftarrow 1$ to m **do**
23. **if** $GBF_1[i] ==$ NULL **then**
24. $GBF_1[i] \xleftarrow{r} \{0,1\}^{2\lambda}$; //为没有被使用的$GBF_1$中的位置赋随机值
25. **end if**
26. **end for**
27. **return** GBF_1

在线阶段包括两个部分, 分别为份额收发阶段和交集计算阶段. 其中在份额收发阶段中, 引入了一个异或秘密共享方案来隐藏各个参与方的混淆布隆过滤器, 在交集计算阶段, 利用弹性秘密共享能否重构出秘密判断交集基数是否达到门限值, 见算法 2.

算法 2. 判断交集基数是否达到门限值的算法.

输入: 参与方 P_1 的集合 $X_1 = \left\{x_1^1, x_2^1, \ldots, x_n^1\right\}$, 集合元素个数 n, 集合元素的比特长度 λ, 混淆布隆过滤器长度 m, k 个哈希函数 $h_1, \ldots, h_k$, 混淆布隆过滤器 $RGBF$, 秘密 s, 弹性秘密共享重构函数 RSS;
输出: 是否达到门限值.

1. 生成两个空集合 $Share', Index'$;
2. **for** $l \leftarrow 1$ to n **do**
3. $result \leftarrow \{0\}^{2\lambda}$;
4. **for** $i \leftarrow 1$ to k **do**
5. $j \leftarrow h_i(x_l)$; //得到对集合元素的哈希索引
6. $result \leftarrow result \oplus RGBF[j]$;
7. **end for**
8. $s_l' \| ind_l' \leftarrow result$; //将 $result$ 分为两部分
9. 将 s_l' 添加到集合 $Share'$ 中;
10. 将 ind_l' 添加到集合 $Index'$ 中;
11. **end for**
12. 执行 RSS 的重构算法得到秘密多项式 $f'(x) \leftarrow RSS(Share', Index')$;
13. 将 0 代入秘密多项式得到 $s' \leftarrow f'(0)$;
14. **if** $s' == s$ **then**
15. **return** True;
16. **else**
17. **return** False;
18. **end if**

离线阶段具体协议如下.

(1) 对于 $i\in[1,N]$, 参与方 P_i 计算随机种子 $seed=seed_1\oplus\ldots\oplus seed_N$, 然后根据 $seed$ 生成 $n-t$ 个相同伪元素, 将这些伪元素添加在自己的集合中生成一个新的集合 $X_i'=\left\{x_1^i,\ldots,x_n^i,x_{n+1}^i,\ldots,x_{2n-t}^i\right\}$.

(2) P_1 随机选择一个秘密值 s 与 $2n-t$ 个点 $\{ind_1,\ldots,ind_{2n-t}\}$ 运行一个 RSS 算法, 得到 $2n-t$ 个秘密份额 s_l, $l\in[2n-t]$, 其中 $s_l=f(ind_l)$.

(3) P_1 以自己的集合 X_1' 为输入, 根据 k 个哈希函数 $h_1,\ldots,h_k$ 生成自己的混淆布隆过滤器 GBF_1, 满足 $GBF_1\left[h_1(x_l^1)\right]\oplus\ldots\oplus GBF_1\left[h_k(x_l^1)\right]=s_l\|ind_l$, $l\in[2n-t]$. 接着将 GBF_1 拆分为 $N-1$ 份, 满足 $\oplus_{i=2}^N GBF_i=GBF_1$. P_1 发送 GBF_i 给参与方 P_i, $i\in[2,N]$.

在线阶段具体协议如下.

(1) 份额收发阶段.

① $P_2,\ldots,P_N$ 运行一个异或秘密共享方案, 获取各自的秘密份额 rs_i, 并公开 $RS=\oplus_{i=2}^N rs_i$. 对于 $i\in[2,N]$, P_i 根据自己的集合 X_i' 生成自己的布隆过滤器 BF_i. 每一个参与方 P_i 做以下操作: 对于 $b\in[m]$, 如果 $BF_i[b]=0$, 则生成随机数 R 赋值给 $GBF_i[b]$, 即 $GBF_i[b]=R$. 如果 $BF_i[b]=1$, 则不改变 $GBF_i[b]$ 的值. 然后构造混淆布隆过滤器 $SGBF_i$, 满足 $SGBF_i[b]=GBF_i[b]\oplus rs_i$. P_i 发送 $SGBF_i$ 给 P_1.

② 对于 $b\in[m]$, P_1 计算 $RGBF$, 满足:

$$RGBF[b]=SGBF_2[b]\oplus\ldots\oplus SGBF_N[b]\oplus RS.$$

(2) 交集计算阶段.

① P_1 根据 X_1' 查询 $RGBF$. 对于 $l\in[2n-t]$, 计算:

$$s_l'\|ind_l'=RGBF\left[h_1\left(x_l^1\right)\right]\oplus\ldots\oplus RGBF\left[h_k\left(x_l^1\right)\right].$$

前 λ 位看作是份额, 后 λ 位看作是份额对应的索引, 以 s_l', ind_l' 为输入, 根据 RSS 方案重构秘密多项式 $f'(x)$, 计算 $s'=f'(0)$. 如果 $s'=s$, 则交集数量达到门限值, 继续执行协议; 否则, 判定交集数量没有达到门限值并终止协议.

② P_1 检查 s_l', ind_l' 的有效性, 如果 $s_l'=f'(ind'_l)$, 那么 x_l^1 就在交集中, 将 x_l^1 添加到交集 I 中, 否则该元素不在交集中. 最终 P_1 得到交集 I, 并将 I 发送给其他参与方.

3.2 安全证明

引理 1. 混淆布隆过滤器创建失败的概率至多为 $\hat{p}^k\times\left(1+\mathrm{O}\left(\frac{k}{\hat{p}}\sqrt{\frac{\ln m-2k\ln\hat{p}}{m}}\right)\right)$.

证明: 在向 GBF 插入第 n 个元素时, 某一特定位置被占据的最大概率为: $\hat{p}=1-\left(1-\frac{1}{m}\right)^{k(n-1)}$, 其中 m 为 GBF 的长度, k 为哈希函数的个数. 定义 V 为向 GBF 中插入 n 个元素后, GBF 中已经存储字符串的箱子的数量, 有: $E[V]=m\left(1-\left(1-\frac{1}{m}\right)^{kn}\right)$. 设 $w=E[V]+\sqrt{m(\ln m-2k\ln p)}$, 在插入一个元素时, k 个位置均被占据, 也就是混淆布隆过滤器创建失败的最大概率 $\Pr[U]$ 为:

$$\Pr\{U\}=\Pr\{V\leqslant w\}\times\Pr\{U|V\leqslant w\}+\Pr\{V>w\}\times\Pr\{U|V>w\}\leqslant 1\times\Pr\{U|V=w\}+\Pr\{V>w\}\times 1$$

$$\leqslant\left(\frac{E[V]+\sqrt{m(\ln m-2k\ln\hat{p})}}{m}\right)^k+2\mathrm{e}^{\left(-\frac{m(\ln m-2k\ln\hat{p})}{2m}\right)}=\left(\hat{p}+\sqrt{\frac{\ln m-2k\ln\hat{p}}{m}}\right)^k+\frac{2\hat{p}^k}{\sqrt{m}}\leqslant\sum_{i=0}^k\hat{p}^{k-i}\left(k\sqrt{\frac{\ln m-2k\ln\hat{p}}{m}}\right)^i$$

$$+\frac{2\hat{p}^k}{\sqrt{m}}\leqslant\hat{p}^k\times\left(\left(\sum_{i=0}^k\frac{k}{\hat{p}}\sqrt{\frac{\ln m-2k\ln\hat{p}}{m}}\right)^i+\frac{2}{\sqrt{m}}\right)=\hat{p}^k\times\left(\frac{1-\left(\frac{k}{\hat{p}}\sqrt{\frac{\ln m-2k\ln\hat{p}}{m}}\right)^{k+1}}{1-\frac{k}{\hat{p}}\sqrt{\frac{\ln m-2k\ln\hat{p}}{m}}}+\frac{2}{\sqrt{m}}\right)$$

$$\leqslant\hat{p}^k\times\left(\frac{1}{1-\frac{k}{\hat{p}}\sqrt{\frac{\ln m-2k\ln\hat{p}}{m}}}+\frac{2}{\sqrt{m}}\right)=\hat{p}^k\times\left(1+\frac{\frac{k}{\hat{p}}\sqrt{\frac{\ln m-2k\ln\hat{p}}{m}}}{1-\frac{k}{\hat{p}}\sqrt{\frac{\ln m-2k\ln\hat{p}}{m}}}+\frac{2}{\sqrt{m}}\right)=\hat{p}^k\times\left(1+\mathrm{O}\left(\frac{k}{\hat{p}}\sqrt{\frac{\ln m-2k\ln\hat{p}}{m}}\right)\right).$$

定理 1. 本文第 3.1 节的协议在半诚实模型下是安全的.

证明: 首先证明该协议的正确性, 即执行协议能得到正确的结果, 然后证明该协议在半诚实环境下的安全性.

● 正确性. 如果 $P_1,\ldots,P_N$ 的交集基数小于 t, 那么该协议输出终止. 如果 $P_1,\ldots,P_N$ 的交集基数大于等于 t, 那么 P_1 得到的可以重构的正确份额的数量也大于等于 t.

首先分析 RSS 协议重构秘密的正确性. 如第 2.1 节所言, 在一个 (t,n) *RSS* 方案中, 如果正确份额的数量大于 t, 秘密 s 可以被恢复, 即秘密多项式 $f(x)$ 可以被正确重构[33]. 为了保证 *RSS* 能在任意门限值下成功重构秘密, 本文为各参与方添加相同的伪元素, 其数量与门限值 t 和各方集合大小 n 相关. 对于一个 (t,n)TPSI 协议, 为了使 *RSS* 能够正确重构秘密, 需要保证 $|X_1\cap\ldots\cap X_N|\geqslant t+(n+d-t)/2$. 即正确份额数量$\geqslant t+(n+d-t)/2$. d 表示冗余码的数量, 添加冗余码后正确份额数量实际为 $t+d$, 根据上述关系有 $t+d=t+(n+d-t)/2$, 则有 $d\geqslant n-t$, 所以在协议的离线阶段各参与方通过相同的随机种子 *seed* 生成 $n-t$ 个相同的伪元素.

然后分析 *BF* 与 *GBF* 出现假阳性的概率, 证明在给定参数下混淆布隆过滤器能够成功创建并且没有出现假阳性时协议的正确性可以保证. 根据引理 1, 在选定合适参数下混淆布隆过滤器创建失败的概率是可忽略的, 同时, 混淆布隆过滤器出现假阳性的概率也是可忽略的.

接着证明当交集大小达到门限值 t 时, P_1 可以得到足够数量的正确的份额来重构秘密 s. 当一个元素 x_l^1 是交集中的元素时, P_1 通过元素 x_l^1 查询混淆布隆过滤器 *RGBF* 得到:

$$s_l'\|ind_l'=RGBF\left[h_1(x_l^1)\right]\oplus\ldots\oplus RGBF\left[h_k(x_l^1)\right].$$

根据混淆布隆过滤器的同态性质可以得出正确的份额, 推导如下:

$$\begin{aligned}s_l'\|ind_l'&=RGBF\left[h_1(x_l^1)\right]\oplus\ldots\oplus RGBF\left[h_k(x_l^1)\right]=\oplus_{i=1}^{k}\left(\oplus_{j=2}^{N}SGBF_j\left[h_i\left(x_l^1\right)\right]\oplus RS\right)\\&=\oplus_{i=1}^{k}\left(\left(\oplus_{j=2}^{N}GBF_j\left[h_i\left(x_l^1\right)\right]\oplus rs_j\right)\oplus RS\right)=\oplus_{i=1}^{k}GBF_1\left[h_i\left(x_l^1\right)\right]\\&=GBF_1\left[h_1\left(x_l^1\right)\right]\oplus\ldots\oplus GBF_1\left[h_k\left(x_l^1\right)\right]=s_l\|ind_l.\end{aligned}$$

因此, 当交集数量达到门限值 t 时, 可以获得足够多的正确份额重构秘密 s. 综上所述, 协议可以保证正确性.

● 安全性. 首先证明协议在单个腐败方存在下的安全性, 为每个参与方构建了独立的模拟器, 模拟器 S_1 模拟 P_1 的视图, S_i 模拟 $P_i(i\in[2,N])$ 的视图.

首先假设 P_1 为腐败方, 构造模拟器 S_1 模拟 P_1 的视图, 模拟器 S_1 收到 P_1 的集合输入 X_1 和安全参数 λ 以及交集 I.

(1) S_1 选择一个均匀分布的随机值 s 和 $2n-t$ 个索引 $\{ind_1,\ldots,ind_{2n-t}\}$.

(2) S_1 计算 $\{s_1,\ldots,s_{2n-t}\}\leftarrow RSS(s,\{ind_1,\ldots,ind_{2n-t}\})$, 使用上述产生的 s 和 $\{ind_1,\ldots,ind_{2n-t}\}$.

(3) S_1 计算混淆布隆过滤器 $GBF_1\leftarrow(X_1',\{s_1,\ldots,s_{2n-t}\},\{ind_1,\ldots,ind_{2n-t}\})$, 将其拆分成 $N-1$ 份: $\oplus_{i=2}^{N}GBF_i=GBF_1$.

(4) S_1 根据 GBF_i 和交集 I 独立随机采样混淆布隆过滤器 $SGBF_i$.

(5) 最终, S_1 输出 $(X_1,I,SGBF_i)$, $SGBF_i$ 模拟的是协议中 P_i 发送给 P_1 的消息.

S_1 无法像诚实方 P_i 一样计算混淆布隆过滤器 $SGBF_i$, 因为 S_1 没有诚实方 P_i 的输入集合 X_i. 然而, 由于混淆布隆过滤器的混淆性质, P_1 无法区分模拟器随机生成的 $SGBF_i$ 和真实协议执行过程中生成的 $SGBF_i$ (因为真实协议执行过程中生成的 $SGBF_i$ 在 P_1 的视角看来也为随机值), 二者在计算上是不可区分的:

$$\{SGBF_i\}\overset{c}{\equiv}\{F(GBF_i,X_i)\}(i\in[2,N]),$$

其中, F 为真实协议执行中根据输入执行的算法. 所以, 腐败方 P_1 的视图在理想世界和现实世界是不可区分的, 即:

$$\{S_1(X_1,\lambda,I)\}\overset{c}{\equiv}\{view_1^{\pi}(X_1,\ldots,X_N)\}.$$

然后假设 $P_i(i\in[2,N])$ 为腐败方, 构造模拟器 S_i 模拟 P_i 的视图, 模拟器 S_i 收到 P_i 的集合输入 X_i 和安全参数 λ:

(1) S_i 选择一个均匀分布的随机值 s 和 $2n-t$ 个索引 $\{ind_1,\ldots,ind_{2n-t}\}$.

(2) S_i 计算 $\{s_1,\ldots,s_{2n-t}\}\leftarrow RSS(s,\{ind_1,\ldots,ind_{2n-t}\})$, 使用上述产生的 s 和 $\{ind_1,\ldots,ind_{2n-t}\}$.

(3) S_i 根据 $(s,\{s_1,\ldots,s_{2n-t}\},\{ind_1,\ldots,ind_{2n-t}\})$随机采样混淆布隆过滤器 GBF_1, 将其拆分成 N–1份: $\oplus_{i=2}^{N}GBF_i$=GBF_1.

(4) 最终, S_i 输出 (X_i,GBF_i), GBF_i 模拟的是协议中 P_1 发送给 P_i 的消息.

S_i无法像诚实方P_1一样计算混淆布隆过滤器GBF_i, 因为S_i没有诚实方P_1的输入. 由于混淆布隆过滤器的混淆性质, P_i无法区分模拟器随机生成的GBF_i和真实协议执行过程中生成的GBF_i (因为真实协议执行过程中生成的GBF_i在P_i的视角看来也为随机值), 二者在计算上是不可区分的:

$$\{GBF_i\}\overset{c}{\equiv}\{F(X_1,s,\{s_1,\ldots,s_{2n-t}\},\{ind_1,\ldots,ind_{2n-t}\})\},$$

其中, F为真实协议执行中根据输入执行的算法. 所以, 腐败方P_i的视图在理想世界和现实世界是不可区分的, 即:

$$\{S_i(X_i,\lambda)\}\overset{c}{\equiv}\{view_i^{\pi}(X_1,\ldots,X_N)\}.$$

下面证明协议存在不超过$N-2$个腐败方合谋情况下的安全性:

令C表示腐败方的集合: $C=\{P_{i_1},\ldots,P_{i_q}\}\subseteq\{P_1,\ldots,P_N\}$. 为了证明协议可以保护诚实方的非交集信息的安全性, 根据执行协议的顺序通过模拟器S_C模拟C的视图, 考虑以下两种情况:

(1) C中不包含参与方P_1时, C的模拟视图为:

$$\{S_C(X_{i_1},\ldots,X_{i_q})\}.$$

(2) C中包含参与方P_1时, C的模拟视图为:

$$\{S_C(X_{i_1},\ldots,X_{i_q},I)\}.$$

情况 1. C中不包含参与方P_1. 在构造GBF_i时, P_1选取的随机数都独立于自己的隐私信息, 所以当腐败方接收到GBF_i后, 不能获取任何关于P_1的信息, 真实协议产生GBF_i的值与随机值对腐败方来说是不可区分的. 而除P_1外的诚实方在协议的执行过程中不与腐败方C进行交互, 所以腐败方C得不到其他诚实参与方的隐私信息, 模拟视图与真实视图是不可区分的, 即:

$$\{S_C(X_{i_1},\ldots,X_{i_q})\}\overset{c}{\equiv}\{view_C^{\pi}(X_1,\ldots,X_N)\}.$$

情况 2. C中包含参与方P_1. 诚实方P_i发送$SGBF_i$给P_1后, 将$SGBF_i$添加到腐败方C的视图中. $SGBF_i$是P_i通过对GBF_i与rs_i进行异或得到的, 其中rs_i是随机选择的, 独立于P_i的隐私信息, 又由于秘密共享方案的安全性, 保证了rs_i的安全性, 因此也保证了$SGBF_i$的安全性, 真实协议产生$SGBF_i$的值与随机值对腐败方C来说是不可区分的, 所以腐败方无法得到诚实方的隐私信息, 模拟视图与真实视图是不可区分的, 即:

$$\{S_C(X_{i_1},\ldots,X_{i_q},I)\}\overset{c}{\equiv}\{view_C^{\pi}(X_1,\ldots,X_N)\}.$$

4 抗合谋攻击的多方 TPSI 协议

为了抵抗至多$N-1$个敌手合谋, 本节在第 3 节的基础上提出一种基于 OT 的多方 TPSI 协议. P_1构造布隆过滤器, P_2执行第 3.1 节设计的算法 1 构造包含秘密份额的混淆布隆过滤器, 其他参与方构造普通混淆布隆过滤器. 最终P_1通过执行第 3.1 节设计的算法 2 查询混淆布隆过滤器来判断交集基数是否达到门限值. 协议的符号说明见前文表 1, 流程如后文图 7.

4.1 协议的构造

现有N个参与方$P_1,\ldots,P_N$, 每个参与方P_i都有自己的集合$X_i=\{x_1^i,x_2^i,\ldots,x_n^i\}$, 集合中的每个元素长为$\lambda$比特, 设置混淆布隆过滤器与布隆过滤器的长度为$m$, 门限值为$t$, 一个哈希函数$H:\{0,1\}^*\to\{0,1\}^{2\lambda}$. 所有参与方通过一个投币协议选择$k$个用于生成布隆过滤器的哈希函数$h_1,\ldots,h_k:\{0,1\}^*\to[m]$与$N$个随机种子$seed_1,\ldots,seed_N\in\{0,1\}^{\lambda}$. TPSI 协议分为离线阶段与在线阶段两个阶段, 在线阶段包括份额收发阶段和交集计算阶段.

离线阶段协议如下:

(1) 对于$i\in[1,N]$, 参与方P_i计算随机种子$seed=seed_1\oplus\ldots\oplus seed_N$, 然后根据$seed$生成$n-t$个相同伪元素, 将这些伪元素添加在自己的集合中, 生成一个新的集合$X_i'=\{x_1^i,\ldots,x_n^i,x_{n+1}^i,\ldots,x_{2n-t}^i\}$.

(2) P_1根据自己的集合X_1'构造布隆过滤器BF_1.

(3) P_2随机选择一个秘密值s与$2n-t$个索引$\{ind_1,\ldots,ind_{2n-t}\}$运行一个$RSS$份额生成算法, 得到$2n-t$个秘密

份额 s_l, $l \in [2n-t]$, 其中 $s_l = f(ind_l)$. P_2 发送 $H(s)$ 给 P_1.

(4) P_2 根据自己的集合 X_2' 作为输入构造混淆布隆过滤器 GBF_2, 构造满足 $GBF_2\left[h_1(x_l^2)\right] \oplus \ldots \oplus GBF_2\left[h_k(x_l^2)\right] = s_l \| ind_l$. 将 GBF_2 拆分为 $N-1$ 份, 使 $\oplus_{j=2}^{N} GBF_2^j = GBF_2$, 并将 GBF_2^j 发送给 P_j, $j \in [3,N]$.

(5) 对于 $i \in [3,N]$, 参与方 P_i 根据自己的集合 X_i' 生成混淆布隆过滤器 GBF_i, 构造满足 $GBF_i\left[h_1\left[x_l^i\right]\right] \oplus \ldots \oplus GBF_i\left[h_k\left[x_l^i\right]\right] = H\left(x_l^i\right)$, 将 GBF_i 拆分为 $N-1$ 份使得 $\oplus_{j=2}^{N} GBF_i^j = GBF_i$, 并将 GBF_i^j 发送给 P_j ($j \in [2,N], j \neq i$).

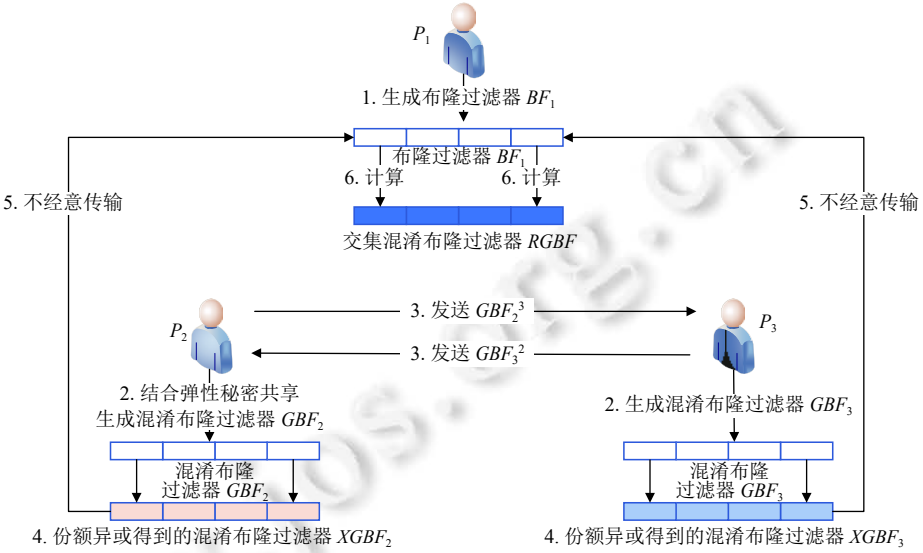

图 7 抗合谋攻击的多方 TPSI 协议流程图 (以 3 个参与方为例)

在线阶段协议如下.

(1) 份额收发阶段

① 对于 $j \in [2,N]$, P_j 在从其他所有参与方接收到混淆布隆过滤器后计算: $XGBF_j = \oplus_{i=2}^{N} GBF_i^j$.

② P_1 作为接受方分别与 $P_{j \in [2,N]}$ 执行 OT 扩展协议, P_1 使用 BF_1 作为输入, P_j 使用 $XGBF_j$ 作为输入. 对于 $b \in [m]$, 如果 $BF_1[b] = 0$, 则 $RGBF_j[b] = r$, 其中 r 为随机数; 如果 $BF_1[b] = 1$, 则 $RGBF_j[b] = XGBF_j[b]$. 最终 P_1 得到 $RGBF_j$.

③ P_1 计算 $RGBF = \oplus_{j=2}^{N} RGBF_j$.

(2) 交集计算阶段

① 对于 $l \in [2n-t]$, 参与方数量 N 为奇数时, P_1 计算 $s_l' \| ind_l' = RGBF\left[h_1\left(x_l^1\right)\right] \oplus \ldots \oplus RGBF\left[h_k\left(x_l^1\right)\right] \oplus H\left(x_l^1\right)$, 当参与方数量 N 为偶数时, P_1 计算 $s_l' \| ind_l' = RGBF\left[h_1\left(x_l^1\right)\right] \oplus \ldots \oplus RGBF\left[h_k\left(x_l^1\right)\right]$. 前 λ 位看作是份额, 后 λ 位看作是份额对应的索引, 并以 s_l', ind_l' 为输入执行 *RSS* 重构算法, 得到多项式 $f'(x)$, 计算 $s' = f'(0)$. 检查 $H(s') \stackrel{?}{=} H(s)$, 如果等式成立, 说明各参与方的交集数量达到门限值, 继续执行协议; 否则终止协议.

② 对于 $l \in [2n-t]$, P_1 检查 s_l' 是否是有效份额, 如果 $s_l' = f'(ind_l')$, 则 x_l^1 是交集中的元素, 将 x_l^1 添加到交集 I 中. 最终 P_1 得到交集 I, 并将 I 发送给其他参与方.

4.2 安全证明

定理 2. 本文第 4.1 节的协议在半诚实模型中是安全的, 可以抵抗至多 $N-1$ 个敌手合谋.

证明: 首先证明该协议的正确性, 即执行协议能得到正确的结果, 然后证明该协议在半诚实环境下的安全性.

● 正确性. 对 *RSS* 和布隆过滤器的正确性证明与第 3.2 节相同. 下面证明当交集大小达到门限值 t 时, P_1 可以得到足够数量的正确的份额来重构秘密 s. 当元素 x_l^1 是交集中的元素时, P_1 通过 x_l^1 查询混淆布隆过滤器 $RGBF$ 得

到 $s'_l||ind'_l = RGBF_1\left[h_1(x_l^1)\right]\oplus\ldots\oplus RGBF_1\left[h_k(x_l^1)\right]$ 或 $s'_l||ind'_l = RGBF_1\left[h_1(x_l^1)\right]\oplus\ldots\oplus RGBF_1\left[h_k(x_l^1)\right]\oplus H\left(x_l^1\right)$ (取决于参与方的数量 N 为奇数还是偶数) 根据混淆布隆过滤器的同态性质可以得出正确的份额, 推导如下:

(1) 当参与方的数量 N 为偶数时:

$$s'_l||ind'_l = RGBF\left[h_1(x_l^1)\right]\oplus\ldots\oplus RGBF\left[h_k(x_l^1)\right] = \oplus_{i=1}^{k}\left(\oplus_{j=2}^{N}RGBF_j\left[h_i\left(x_l^1\right)\right]\right) = \oplus_{i=1}^{k}\left(\oplus_{j=2}^{N}XGBF_j\left[h_i\left(x_l^1\right)\right]\right)$$
$$= \oplus_{i=1}^{k}\left(\oplus_{j=2}^{N}\left(\oplus_{a=2}^{N}GBF_a^j\left[h_i\left(x_l^1\right)\right]\right)\right) = \oplus_{i=1}^{k}\left(\oplus_{j=2}^{N}GBF_i\left[h_i\left(x_l^1\right)\right]\right) = (s_l||ind_l)\oplus\left(\oplus_{j=3}^{N}H\left(x_l^1\right)\right) = s_l||ind_l\ .$$

(2) 当参与方的数量 N 为奇数时:

$$s'_l||ind'_l = RGBF\left[h_1(x_l^1)\right]\oplus\ldots\oplus RGBF\left[h_k(x_l^1)\right]\oplus H\left(x_l^1\right) = \oplus_{i=1}^{k}\left(\oplus_{j=2}^{N}RGBF_j\left[h_i\left(x_l^1\right)\right]\right)\oplus H\left(x_l^1\right)$$
$$= \oplus_{i=1}^{k}\left(\oplus_{j=2}^{N}XGBF_j\left[h_i\left(x_l^1\right)\right]\right)\oplus H\left(x_l^1\right) = \oplus_{i=1}^{k}\left(\oplus_{j=2}^{N}\left(\oplus_{a=2}^{N}GBF_a^j\left[h_i\left(x_l^1\right)\right]\right)\right)\oplus H\left(x_l^1\right)$$
$$= \oplus_{i=1}^{k}\left(\oplus_{j=2}^{N}GBF_i\left[h_i\left(x_l^1\right)\right]\right)\oplus H\left(x_l^1\right) = (s_l||ind_l)\oplus\left(\oplus_{j=3}^{N}H\left(x_l^1\right)\right)\oplus H\left(x_l^1\right) = s_l||ind_l.$$

因此, 当交集数量达到门限值 t 时, 可以获得足够多的正确份额重构秘密出 s.

● 安全性. 首先证明协议在单个腐败方存在下的安全性, 为每个参与方构建了独立的模拟器, 模拟器 S_1 模拟 P_1 的视图, 模拟器 S_2 模拟 P_2 的视图, S_j 模拟 $P_j(j\in[3,N])$ 的视图.

首先假设 P_1 为腐败方, 构造模拟器 S_1 模拟 P_1 的视图, 模拟器 S_1 收到 P_1 的集合输入 X_1 和安全参数 λ 以及交集 I.

(1) S_1 根据 P_1 的输入集合构造 X_1 布隆过滤器 BF_1.

(2) S_1 选择一个均匀分布的随机值 s 和 $2n-t$ 个索引 $\{ind_1,\ldots,ind_{2n-t}\}$, 计算 $\{s_1,\ldots,s_{2n-t}\}\leftarrow RSS(s,\{ind_1,\ldots,ind_{2n-t}\})$ 和 $H(s)$.

(3) S_1 根据 $(s,\{s_1,\ldots,s_{2n-t}\},\{ind_1,\ldots,ind_{2n-t}\})$ 以及交集 I 随机采样混淆布隆过滤器 GBF_2, 随机采样混淆布隆过滤器 GBF_i, 将其拆分成 $N-1$ 份: $\oplus_{j=2}^{N}GBF_i^j = GBF_i$.

(4) S_1 计算 $XGBF_j = \oplus_{i=2}^{N}GBF_i^j$, 然后计算 $RGBF_j\leftarrow\left(BF_1,XGBF_j\right)$.

(5) 最终, S_1 输出 $\left(X_1,I,H(s),RGBF_j\right)$, $H(s)$ 模拟的是 P_2 发送给 P_1 的消息, $RGBF_j$ 模拟的是协议中 P_j 发送给 P_1 的消息.

S_1 无法像诚实方 P_j 一样计算混淆布隆过滤器 $RGBF_j$, 因为 S_i 没有诚实方 P_j 的输入. 基于 OT 协议的安全性, P_1 无法区分模拟器随机生成的 $RGBF_j$ 和真实协议执行过程中通过 OT 获得的 $RGBF_j$, 二者在计算上是不可区分的:

$$\left\{RGBF_j\right\}\overset{c}{\equiv}\left\{F\left(X_j,s,\{s_1,\ldots,s_{2n-t}\},\{ind_1,\ldots,ind_{2n-t}\},BF_1\right)\right\},$$

其中, F 为真实协议执行中根据输入执行的算法. 在真实世界的执行中, P_1 获得 P_2 发送的 $H(s)$, 基于哈希求逆的困难性, P_1 无法区分模拟器随机生成的 $H(s)$ 和真实协议执行过程中产生的 $H(s)$. 所以, 腐败方 P_1 的视图在理想世界和现实世界是不可区分的, 即:

$$\{S_1(X_1,\lambda,I)\}\overset{c}{\equiv}\{view_1^{\pi}(X_1,\ldots,X_N)\}.$$

然后假设 P_2 为腐败方, 构造模拟器 S_2 模拟 P_2 的视图, 模拟器 S_2 收到 P_2 的集合输入 X_2 和安全参数 λ:

(1) S_2 选择一个均匀分布的随机值 s 和 $2n-t$ 个索引 $\{ind_1,\ldots,ind_{2n-t}\}$, 计算 $\{s_1,\ldots,s_{2n-t}\}\leftarrow RSS(s,\{ind_1,\ldots,ind_{2n-t}\})$ 并结合 P_2 的集合输入 X_2 计算混淆布隆过滤器 GBF_2.

(2) S_2 随机均匀采样一组混淆布隆过滤器 GBF_i^2 $(i\in[3,N])$.

(3) 最终, S_2 输出 $\left(X_2,GBF_i^2\right)$, GBF_i^2 模拟的是协议中 P_i 发送给 P_2 的消息.

S_2 无法像诚实方 P_i 一样计算混淆布隆过滤器 GBF_i^2, 因为 S_2 没有诚实方 P_i 的输入. 由于混淆布隆过滤器的混淆性质, P_2 无法区分模拟器随机生成的 GBF_i^2 和真实协议执行过程中生成的 GBF_i^2 (因为真实协议执行过程中生成的 GBF_i^2 在 P_2 的视角看来也为随机值), 二者在计算上是不可区分的:

$$\left\{GBF_i^2\right\}\overset{c}{\equiv}\{F(X_i)\},$$

其中, F 为真实协议执行中根据输入执行的算法. 所以, 腐败方 P_2 的视图在理想世界和现实世界是不可区分的, 即:

$$\{S_2(X_2,\lambda)\}\overset{c}{\equiv}\{view_2^{\pi}(X_1,\ldots,X_N)\}.$$

最后假设 $P_j(j\in[3,N])$ 为腐败方, 构造模拟器 S_j 模拟 P_j 的视图, 模拟器 S_j 收到 P_j 的集合输入 X_j 和安全参数 λ.

(1) S_j 选择一个均匀分布的随机值 s 和 $2n-t$ 个索引 $\{ind_1,\dots,ind_{2n-t}\}$.

(2) S_j 计算 $\{s_1,\dots,s_{2n-t}\}\leftarrow RSS(s,\{ind_1,\dots,ind_{2n-t}\})$, 使用上述产生的 s 和 $\{ind_1,\dots,ind_{2n-t}\}$.

(3) S_j 根据 $(s,\{s_1,\dots,s_{2n-t}\},\{ind_1,\dots,ind_{2n-t}\})$ 随机采样混淆布隆过滤器 GBF_2, 将其拆分成 $N-1$ 份: $\oplus_{j=2}^{N}GBF_2^j=GBF_2$. 根据 P_j 的集合输入 X_j 计算混淆布隆过滤器 GBF_j.

(4) S_j 随机均匀采样一组混淆布隆过滤器 $GBF_i^j(i\in[3,N])$.

(5) 最终, S_j 输出 $\left(X_j,GBF_i^j\right)$, GBF_i^j 模拟的是协议中 P_i 发送给 P_j 的消息.

S_j 无法像诚实方 P_i 一样计算混淆布隆过滤器 GBF_i^j, 因为 S_j 没有诚实方 P_i 的输入. 由于混淆布隆过滤器的混淆性质, P_j 无法区分模拟器随机生成的 GBF_i^j 和真实协议执行过程中生成的 GBF_i^j (因为真实协议执行过程中生成的 GBF_i^j 在 P_j 的视角看来也为随机值), 二者在计算上是不可区分的:

$$\left\{GBF_i^j\right\}\overset{c}{\equiv}\{F(X_i)\},$$

其中, F 为真实协议执行中根据输入执行的算法. 所以, 腐败方 P_j 的视图在理想世界和现实世界是不可区分的, 即:

$$\left\{S_j\left(X_j,\lambda\right)\right\}\overset{c}{\equiv}\left\{view_j^{\pi}(X_1,\dots,X_N)\right\},$$

然后证明协议存在 $N-1$ 个腐败方合谋情况下的安全性:

假设 C 是腐败方的集合: $C=\left\{P_{i_1},\dots,P_{i_q}\right\}\subseteq\{P_1,\dots,P_N\}$. 为了证明该协议能够保护诚实方的非交集信息的隐私, 考虑了最极端的情况, 即只有一个诚实方, 而其他所有的参与方都是腐败的. 通过模拟器 S_C 模拟 C 的视图, 按照协议执行的顺序分为以下两种情况.

(1) C 中不包含参与方 P_1 时, C 的模拟视图为:

$$\left\{S_C\left(X_{i_1},\dots,X_{i_q}\right)\right\}.$$

(2) C 中包含参与方 P_1 时, C 的模拟视图为:

$$\left\{S_C\left(X_{i_1},\dots,X_{i_q},I\right)\right\}.$$

情况 1. C 中不包含参与方 P_1 时. 这种情况下, P_1 只与 C 进行交互, P_1 (扮演接收者的角色) 与 C (扮演发送者的角色) 执行 OT 协议. 基于 OT 协议的安全性, C 无法获取 P_1 的隐私信息. 所以腐败方无法获得诚实方的隐私信息, 模拟视图与真实视图是不可区分的, 即 $\left\{S_C\left(X_{i_1},\dots,X_{i_q}\right)\right\}\overset{c}{\equiv}\left\{view_C^{\pi}(X_1,\dots,X_N)\right\}$.

情况 2. C 中包含参与方 P_1 时. 在这种情况下, 诚实方 P_i 将 GBF_i 的子份额发送给 C, 由于 P_i 保留了一个子份额, 没有发送给其他参与方, 所以 C 合谋也无法获得 P_i 的 GBF_i. P_i (作为发送方) 与腐败方 P_1 (作为接收方) 执行 OT 协议. 基于 OT 协议的安全性, C 无法获取 P_i 的隐私信息. P_i 的 $XGBF_i$ 中对应 P_1 的 BF_1 中为 0 的值, 独立于 C 的视图. 所以腐败方无法获得诚实方的隐私信息, 模拟视图与真实视图是不可区分的, 即 $\left\{S_C\left(X_{i_1},\dots,X_{i_q},I\right)\right\}\overset{c}{\equiv}\left\{view_C^{\pi}(X_1,\dots,X_N)\right\}$.

5 实验与分析

本文的协议使用 C++实现, 并在 Ubuntu 18.04 系统上进行演示实验, 其中 CPU 为 AMD Ryzen 7 4800U@1.80 GHz, 内存为 16 GB. 设置协议计算安全参数为 $\lambda=128$, 统计安全参数 $\sigma=40$, 布隆过滤器哈希函数个数 $k=128$, 布隆过滤器的大小 $m=1.44kn$ [36].

5.1 计算复杂度

本节对协议的计算复杂度进行分析. 对于本文提出的第 1 种轻量级多方 TPSI 协议, 各方生成自己的布隆过滤器与混淆布隆过滤器时, 其计算复杂度为 $O(nk)$. 此外, 在在线阶段时, $P_2,\dots,P_N$ 还需要计算 $SGBF_i$, 其计算复杂度与 $SGBF_i$ 的长度 m 相关, 为 $O(m)$. P_1 在计算 $RGBF$ 时, 计算复杂度与 N 和 m 相关, 为 $O(Nm)$. 在重构秘密多项式时,

RSS 方案的重构阶段计算复杂度为 $O(n\log n)$. P_1 需要检查 n 个份额, 计算复杂度为 $O(n)$. 所以, 该协议计算复杂度为 $O(nk+Nm+n\log n)$. 同理, 第 2 种抗合谋攻击的多方 TPSI 协议的计算复杂度也为 $O(nk+Nm+n\log n)$.

5.2 通信复杂度

本节对协议的通信复杂度进行分析. 对于本文提出的第 1 种轻量级多方 TPSI 协议, 在离线阶段, 参与方 P_1 发送 GBF_i 给各参与方, 其通信复杂度取决于参与方的数量 N 与混淆布隆过滤器的长度 m, 为 $O(Nm\lambda)$. 在在线阶段, 各参与方参与一个异或秘密共享方案, 其通信复杂度与 N 相关, 为 $O(N\lambda)$; 各参与方发送自己的 $SGBF_i$ 给 P_1, 其通信复杂度与 N 和 m 有关, 为 $O(Nm\lambda)$. 所以该协议通信复杂度为 $O(Nm\lambda)$. 第 2 种抗合谋攻击的多方 TPSI 协议在在线阶段参与方相互发送混淆布隆过滤器, 其通信复杂度为 $O(N^2m\lambda)$, 所以抗合谋攻击 TPSI 协议的通信复杂度为 $O(N^2m\lambda)$.

5.3 实验结果与分析

本文协议分别与 Satrajit 等人发表的两方 TPSI 协议 (CRYPTO)[18]和现有的多方 TPSI 协议[14,15]进行比较, 上述协议均使用了开销较大的算法, 未能有效实现. 如表 2 所示, 本文在算法的计算开销上有较大优势, 根据具体的实验数据表明, 该协议优于现有的 TPSI 协议, 具备实际的应用价值.

表 2 与其他文献的效率对比

协议	通信复杂度	计算复杂度	参与方数量	大开销算法	是否实现
文献[18]	$O(t^2\lambda)$	$O(t^2\log t)$	2	同态加密	否
文献[14]	$O(N^2t\lambda)$	—	N	全同态加密	否
文献[15]	$O(Nt^2)$	—	N	同态加密	否
轻量级TPSI	$O(Nm\lambda)$	$O(nk+Nm+n\log n)$	N	无	是
抗合谋TPSI	$O(N^2m\lambda)$	$O(nk+Nm+n\log n)$	N	无	是

首先, 测试本文第 3 节提出的轻量级多方 TPSI 协议, 设置协议的参与方数量为 $N=3$. 对于参与方集合大小 n, 分别设置 6 个级别为 $2^9,2^{10},\ldots,2^{14}$. 对于门限值 t, 分别设置 3 个级别为 30%, 60%, 80%, 即 $t=0.3n$、$t=0.6n$、$t=0.8n$. 当 $t=0.3n$ 时, 该协议的时间成本如表 3, 对于门限值 $t=0.6n$, 本协议时间成本如表 4, 对于门限值 $t=0.8n$, 本协议时间成本如表 5.

表 3 $t=0.3n$ 时协议各阶段的运行时间 (s)

集合大小	离线阶段	份额收发	交集计算	总时间
2^9	0.16	0.02	0.22	0.40
2^{10}	0.48	0.03	0.80	1.31
2^{11}	1.67	0.06	3.11	4.84
2^{12}	6.23	0.16	11.92	18.31
2^{13}	22.28	0.27	47.76	70.31
2^{14}	85.62	0.45	202.34	288.41

表 4 $t=0.6n$ 时协议各阶段的运行时间 (s)

集合大小	离线阶段	份额收发	交集计算	总时间
2^9	0.19	0.02	0.21	0.42
2^{10}	0.69	0.03	0.80	1.52
2^{11}	2.52	0.06	3.06	5.64
2^{12}	9.11	0.09	11.59	20.79
2^{13}	35.05	0.16	46.04	81.25
2^{14}	140.61	0.34	188.09	329.04

然后, 设置协议的门限值为 $t=0.8n$. 对于参与方的数量, 分别设置 5 个级别为 3、4、5、7、10. 对于参与方集合大小 n, 分别设置 6 个级别为: $2^9,2^{10},\ldots,2^{14}$. 该协议时间成本如表 6 所示, 当门限值与集合大小一定时, 协议的时间成本随着参与方数量增长.

测试本文第 4 节提出的抗合谋攻击的多方 TPSI 协议, 设置协议的参与方数量为 $N=3$. 对于参与方集合大小 n, 分别设置 6 个级别为 $2^9,2^{10},\ldots,2^{14}$. 对于门限值 t, 分别设置 3 个级别为 30%, 60%, 80%, 即 $t=0.3n$、$t=0.6n$、$t=0.8n$. 当 $t=0.3n$ 时, 该协议的时间成本如表 7, 对于门限值 $t=0.6n$, 本协议时间成本如表 8, 对于门限值 $t=0.8n$, 本协议时间成本如表 9.

表 5 $t=0.8n$ 时协议各阶段的运行时间 (s)

集合大小	离线阶段	份额收发	交集计算	总时间
2^9	0.21	0.01	0.21	0.43
2^{10}	0.78	0.02	0.83	1.63
2^{11}	2.74	0.04	3.08	5.86
2^{12}	10.25	0.08	11.85	22.18
2^{13}	40.29	0.14	47.11	87.54
2^{14}	161.28	0.28	191.48	353.04

表 6 $t=0.8n$ 时轻量级 TPSI 协议在不同参与方数量下的总运行时间 (s)

级别	2^9	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}
3方	0.43	1.63	5.86	22.18	87.54	353.04
4方	0.59	2.16	8.28	27.85	110.20	429.39
5方	0.67	2.42	9.11	33.17	129.74	524.85
7方	0.82	3.04	11.43	42.83	165.26	675.42
10方	0.98	4.26	13.56	54.64	212.27	877.15

表 7 $t=0.3n$ 时抗合谋攻击 TPSI 协议各阶段的运行时间 (s)

集合大小	离线阶段	份额收发	交集计算	总时间
2^9	0.20	0.02	0.20	0.42
2^{10}	0.50	0.04	0.81	1.35
2^{11}	1.69	0.08	3.12	4.89
2^{12}	6.25	0.18	11.94	18.37
2^{13}	23.08	0.31	47.78	71.17
2^{14}	85.87	0.49	203.12	289.48

表 8 $t=0.6n$ 时抗合谋攻击 TPSI 协议各阶段的运行时间 (s)

集合大小	离线阶段	份额收发	交集计算	总时间
2^9	0.19	0.03	0.21	0.43
2^{10}	0.69	0.05	0.80	1.54
2^{11}	2.54	0.10	3.08	5.72
2^{12}	9.09	0.18	11.82	21.09
2^{13}	35.12	0.37	46.35	81.84
2^{14}	140.68	0.68	188.96	330.32

然后, 设置协议的门限值为 $t=0.8n$. 对于参与方的数量, 分别设置 5 个级别为 3、4、5、7、10. 对于参与方集合大小 n, 分别设置 6 个级别为 $2^9,2^{10},\ldots,2^{14}$. 该协议时间成本如表 10 所示, 当门限值与集合大小一定时, 协议的时间成本随着参与方数量增长.

表 9 $t=0.8n$ 时抗合谋攻击 TPSI 协议各阶段的运行时间 (s)

集合大小	离线阶段	份额收发	交集计算	总时间
2^9	0.21	0.03	0.21	0.45
2^{10}	0.79	0.06	0.84	1.69
2^{11}	2.76	0.11	3.17	6.04
2^{12}	10.26	0.19	11.9	22.35
2^{13}	40.10	0.39	47.41	87.90
2^{14}	161.33	0.73	193.71	355.77

表 10 $t=0.8n$ 时抗合谋攻击 TPSI 协议在不同参与方数量下的总运行时间 (s)

级别	2^9	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}
3方	0.45	1.69	6.04	22.35	87.90	355.77
4方	0.68	2.30	8.36	27.98	110.34	429.58
5方	0.83	2.32	9.31	33.46	130.96	525.93
7方	0.98	3.35	11.73	43.45	167.35	678.68
10方	1.26	4.64	13.92	56.57	215.59	885.36

本文的协议在不同集合大小下的通信量如后文表 11 所示. 从实验数据可以看出, 轻量级 TPSI 协议的通信开销较小, 且随着参与方的数量增加时通信开销的增幅较小. 在元素长度为 128 比特、集合大小为 2^{10}、10 个参与方的情况下, 协议的通信总量小于 90 MB. 抗合谋 TPSI 协议在相同参数下的通信总量约为 135 MB.

本文的协议与相关文献的实验数据对比如后文表 12 所示. 由于本文设计的 TPSI 协议是首个实现的多方 TPSI 协议, 暂时无法找到能对比的多方 TPSI 协议的实验数据, 所以本文与目前能实现的两方 TPSI 协议的实验数据进行对比. 从实验数据可以看出本文的多方 TPSI 协议比目前的两方 TPSI 协议更有效, 本文协议在 10 方场景下, 集合大小为 2^{11} 时总的运行时间大约是文献 [16] 的 1/55, 当集合大小为 2^{12} 时总的运行时间大约是文献 [12] 的 1/100.

6 总 结

本文在半诚实模型下提出并首次实现了两个多方 TPSI 协议, 相较于以往的协议, 本文提出的第 1 个协议可以避免使用开销大的公钥算法, 第 2 个协议通过结合不经意传输能够实现抵抗至多 $N-1$ 个敌手的合谋攻击. 同时所

设计的两个协议都能防止交集基数的泄露. 在本文的最后进行与其他协议的对比, 实验数据表明所设计的两个协议是有效的, 具有较高的理论意义和应用价值.

表 11　协议在不同集合大小下的通信量 (MB)

协议	2^9	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}
轻量级TPSI (3方)	9.61	19.20	38.40	76.81	153.61	307.20
轻量级TPSI (4方)	14.41	28.80	57.61	115.20	230.41	460.81
轻量级TPSI (5方)	19.22	38.41	76.82	153.62	307.28	614.55
轻量级TPSI (7方)	28.82	57.61	115.22	230.43	460.82	921.61
轻量级TPSI (10方)	43.24	86.41	172.83	345.65	691.24	1 382.42
抗合谋TPSI (3方)	16.82	33.61	67.21	134.73	268.82	537.61
抗合谋TPSI (4方)	24.03	48.02	96.02	192.04	384.03	768.02
抗合谋TPSI (5方)	31.24	62.42	124.84	249.65	499.24	998.42
抗合谋TPSI (7方)	45.66	91.24	182.42	364.81	729.62	1 459.22
抗合谋TPSI (10方)	67.29	135.07	270.12	540.28	1 080.52	2 161.11

表 12　与相关文献的实验数据对比 (s)

协议	2^9	2^{10}	2^{11}	2^{12}
文献[12]	13.57	96.78	728.10	5 627
文献[16]	199.57	400.86	778.96	—
轻量级TPSI (10方)	0.98	4.26	13.56	54.64
抗合谋TPSI (10方)	1.26	4.64	13.92	56.57

References:

[1] Zhang E, Liu FH, Lai QQ. Efficient multi-party private set intersection against malicious adversaries. In: Proc. of the 2019 ACM SIGSAC Conf. on Cloud Computing Security Workshop. London: ACM, 2019. 93–104. [doi: 10.1145/3338466.3358927]

[2] Garimella G, Pinkas B, Rosulek M, Trieu N, Yanai A. Oblivious key-value stores and amplification for private set intersection. In: Proc. of the 41st Annual Int'l Cryptology Conf. Springer, 2021. 395–425. [doi: 10.1007/978-3-030-84245-1_14]

[3] Chase M, Miao PH. Private set intersection in the internet setting from lightweight oblivious PRF. In: Proc. of the 40th Annual Int'l Cryptology Conf. Santa Barbara: Springer, 2020. 34–63. [doi: 10.1007/978-3-030-56877-1_2]

[4] Gong LM, Wang DS, Liu MM, Gao QL, Shao LH, Wang MM. PSI computation based on no matching errors. Chinese Journal of Computers, 2020, 43(9): 1769–1790 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2020.01769]

[5] Le PH, Ranellucci S, Gordon SD. Two-party private set intersection with an untrusted third party. In: Proc. of the 2019 ACM SIGSAC Conf. on Computer and Communications Security. London: ACM, 2019. 2403–2420. [doi: 10.1145/3319535.3345661]

[6] Rindal P, Schoppmann P. VOLE-PSI: Fast OPRF and circuit-PSI from vector-OLE. In: Proc. of the 40th Annual Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Zagreb: Springer, 2021. 901–930. [doi: 10.1007/978-3-030-77886-6_31]

[7] Chen ZH, Li SD, Huang Q, Ding Y, Liu YR. Secure computation of two set-relationships with the unencrypted method. Ruan Jian Xue Bao/Journal of Software, 2018, 29(2): 473−482 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5262.htm [doi: 10.13328/j.cnki.jos.005262]

[8] Wang YH, Huang Q, Li HB, Xiao MY, Ma S, Susilo W. Private set intersection with authorization over outsourced encrypted datasets. IEEE Trans. on Information Forensics and Security, 2021, 16: 4050–4062. [doi: 10.1109/TIFS.2021.3101059]

[9] Li SD, Zhou SF, Guo YM, Dou JW. Secure set computing in cloud environment. Ruan Jian Xue Bao/Journal of Software, 2016, 27(6): 1549–1565 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4996.htm [doi: 10.13328/j.cnki.jos.004996]

[10] Dou JW, Liu XH, Wang WL. Privacy preserving two-party rational set computation. Chinese Journal of Computers, 2020, 43(8): 1397–1413 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2020.01397]

[11] Mohassel P, Zhang YP. SecureML: A system for scalable privacy-preserving machine learning. In: Proc. of the 2017 IEEE Symp. on Security and Privacy. San Jose: IEEE, 2017. 19–38. [doi: 10.1109/SP.2017.12]

[12] Hallgren P, Orlandi C, Sabelfeld A. PrivatePool: Privacy-preserving ridesharing. In: Proc. of the 30th IEEE Computer Security Foundations Symp. Santa Barbara: IEEE, 2017. 276–291. [doi: 10.1109/CSF.2017.24]

[13] Freedman MJ, Nissim K, Pinkas B. Efficient private matching and set intersection. In: Proc. of the Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Interlaken: Springer, 2004. 1–19. [doi: 10.1007/978-3-540-24676-3_1]

[14] Badrinarayanan S, Miao PH, Raghuraman S, Rindal P. Multi-party threshold private set intersection with sublinear communication. In: Proc. of the 24th IACR Int'l Conf. on Practice and Theory of Public Key Cryptography. Springer, 2021. 349–379. [doi: 10.1007/978-3-030-75248-4_13]

[15] Branco P, Döttling N, Pu SH. Multiparty cardinality testing for threshold private intersection. In: Proc. of the 24th IACR Int'l Conf. on Practice and Theory of Public Key Cryptography. Springer, 2021. 32–60. [doi: 10.1007/978-3-030-75248-4_2]

[16] Zhao YJ, Chow SSM. Are you the one to share? Secret transfer with access structure. Proc. on Privacy Enhancing Technologies, 2017, 2017(1): 149–169. [doi: 10.1515/popets-2017-0010]

[17] Ghosh S, Nilges T. An algebraic approach to maliciously secure private set intersection. In: Proc. of the 38th Annual Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Darmstadt: Springer, 2019. 154–185. [doi: 10.1007/978-3-030-17659-4_6]

[18] Ghosh S, Simkin M. The communication complexity of threshold private set intersection. In: Proc. of the 39th Annual Int'l Cryptology Conf. Santa Barbara: Springer, 2019. 3–29. [doi: 10.1007/978-3-030-26951-7_1]

[19] Pinkas B, Schneider T, Weinert C, Wieder U. Efficient circuit-based PSI via cuckoo hashing. In: Proc. of the 37th Annual Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Tel Aviv: Springer, 2018. 125–157. [doi: 10.1007/978-3-319-78372-7_5]

[20] Zhao YJ, Chow SSM. Can you find the one for me? In: Proc. of the 2018 Workshop on Privacy in the Electronic Society. Toronto: ACM, 2018. 54–65. [doi: 10.1145/3267323.3268965]

[21] Shamir A. How to share a secret. Communications of the ACM, 1979, 22(11): 612–613. [doi: 10.1145/359168.359176]

[22] Blakley GR. Safeguarding cryptographic keys. In: Proc. of the 1979 Int'l Workshop on Managing Requirements Knowledge. New York: IEEE, 1979. 313–318. [doi: 10.1109/MARK.1979.8817296]

[23] Zhang E, Li M, Yiu SM, Du J, Zhu JZ, Jin GG. Fair hierarchical secret sharing scheme based on smart contract. Information Sciences, 2021, 546: 166–176. [doi: 10.1016/j.ins.2020.07.032]

[24] Manurangsi P, Srinivasan A, Vasudevan PN. Nearly optimal robust secret sharing against rushing adversaries. In: Proc. of the 40th Annual Int'l Cryptology Conf. Santa Barbara: Springer, 2020. 156–185. [doi: 10.1007/978-3-030-56877-1_6]

[25] Kurosawa K, Obana S, Ogata W. *t*-cheater identifiable (*k*, *n*) threshold secret sharing schemes. In: Proc. of the 15th Annual Int'l Cryptology Conf. Santa Barbara: Springer, 1995. 410–423. [doi: 10.1007/3-540-44750-4_33]

[26] Zhang E, Geng K, Jin W, Li YJ, Sun YQ, Li FH. Cloud outsourcing secret sharing scheme against covert adversaries. Journal on Communications, 2017, 38(5): 57–65 (in Chinese with English abstract). [doi: 10.11959/j.issn.1000-436x.2017100]

[27] Zhang E, Cai YQ. A verifiable rational secret sharing scheme based on bilinear pairing. Acta Electronica Sinica, 2012, 40(5): 1050–1054 (in Chinese with English abstract). [doi: 10.3969/j.issn.0372-2112.2012.05.031]

[28] Liu H, Li XH, Tian YL, Luo B, Ma JF, Peng CG. Rational fair secret sharing scheme. Chinese Journal of Computers, 2020, 43(8): 1517–1533 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2020.01517]

[29] Yu J, Chen YK, Hao R, Kong FY, Cheng XG, Pan ZK. Publicly verifiable multi-secret sharing without trusted centers. Chinese Journal of Computers, 2014, 37(5): 1030–1038 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2014.01030]

[30] Zhang YS, Li WJ, Chen L, Bi W, Yang T. Verifiable special threshold secret sharing scheme based on eigenvalue. Journal on Communications, 2018, 39(8): 169–175 (in Chinese with English abstract). [doi: 10.11959/j.issn.1000-436x.2018143]

[31] Peng Q, Tian YL. A secret sharing scheme based on multilinear Diffie-Hellman problem. Acta Electronica Sinica, 2017, 45(1): 200–205 (in Chinese with English abstract). [doi: 10.3969/j.issn.0372-2112.2017.01.027]

[32] Cevallos A, Fehr S, Ostrovsky R, Rabani Y. Unconditionally-secure robust secret sharing with compact shares. In: Proc. of the 31st Annual Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Cambridge: Springer, 2012. 195–208. [doi: 10.1007/978-3-642-29011-4_13]

[33] Gao SH. A new algorithm for decoding Reed-Solomon codes. In: Bhargava VK, Poor HV, Tarokh V, Yoon S, eds. Communications, Information and Network Security. Boston: Springer, 2002. 55–68. [doi: 10.1007/978-1-4757-3789-9_5]

[34] Naor M, Yogev E. Bloom filters in adversarial environments. In: Proc. of the 35th Annual Cryptology Conf. Santa Barbara: Springer, 2015. 565–584. [doi: 10.1007/978-3-662-48000-7_28]

[35] Derler D, Jager T, Slamanig D, Striecks C. Bloom filter encryption and applications to efficient forward-secret 0-RTT key exchange. In: Proc. of the 37th Annual Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Tel Aviv: Springer, 2018. 425–455. [doi: 10.1007/978-3-319-78372-7_14]

[36] Dong CY, Chen LQ, Wen ZK. When private set intersection meets big data: An efficient and scalable protocol. In: Proc. of the 2013 ACM SIGSAC Conf. on Computer & Communications Security. Berlin: ACM, 2013. 789–800. [doi: 10.1145/2508859.2516701]

[37] Asharov G, Lindell Y, Schneider T, Zohner M. More efficient oblivious transfer and extensions for faster secure computation. In: Proc. of the 2013 ACM SIGSAC Conf. on Computer & Communications Security. Berlin: ACM, 2013. 535–548. [doi: 10.1145/2508859.2516738]

[38] Ishai Y, Kilian J, Nissim K, Petrank E. Extending oblivious transfers efficiently. In: Proc. of the 23rd Annual Int'l Cryptology Conf. Berlin: Springer, 2003. 145–161. [doi: 10.1007/978-3-540-45146-4_9]

[39] Döttling N, Garg S, Hajiabadi M, Masny D, Wichs D. Two-round oblivious transfer from CDH or LPN. In: Proc. of the 39th Annual Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Zagreb: Springer, 2020. 768–797. [doi: 10.1007/978-3-030-45724-2_26]

[40] Grilo AB, Lin HJ, Song F, Vaikuntanathan V. Oblivious transfer is in MiniQCrypt. In: Proc. of the 40th Annual Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Zagreb: Springer, 2021. 531–561. [doi: 10.1007/978-3-030-77886-6_18]

[41] Goldreich O. The Foundations of Cryptography—Volume 2: Basic Applications. Cambridge: Cambridge University Press, 2004.

[42] Canetti R. Universally composable security: A new paradigm for cryptographic protocols. In: Proc. of the 42nd IEEE Symp. on Foundations of Computer Science. Newport Beach: IEEE, 2001. 136–145. [doi: 10.1109/SFCS.2001.959888]

附中文参考文献:

[4] 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. [doi: 10.11897/SP.J.1016.2020.01769]

[7] 陈振华, 李顺东, 黄琼, 丁勇, 刘娅茹. 非加密方法安全计算两种集合关系. 软件学报, 2018, 29(2): 473–482. http://www.jos.org.cn/1000-9825/5262.htm [doi: 10.13328/j.cnki.jos.005262]

[9] 李顺东, 周素芳, 郭奕旻, 窦家维, 王道顺. 云环境下集合隐私计算. 软件学报, 2016, 27(6): 1549–1565. http://www.jos.org.cn/1000-9825/4996.htm [doi: 10.13328/j.cnki.jos.004996]

[10] 窦家维, 刘旭红, 王文丽. 有理数域上两方集合的高效保密计算. 计算机学报, 2020, 43(8): 1397–1413. [doi: 10.11897/SP.J.1016.2020.01397]

[26] 张恩, 耿魁, 金伟, 李勇俊, 孙韵清, 李凤华. 抗隐蔽敌手的云外包秘密共享方案. 通信学报, 2017, 38(5): 57–65. [doi: 10.11959/j.issn.1000-436x.2017100]

[27] 张恩, 蔡永泉. 基于双线性对的可验证的理性秘密共享方案. 电子学报, 2012, 40(5): 1050–1054. [doi: 10.3969/j.issn.0372-2112.2012.05.031]

[28] 刘海, 李兴华, 田有亮, 雒彬, 马建峰, 彭长根. 理性公平的秘密共享方案. 计算机学报, 2020, 43(8): 1517–1533. [doi: 10.11897/SP.J.1016.2020.01517]

[29] 于佳, 陈养奎, 郝蓉, 孔凡玉, 程相国, 潘振宽. 无可信中心的可公开验证多秘密共享. 计算机学报, 2014, 37(5): 1030–1038. [doi: 10.3724/SP.J.1016.2014.01030]

[30] 张艳硕, 李文敬, 陈雷, 毕伟, 杨涛. 基于特征值的可验证特殊门限秘密共享方案. 通信学报, 2018, 39(8): 169–175. [doi: 10.11959/j.issn.1000-436x.2018143]

[31] 彭巧, 田有亮. 基于多线性Diffie-Hellman问题的秘密共享方案. 电子学报, 2017, 45(1): 200–205. [doi: 10.3969/j.issn.0372-2112.2017.01.027]

张恩(1974－), 男, 博士, 教授, CCF 高级会员, 主要研究领域为网络安全, 密码学.

杨刃林(1997－), 男, 硕士生, 主要研究领域为网络安全, 密码学.

秦磊勇(1997－), 男, 硕士生, 主要研究领域为网络安全, 密码学.

李功丽(1981－), 女, 博士, 副教授, CCF 专业会员, 主要研究领域为网络安全, 密码学.