基于 Secret Sharing 与同态加密的反洗钱客户姓名隐私模糊查询

author1[a*], author2[a,b], author2[b]
[a] School of Computer Science, Wuhan University, Wuhan, China
[b] Department of Computer Science and Technology, Tsinghua University, Beijing, China
{zhangsan}@XXX.com, {lisi, wangwu}@XXX.edu.cn

March 1, 2023

Abstract

针对反洗钱客户姓名数据在跨境模糊查询过程中存在的数据交互安全性与合规性问题，提出一种基于 Paillier 同态加密的客户姓名模糊查询数据加密方案。分析客户姓名模糊查询业务流程和数据传输特征，结合 Paillier 算法同态特性，进行模糊查询数据的 xxxxxx 以提高数据交互的安全性。通过试验，验证方案的效率和加解密正确率等性能，并进行对比分析。结果表明，方案能有效保障数据完整性和安全可靠传输。

关键词：Secret Sharing; 同态加密; 反洗钱; 模糊匹配;

1 Introduction

近年来，随着金融机构信息技术和商业模型的创新发展，金融产品、业务服务模式的创新和科技化改造极易被不法分子利用，成为洗钱等违法犯罪活动新途径；同时，银行、证券等金融机构数据的流通不仅关系到个体隐私安全，更关系到国家金融安全与信息安全，在此背景下《数据安全法》与《个人信息保护法》的相继出台与实施，切实地保障了个人隐私数据的安全与国家金融信息安全，但是在满足数据合规的基础上，金融数据跨境传输是否合规成为反洗钱全球网络系统数据完整性的一个痛点。

保证满足数据相关法律法规下的反洗钱金融业务数据跨境传输是当前金融行业数据跨境传输面临的一个不可忽视的问题。在银行实际区洗钱场景中细化为两个是体场品问题：反洗钱客户精确匹配、反洗钱客户模糊匹配问题。本文旨在介绍多方安全计算相关协议在反洗钱客户模糊匹配场景下的应用与创新。针对反洗钱客户模糊匹配（查询）场景，基于同态加密的门限秘密分享技术将客户姓名数据的密文切分成多份子秘密进行存储、传输，并能够使用 K 份子秘密还原秘密客户姓名，并利用插值多项式原理在还原过程中的门限技术，实现模糊匹配，这在一定程度上有效解决了数据跨境传输、合规、模糊匹配这三维合一的问题。

（模糊匹配的定义）（隐私模糊匹配的动机）

2 相关理论知识

2.1 基于多项式的 Shamir 秘密分享

秘密分享 (Secret Shaning SS) 是多方安全计算中常用的协议之一。秘密分享的主要思路是将一个密文信息拆分成多份秘密片段，并将每份秘密分发到不同参与方，从而使得不同参与方在握有秘密片段的同时，又无法得知真正的完整秘密数据。例如：在两方秘密分享场景中，密文信息 M 可以被秘密分享为 <MI, M2>，满足 MI+M2=M，其中，参与方 A 持有密文 M1，参与方 B 持有密文 M2。

本文关注于两方的其于多项式的秘密分享。早期的 psS 协议主要基于拉格朝日插值法。Blaklex 在 1979 年提出了一种基于多项式插值的 PSS 协议，该协议基于拉格朗日插值公式构造多项式，并将其分成多份。接着，Shamir 在 1984 年提出了著名的 Shamir 秘密分享协议，该协议使用了拉格朗日插值法和有限域上的运算，从而提高了安全性和效率。

然后，研究者们开始尝试使用不同的数学工具和技术来改进 PSS 协议。其中一项主要的改进是使用多项式残差码 (Polynomial Residue code, PRC) 来实现秘密共享。Prabhakaran 在 2004 年提出了基于 PRC 的 PSS 协议，该协议使用的是多项式加法和乘法，从而避免了有限域上的运算。在此之后，基手 PRC 的 PSS 协议不断得到完善，成为一种有效的秘密共享

2.2 Paillier 同态加密算法原理

目前加密算法大多基于计算复杂理论中的整数分解难题、离散对数问题、n 阶剩余类问题、近似最大公因子难题和子群判定难题等［10］。Paillier 同态加密算法在分析 n 阶剩余类难题的基础上保证其安全性，为云环境下的 KYC 客户模糊查询场景中的数据安全传输奠定基础。

Paillier 同态加密包括密钥生成、加密和解密环节，其中:

密钥生成环节: 随机选取两个大素数 p、q 和整数 y Z* n2 ，计算 n = pq 和 = lcm(p - 1，q - 1)，并使 gcd［L(y mod n2)，n］= 1 成立，此时公钥为 (n，g)，私钥为 。

密钥生成后，选择随机数 r Zn，计算对数据进行加密，得到密文 c，而 m 为加密信息，计算如式 (1) 所示。

$$c = E(m\ r) = ymrnmodn2 \tag{1}$$

在解密环节，基于复合剩余假设理论，由 p、q 推算出的 ，可以求得式 (1) 在定义域内的逆运算，即对于密文 c，经式 (2) 处理，得到明文 m，其中 L(i) = (i - 1) /n。

$$m = D(c\ ) = L(cmodn2)/L(ymodn2) * modn \tag{2}$$

由 Paillier 加密算法的加法同态性质，对于数据 x 和 t 的密文 E(x) 、E(t)，满足式 (3) 所示关系。

$$\begin{aligned} D\ E(x) \oplus E(t)modn2\ = \\ D\ (yxrn1) \oplus (ytrn2)E(t)modn2\ = \\ D\ yx + t(r1r2)nmodn2\ = D\ E(x+t)modn2 \end{aligned} \tag{3}$$

通过对 E(x) 、E(t) 的处理，得到 E(x + t)，即可得到数据 x + t 的具体值，该原理可推广至多组数据的情况。Paillier 算法的性质为数据加密的防篡改和安全性保障提供了灵活的思路。在此基础上，本文考虑 KYC 反洗钱客户模糊查询业务场景的特点，提出基于 Paillier 算法的“二次加密”模糊查询数据安全保障方案。

3 基于秘密分享阈值的隐私模糊匹配方案

假设 A 和 B 分别是存储在服务器和客户端中的姓名，每个姓名由多个单词组成，即 $A = A_1A_2...A_c$，其中 $A_i, i \in [1, c]$ 是姓名 A 中的第 i 个单词，相似地，$B = B_1B_2...B_d$，其中 $B_j, j \in [1, d]$。我们需要判断存储在服务器的姓名 A 和存储在客户端的姓名 B 是否指向同一个人，但是匹配的过程中服务器和客户端都不能让对方知道本方存储的姓名是什么，除非 A 和 B 两个名字相匹配，否则两方都不知道对方存储的姓名的任何信息。

当 A_i 和 B_j 中有一定数量的字母匹配时，我们就认为这两个单词时匹配的。而当姓名 A 和姓名 B 中有两个及以上的单词模糊匹配时，我们就认为姓名 A 和姓名 B 匹配，即它们指向的是同一个人。此外，在服务器上有一个函数 F 来决定两个单词匹配时这两个单词至少所拥有相同字母的数量。换言之，当两个单词 W_1 和 W_2 模糊匹配时，W_1 和 W_2 至少有 y 个字母相同，其中 $y = F(|W_1|, |W_2|)$。F 可以用表格来实现。

首先，客户端生成一组公私钥对 (PK, SK)，并把公钥 PK 发送给服务器，保留私钥 SK 在客户端本地。接着，客户端和服务器约定对称加密的方案 ENC 和一个哈希函数 H。对于任何明文信息 α，我们把用公钥 PK 加密后的密文记作 $[\alpha]_{PK}$。在客户端中，每个单词 A_i 按照拆分一串字母为 $a_{i1}a_{i2}...a_{il_i}$，其中 l_i 是单词 A_i 的长度，a_{it} 是单词 A_i 中的第 t 个字母。根据上述的字母串，我们可以得出基于单词 A_i 的多项式 $P_i(x) = \prod_{t=1}^{l_i}(x - a_{it})$。客户端对每个单词所对应的多项式 $P_i, i \in [1, c]$ 使用公钥 PK 进行加密并把这些加密后的多项式发送至服务器。相似地，服务器把每个单词 B_j 拆分称一串字母 $b_{j1}b_{j2}...b_{jl_j}$，其中 l_j 是单词 B_j 的长度，b_{jt} 是单词 B_j 中的第 t 个字母。此时，对姓名 B 的所有单词进行全排列，共得到有 $\theta = d!$ 种可能，记作 $\Pi = (\pi_1, \pi_2, ..., \pi_\theta)$。对于每一种可能 $\pi \in \Pi$，服务器执行算法 1。

服务器把算法 1 的结果 $(C_\pi, H(K), ENC_K(B)), \pi \in \Pi$ 发送至客户端，其中 $C_\pi = (c_{i1}, c_{i2}, ..., c_{il_i}), i \in [1, m]$。客户端使用算法 2 来计算客户端的一个姓名是否和服务器中的姓名匹配。

如果客户端通过计算算法 2 的结果是“Yes”，则客户端可以使用对称加密的密钥 K 来解密 $ENC_K(B)$ 来获取存储在服务器上与之相匹配的姓名。否则，如果算法 2 计算结果是“No”，这意味着客户端存储的姓名 A 与服务器存储的姓名 B 不匹配。

4 安全性证明

在本章中，我们将证明所提出的方案是隐私安全的，它保护了客户端和服务器的数据隐私，除了相匹配的数据外，不会泄漏任何额外的信息。简而言之，我们的方案提供了以下的安全性保证：

- 除了每个姓名的单词数和每个单词的长度外，客户端无法知道有关存储在服务器中姓名的任何信息，除非两边的姓名存在模糊的逻辑匹配
- 除了每个姓名的单词数和每个单词的长度外，服务器无法知道有关存储在客户端中姓名的任何信息

我们定义，如果客户端存储的姓名 A 有两个或以上的单词和服务器存储的姓名 B 存在模糊逻辑匹配，则称这两个姓名存在逻辑匹配。如果姓名中的单词 A_i 有一定数量的字母在 B_i 中出现，则称单词 A_i 和 B_i 模糊匹配。

让我们考虑如下的安全性试验 $Exp_A^{MatchPriv}(\lambda)$。这个试验将有一个挑战者和一个二阶段的对手 $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$。在试验 $Exp_A^{MatchPriv}(\lambda)$ 中，挑战者首先使用 $Setup$ 函数来生成公开

Algorithm 1 服务器计算隐私模糊匹配的算法

Require: ① 来自客户端的加密多项式 $[P_i(x)]_{PK}, i = 1, 2, ..., c$ ② 存储在服务器的姓名 B ③ $\pi = (z_1, z_2, ..., z_d)$ ④ 客户端生成的公钥 PK ⑤ 哈希函数 H ⑥ 对称加密算法 ENC

Ensure: 客户端 C （???）

$C_\pi = \emptyset$

服务器把姓名 B 解析成 $B_1 B_2 ... B_d$

$m = min(c, d)$

生成对称加密密钥 $K \leftarrow Z_q$

$(\Sigma_1, \Sigma_2, ..., \Sigma_m) \leftarrow SecretShare(K, m, 2)$

while $m > 0$ **do**

  单词 B_{zm} 中的字符数 $t = |B_{zm}|$

  $\mu = F(degree(P_m), t)$

  $(S_1, S_2, ..., S_t) \leftarrow SecretShare(\Sigma_m, t, \mu)$

  $(b_1, b_2, ..., b_t) \leftarrow B_{zm}$

  for $x : 1 \rightarrow t$ **do**

    生成随机数 $r \leftarrow Z_q$

    $c_{mx} = r * [P_i(b_x)]_{PK} + [S_x]_{PK}$

  end for

  $C_\pi \leftarrow C_\pi \cup \{(c_{m1}, cm2, ..., c_{mt})\}$

  $m = m - 1$

end while

return $(C_\pi, H(K), ENC_K(B))$

Algorithm 2 客户端计算隐私模糊匹配的算法

Require: ① 存储在客户端的姓名 A ② 来自服务器的计算结果 $(C_\pi = \{c_{i1}, c_{i2}, ..., c_{il_i}, i \in [1, m]\}, H(K), ENC_K(B))$

Ensure: Yes/No

客户端使用私钥 SK 解密：$D_{ij} \leftarrow Decrypt_{SK}(c_{ij}), i \in [1, m], j \in [1, l_i]$

for $\delta = 1 \rightarrow m$ **do**

  $f = F(degree(P_\delta), t_\delta)$

  $\gamma = 0$

  for 对 $1, 2, ..., t_\delta$ 所有的组合 $\binom{t_\delta}{f}$ $L_1, L_2, ..., L_f$ **do**

    $\sigma[\delta][\gamma] = Reconstruct(D_{\delta L_1}, D_{\delta L_2}, ..., D_{\delta L_f})$

    $\gamma + +$

  end for

end for

if $\exists \delta_1, \delta_2, \gamma_1, \gamma_2$, 使得 $H(K) = H(Reconstruct(\sigma[\delta_1][\gamma_1], \sigma[\delta_2][\gamma_2]))$ **then**

  return Yes

end if

return No

的参数。接着，对手 $\mathcal{A}_0$ 生成两个姓名 $\mathcal{B}_0$ 和 $\mathcal{B}_1$。$\mathcal{B}_0$ 和 $\mathcal{B}_1$ 含有相同数量的单词，而且姓名 $\mathcal{B}_0$ 中的每个单词和姓名 $\mathcal{B}_1$ 中的对应的单词都有相同数量的字母。$\mathscr{B}_0$ 是多重集合，其元素是姓名 $\mathcal{B}_0$ 中的每个单词的长度，相似地，$\mathscr{B}_1$ 中的其元素是姓名 $\mathcal{B}_1$ 中的每个单词的长度。因此 $\mathscr{B}_0 = \mathscr{B}_1$。如果 $\mathscr{B}_0 \neq \mathscr{B}_1$，则试验终止。否则，挑战者随机地选取 $\mathscr{B}_0$ 或 $\mathscr{B}_1$，并调用 $\mathcal{A}_1$, $\mathcal{A}_1$ 可以调用 oracle 的 $Fuzzy-Match()$。每次调用时 $\mathcal{A}_1$ 把姓名 A 传递到 oracle。A 应该使得 A 不与 $\mathcal{B}_0$ 和 $\mathcal{B}_1$ 逻辑匹配。接着，在 $\mathcal{B}_0$ 作为服务器输入，A 作为客户端输入的情况下，oracle 返回服务器输出。首先，oracle 将 A 中的单词转换为一组 $|A|$ 多项式。假设 $A = A_1A_2...A_c$, P_i 代表单词 $A_i, i \in [1, c]$。对于 $(1, 2, ..., c)$ 中的每种组合 π, oracle 执行算法 1，并把输出 $C_\pi, H(K), ENC_K(B)$ 发送到对手 $\mathcal{A}_1$。$\mathcal{A}_1$ 可以调用任意次数的 $Fuzzy-Match()$。对手 $\mathcal{A}_1$ 的目标是识别出 d。如果对手 $\mathcal{A}_1$ 能够正确地识别出 d，则该试验成功。

Algorithm 3 $Exp_A^{MatchPriv}(\lambda)$

$(PP, p, Pk, Sk) \leftarrow Setup(\lambda)$
$(B_0, B_1, r, st) \leftarrow \mathcal{A}_0(PP, p, n, Pk)$
Parse B_i as $B_{i1}, B_{i2}, ..., B_{ir} : i = 0, 1$
$\mathscr{B}_0 \leftarrow \{|B_{0i}| : i \in [1, r]\}_M$
$\mathscr{B}_1 \leftarrow \{|B_{1i}| : i \in [1, r]\}_M$
if $\mathscr{B}_0 \neq \mathscr{B}_1$ **then**
  return 0
end if
$d \leftarrow \{0, 1\}$
$d' \leftarrow \mathcal{A}_1^{Fuzzy-Match()}(st)$
return $d = d'$

Algorithm 4 $Fuzzy-Match(A)$

Parse A as $A_1A_2...A_c$
Parse B_i as $B_{i1}B_{i2}...B_{it} : i = 0, 1$
if $\exists k_1, k_2 \in [1, t], l_1, l_2 \in [1, c], k_1 \neq k_2, l_1 \neq l_2$ **then**
  满足对于任意的 $i \in [0, 1]$, $|B_{ik_1} \cap A_{l_1}| \geq F(|B_{ik_1}|, |A_{l_1}|) \wedge |B_{ik_2} \cap A_{l_2}| \geq F(|B_{ik_2}|, |A_{l_2}|)$
  return 0
else
  return 服务器输出 (B_d, A)
end if

对手 $\mathcal{A}$ 在试验 $Exp_A^{MatchPriv}(\lambda)$ 的优势通过 $Adv_{\mathscr{A}^{MatchPriv}}(\lambda) = |Pr[Exp_A^{MatchPriv}(\lambda) = 1] - \frac{1}{2}|$ 可以算出

5 实验验证与性能分析

为验证本方案对于数据加密的可靠程度，本文在模拟环境下对该算法和其他同类算法进行相同条件下的测试，并对结果进行分析比较。试验环境如表 1 所示。

在表 1 试验环境下，对本文所提加密方案进行试验验证，并与只进行一次加密的 Paillier 加密方案和基于 AES 算法、国密算法和 RSA 算法的方案进行相同条件下的测试，从算法的

Table 1: 表 1 试验环境

环境类型	说明
操作系统	Linux
运行内存	8Gb
CPU	Intel(R)Core(TM)i5 - 10210 CPU@1.60GHz

加解密时间、数据交互正确率等方面进行分析比较。在 Paillier 算法进行反洗钱客户模糊查询数据加解密的过程中，公钥中 n 的大小决定了密钥的复杂程度，也间接决定了加解密的时间。本文分别生成了 n 为 32bit、64bit、128bit、256bit、512bit、1024bit 和 2048bit 的密钥。对随机生成的相同长度数据进行多次加解密试验，对结果进行整理分析，得到如图所示的 Paillier 同态加密算法加密、二次加密和解密执行时间与密钥长度之间的关系。解密时间包括对“二次加密”密文的解密和对交易数据的解密过程。

Figure 1: Paillier 算法执行时间与密钥长度关系

由图 XX 可知，密钥长度为 1024bit 以下范围内，Paillier 同态加密算法随着密钥长度变大，其加解密时间略有增加但变化不大，超过 1024bit 以上长度，密钥相当复杂，所耗费的加解密时间也呈比例增加，对硬件的要求也有所增加。

在加解密不同长度反洗钱客户姓名数据信息的前提下，本文方法与只进行一次加解密的 Paillier 加解密方案以及其他加解密方所花费的加解密执行时间如图所示。

Figure 2: 不同算法加解密执行时间对比

由图 xxx 可知，与只进行一次加解密的 Paillier 算法加解密方案相比，本文方案在加解密的执行时间上略有增加，但相较于文献 xxxx 所提方法，本文方案在加解密过程的执行时间上具有一定优势。

除加解密过程的效率问题外，反洗钱客户模糊查询信息加解密成功率同样是衡量数据安全保障方案的重要指标，利用不同的加解密算法对不同长度反洗钱客户姓名数据进行多次加解密试验验证，结果如图 xxx 所示。

Figure 3: 不同算法加解密正确率

从图 xxx 可以看出：与文献 xxxx 算法相比，在相同密钥长度和相同加密信息的前提下，本文算法的加解密正确率均在 95% 以上与只进行一次 Paillier 加密的加密方案相比较，本文方案在加解密正确率上也同样具有优势，这表明该方案对于保障反洗钱客户模糊查询数据传输的可靠性具有积极作用。

6 结束语

本文结合反洗钱客户模糊查询过程的数据交互特点和 Paillier 加密算法的同态性质，提出基于 Paillier 同态加密性质的反洗钱客户模糊查询数据质量“二次加密”方案，从加解密的时间和正确率等方面进行试验验证，并与文献 xxx 加密算法进行对比。结果表明，本文方案加解密时间相对较短，数据加解密成功率较高，为保障反洗钱客户模糊查询数据传输安全提供了一种可行方案，同时对于反洗钱客户模糊查询的可靠交互具有借鉴作用。

References

[1] Zheng L, Wang S, Tian L, et al., Query-adaptive late fusion for image search and person re-identification, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015: 1741-1750.

[2] Arandjelović R, Zisserman A, Three things everyone should know to improve object retrieval, Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012: 2911-2918.

[3] Lowe D G. Distinctive image features from scale-invariant keypoints, International journal of computer vision, 2004, 60(2): 91-110.

[4] Philbin J, Chum O, Isard M, et al. Lost in quantization: Improving particular object retrieval in large scale image databases, Computer Vision and Pattern Recognition, 2008. CVPR 2008, IEEE Conference on, IEEE, 2008: 1-8.