

# 判断集合包含关系的安全计算协议

李荣花<sup>1),2)</sup> 武传坤<sup>3)</sup> 张玉清<sup>2)</sup>

<sup>1)</sup>(中国科学院研究生院 北京 100049)

<sup>2)</sup>(国家计算机网络入侵防范中心 北京 100049)

<sup>3)</sup>(中国科学院软件研究所信息安全国家重点实验室 北京 100080)

**摘 要** 研究了安全计算中关于集合的问题:  $A$  拥有一个秘密的集合  $S_A$ ,  $B$  拥有一个秘密的集合  $S_B$  ( $S_A$  和  $S_B$  来自一个全集), 双方希望知道  $S_A$  是否包含  $S_B$ , 但是不希望泄漏关于集合  $S_A$  和  $S_B$  的其它有用信息. 针对此问题, 提出了3个具有不同效率和安全性安全计算协议. 设集合  $S_B$  的大小为  $N_B$ . 第1个协议基于叠加密(或者支持门限解密的加法同态加密方案), 需要  $N_B$  轮通信. 另外两个协议基于普通的加法同态加密方案, 仅需一轮通信. 与同类成果比, 前两个协议使用了新的集合表示法, 第3个协议在输出结果阶段不需要门限解密, 通信效率较好.

**关键词** 安全计算; 集合包含; 叠加密; 同态加密

**中图法分类号** TP309 **DOI号**: 10.3724/SP.J.1016.2009.01337

## Secure Computation Protocol for Testing the Inclusion Relation of Sets

LI Rong-Hua<sup>1),2)</sup> WU Chuan-Kun<sup>3)</sup> ZHANG Yu-Qing<sup>2)</sup>

<sup>1)</sup>(Graduate University of Chinese Academy of Sciences, Beijing 100049)

<sup>2)</sup>(National Computer Network Intrusion Protection Center, Beijing 100049)

<sup>3)</sup>(State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences, Beijing 100080)

**Abstract** A set operation problem is considered in the secure computation setting. Player  $A$  has a secret set  $S_A$  and player  $B$  has a secret set  $S_B$ , where  $S_A$  and  $S_B$  belong to a universe. They want to know if  $S_A$  includes  $S_B$  without leaking anything else about their secret sets. Three protocols of different levels of security and efficiency are proposed for this problem. The first protocol can be based on the so-called Superposed Encryption scheme or threshold encryption scheme. Let  $N_B$  be the size of set  $S_B$ , the first protocol needs  $N_B$  rounds of communication. The other two protocols are based on standard public-key encryption schemes with additively homomorphic property, and need only one round of communication. Compared to previous results, the first two protocols use a new technique of representing sets, and the third protocol avoids threshold decryption in the output phase, and is more efficient.

**Keywords** secure computation; set-inclusion; superposed encryption; homomorphic encryption

## 1 引 言

1982年, Yao<sup>[1]</sup>提出了两方安全计算的问题, 后

来 Goldreich、Micali 和 Wigderson<sup>[2]</sup>把两方安全计算推广到多方安全计算. 多方安全计算要解决的问题是:  $n$  个成员  $P_1, P_2, \dots, P_n$ , 各自有一个秘密输入  $x_1, x_2, \dots, x_n$ , 他们要计算这些秘密输入的一个

函数  $f(x_1, x_2, \dots, x_n) = (y_1, y_2, \dots, y_n)$ , 成员  $P_i$  得到输出  $y_i$ , 条件是: 对于任何一个成员  $P_i$ , 除了  $y_i$  和  $x_i$  所隐含的信息,  $P_i$  不能得到任何其它信息. 一个解决多方安全计算问题的协议由多个分布式算法组成, 每个成员运行一个算法, 成员之间经过若干次交互完成计算任务, 得到相应的输出.

安全计算领域中有关集合的问题有: 求并集<sup>[3]</sup>、求并集的大小<sup>[3]</sup>、求交集<sup>[3-4]</sup>、求交集的大小<sup>[3-4]</sup>以及一些变异问题<sup>[3]</sup>; 判断集合是否不相交<sup>[5]</sup>; 求集合的包含关系<sup>[6]</sup>等. 本文研究的是多方安全计算意义下判断集合是否存在包含关系, 针对此问题的协议其输出仅为一个比特, 比如 1 表示集合  $S_A$  包含集合  $S_B$ , 0 表示集合  $S_A$  不包含集合  $S_B$ .

1.1 相关工作

解决判断集合包含关系有两种途径: 一般多方安全计算协议和专门设计的安全计算协议. 一般多方安全计算协议(比如文献[2, 7])可以用于解决任何多方计算问题, 自然也包含判断集合包含关系的问题. 然而, 一般多方安全计算协议由于重点考虑的是一般性, 而不能兼顾效率问题.

文献[6]的作者提出了一个求集合包含关系的协议. 设集合  $S_A$  的大小为  $N_A$ , 集合  $S_B$  的大小为  $N_B$ , 协议的输出是整数  $p = |S_A \cap S_B|$ . 如果  $p = N_A$  或者  $p = N_B$  表示两集合存在包含关系; 如果  $p = 0$ , 则表示两集合不相交; 其它情况表示集合是相交的但不是包含关系. 这实际上是一个求集合交集的势的协议.

Kissner 和 Song<sup>[3]</sup> 提出了一个判断集合子集关系的协议, 协议的输出为一个比特, 代表集合  $S_A$  是否包含集合  $S_B$ , 协议需要门限加密方案. 对于门限加密方案来说, 解密过程比较复杂, 一般需要分享解密密钥的各个成员贡献一个解密值(decryption share), 并证明这个解密值的正确性, 由这些解密值来得到明文.

文献[5]的作者提出了 3 个判断集合是否不相交的安全计算协议: PIPE #1、PIPE #2 和 PIPE #3 (PIPE 是 Private Intersection Predicate Evaluation 的缩写). 其中协议 PIPE #2 基于叠加密(Superposed Encryption<sup>[5]</sup>), 协议 PIPE #3 基于普通的同态加密方案. 两个协议的输出都是一个比特, 表示集合是否不相交, 其区别在于通信复杂度和计算复杂度不同.

1.2 本文的结果

针对判断集合包含关系的问题, 提出了 3 个具

有不同程度安全性的协议. 协议 1 和协议 2 借鉴文献[5]中判断集合是否不相交的协议的思想, 协议 1 基于叠加密或者支持门限解密的加法同态加密方案, 协议 2 基于普通的加法同态加密. 假设集合  $S_B$  的大小为  $N_B$ , 协议 1 需要  $N_B$  轮通信, 协议 2 需一轮通信. 协议 3 基于加法同态加密, 需一轮通信.

虽然在特殊情况下, 前两个协议向  $A$  泄漏了集合  $S_B$  的信息, 而第 3 个协议不泄漏关于  $S_A$  和  $S_B$  的其它任何信息, 然而前两个协议采用了新的集合表示法(见 2.4 节表示法 1), 跟第 3 个协议采用的集合多项式表示法不同. 新的集合表示法可能会有其它的用处.

本文的协议实际上与文献[6]中的求集合包含关系的协议有很大区别. 本文的 3 个协议解决的是判断集合包含关系的问题, 协议的输出仅为一个比特(代表集合  $S_A$  是否包含集合  $S_B$ ), 文献[6]中的求集合包含关系的协议解决了更广泛一些的问题——判断两个集合是具有包含关系、不相交关系、还是相交非包含关系, 其一般性以泄漏集合交集的大小为代价. 文献[6]的协议需要求出交集的大小  $|S_A \cap S_B|$ , 然后根据  $|S_A \cap S_B|$  和  $|S_A|$ 、 $|S_B|$  判断两个集合之间是包含关系、相交非包含关系, 还是不相交关系, 这实际上是求集合交集大小的协议而不是严格意义上的判断集合是否具有包含关系的协议.

与 Kissner 和 Song<sup>[3]</sup> 提出的判断集合包含关系的协议相比, 本文的协议 1 和协议 2 使用的多项式表示法与文献[3]中使用的表示法不同, 从另一方面显示了多项式在集合操作中的用处. 本文的协议 3 与文献[3]中的协议使用相同的多项式表示法来表示集合, 但是仅需要普通的同态加密方案, 避免使用门限解密过程.

2 基础知识

2.1 概念

**可忽略函数.** 函数  $\epsilon(k)$  是可忽略的如果对于任意一个多项式  $p(k)$ , 存在一个数  $k_0$ , 使得对于所有的  $k > k_0$ , 有  $\epsilon(k) < 1/p(k)$ .

**概率族**(probability ensembles)<sup>[8]</sup>. 一个以  $S \subseteq \{0, 1\}^*$  为索引的概率族是一个集  $\{X_w\}_{w \in S}$ , 使得每一个  $X_w$  是一个取遍  $\{0, 1\}^{poly(|w|)}$  的随机变量. 一般, 考虑  $S = \{0, 1\}$  和  $S = \{1^n : n \in N\}$ , 这里  $N$  是自然数的集合.

**计算不可区分**<sup>[8]</sup>.  $X$  和  $Y$  是计算不可区分的

(记作  $X \stackrel{c}{=} Y$ ) 当且仅当对于每一个多项式时间的算法族、 $\{C_n\}_{n \in \mathbb{N}}$ 、每一个多项式  $p(n)$  和所有足够长的  $w \in S$  有

$$|Pr[C_n(X_w=1)] - Pr[C_n(Y_w=1)]| < 1/p(|w|).$$

## 2.2 加法同态加密

公钥加密方案由密钥产生算法 KGen、加密算法 E、解密算法组成 D. 设  $M, C, R$  分别为明文空间、密文空间和随机空间.

(1) 密钥生成算法 KGen 的输入为一个整数  $k$  (通常称为安全参数), 输出为一对公私钥  $(pk, sk)$ .

(2) 加密算法 E:  $M \times R \rightarrow C$ .

(3) 解密算法 D:  $C \rightarrow M$ .

加密算法具有的加法同态性质是指: 对于两个给定的密文  $c_1 = E(m_1), c_2 = E(m_2)$ , 在不需要解密的情况下, 可以计算出  $m_1 + m_2$  的一个密文  $c' = E(m_1 + m_2) = E(m_1)E(m_2)$ . 对于一个给定的密文  $c = E(m)$  和一个常数  $a$ , 在不需要解密的情况下可以计算出  $am$  的一个密文  $c' = E(am) = E(m)^a$ .

本文需要同态加密方案具有语义安全性(文献[9]提出的安全概念), 即密文不泄漏明文的任何信息. 这可以通过一个攻击游戏说明: 假定有一个敌手, 选择了两个等长的明文  $m_0, m_1$ , 敌手把两个消息发送给用户; 用户随机地选取一个比特  $b \in \{0, 1\}$ , 并对  $m_b$  进行加密得到密文  $c^* = E(m_b)$ , 把  $c^*$  发送给敌手; 则敌手正确猜测  $b$  的概率为  $1/2 + \epsilon(k)$ , 这里  $\epsilon(k)$  是可忽略函数.

## 2.3 叠加密(Superposed Encryption)[5]

在叠加密方案中, 有两个成员  $A$  和  $B$  各自有一对定义在共同系统参数(比如共同的素数模)上的公私钥. 叠加密方案由密钥生成算法  $K$  和  $K'$ 、加密算法  $E_{pk_X}$  和  $E_{pk_A, pk_B}^{\sup, X}$ 、解密算法  $D_{sk_X}$  和  $D_{sk_X}^{\sup}$  ( $X \in \{A, B\}$ ) 组成.

(1) 密钥生成算法由产生公开参数  $param$  的初始步骤  $K$  和密钥生成算法  $K'$  组成. 给定一个整数  $k$ , 初始步骤的算法为  $param \leftarrow K(k)$ ; 给定参数  $param$ , 算法  $K'$  为  $A$  和  $B$  两个用户各产生一对公私钥:  $(pk_A, sk_A), (pk_B, sk_B) \leftarrow K'(param)$ .

(2) 对于每个成员  $X \in \{A, B\}$ : 加密算法  $E_{pk_X}$ :  $P \rightarrow C$ ;  $E_{pk_A, pk_B}^{\sup, X}$ :  $P \times C \rightarrow C^{\sup}$ .

(3) 解密算法满足以下条件: 对于所有的  $m \in P$ , 如果  $X \in \{A, B\}$ , 则有  $D_{sk_X}(E_{pk_X}(m)) = m$ . 对于固定的  $c \in E_{pk_X}(m')$ , 如果  $c'$  按照  $D_{sk_X}^{\sup}(E_{pk_A, pk_B}^{\sup, X}(m, c))$  分布, 则  $c'$  在  $E_{pk_X}(m \cdot m')$  上均匀分布, 这里

$X \in \{A, B\}$  并且  $\bar{X}$  等于  $\{A, B\} - X$ .

加密算法  $E_{pk_X}$  具有加法同态性质:

$$E_{pk_X}(m)E_{pk_X}(m') = E_{pk_X}(m + m').$$

对于给定的密文  $E_{pk_X}(m)$  和一个常数  $a$ , 可以得到  $am$  的密文  $E_{pk_X}(am) = (E_{pk_X}(m))^a$ .

对于固定的  $m$  和  $m'$ , 叠加密  $E_{pk_A, pk_B}^{\sup, X}(m, E_{pk_X}(m'))$  和  $E_{pk_A, pk_B}^{\sup, X}(m, E_{pk_X}(m'))$  是计算不可区分的, 这里  $X \in \{A, B\}$  并且  $\bar{X}$  等于  $\{A, B\} - X$ .

在文献[5]中, 给出了一个具体的叠加密方案.

(1) 给定  $k$ , 算法  $K$  产生一个素数  $p = 2q + 1$ , 这里  $q$  也是素数; 选择  $g' \in \mathbb{Z}_p^*$ , 并令  $g = (g')^2 \pmod{p}$ , 在以  $g$  为生成元的群上随机选择一个元素  $h$ . 公开参数为  $param = (p, q, g, h)$ . 给定  $param$ , 用户密钥的生成过程为: 随机选择一个元素  $x \in \mathbb{Z}_q$ , 令  $y_X = g^x \pmod{p}$ , 则用户的公钥为  $pk_X = y_X$ , 私钥为  $sk_X = x$ .

(2) 加密算法  $E_{pk_X}$ : 对于在  $\mathbb{Z}_q$  中随机选择的元素  $r$ ,  $E_{pk_X}(m) = (g^r, y_X^r h^m)$ , 这里  $X \in \{A, B\}$ .  $A$  的叠加密算法  $E_{pk_A, pk_B}^{\sup, A}$ : 对于密文  $(G, H)$  和明文  $m$ , 在  $\mathbb{Z}_q$  中随机选取元素  $r$  和  $r'$ ,  $E_{pk_A, pk_B}^{\sup, A}(m, (G, H)) = (g^r, G^m g^{r'}, y_A^r y_X^{r'} H^m)$ .  $B$  的叠加密算法  $E_{pk_A, pk_B}^{\sup, B}$ : 对于密文  $(G, H)$  和明文  $m$ , 在  $\mathbb{Z}_q$  中随机选取两个元素  $r$  和  $r'$ , 叠加密过程:  $E_{pk_A, pk_B}^{\sup, B}(m, (G, H)) = (G^m g^r, g^{r'}, y_A^r y_X^{r'} H^m)$ .

(3)  $A$  的解密算法  $D_{sk_A}^{\sup}$ : 对于密文  $c = (G_A, G_B, Y)$ ,  $D_{sk_A}^{\sup}(c) = (G_B, YG_A^{-sk_A})$ .  $B$  的解密算法  $D_{sk_B}^{\sup}$ : 对于密文  $c = (G_A, G_B, Y)$ ,  $D_{sk_B}^{\sup}(c) = (G_A, YG_A^{-sk_B})$ .

上述方案是一个叠加密方案:

$$D_{sk_A}^{\sup}(E_{pk_A, pk_B}^{\sup, B}(m, E_{pk_A}(m')))) =$$

$$D_{sk_A}^{\sup}((g^r, g^{r'}, y_A^r y_X^{r'} h^{m \cdot m'})) = (g^r, y_X^{r'} h^{m \cdot m'}).$$

**定理 1**(见文献[5]). 在判定性 Diffie-Hellman 假想(DDH)下, 上面的叠加密方案满足选择明文安全性(叠加密方案的选择明文安全性也即是前面提到的语义安全性).

## 2.4 集合的多项式表示

**表示法 1.** 设集合  $S_1$  和  $S_2$  来自同一个全集  $U$ ,  $S_1 = \{a_1, a_2, \dots, a_m\}$ ,  $S_2 = \{b_1, b_2, \dots, b_n\}$ . 可以构造一个  $m$  次多项式  $f(x) = \alpha_0 + \alpha_1 x + \dots + \alpha_m x^m$ , 使得  $f(a) = 1$  当且仅当  $a \in S_1$ , 也就是说集合  $S_1$  可以用多项式  $f(x)$  来表示. 如果  $f(b_1)f(b_2)\dots f(b_n) = 1$ , 则有  $S_1 \supseteq S_2$ .

**表示法 2.** 设集合  $S_1$  和  $S_2$  来自同一个全集  $U$ ,  $S_1 = \{a_1, a_2, \dots, a_m\}$ ,  $S_2 = \{b_1, b_2, \dots, b_n\}$ . 可以构

造一个  $m$  次多项式  $f(x) = \alpha_0 + \alpha_1 x + \dots + \alpha_m x^m$ , 使得  $f(a) = 0$  当且仅当  $a \in S_1$ . 如果  $f(b_1) + f(b_2) + \dots + f(b_n) = 0$ , 则有  $S_1 \supseteq S_2$ .

## 2.5 协议安全性定义

**定义 1** (针对半诚实行为)<sup>[9]</sup>. 设  $f$  是一个确定性  $n$  元函数,  $f$  代表  $f(x_1, \dots, x_n)$  的第  $i$  个项. 对于  $I = \{i_1, \dots, i_t\} \subseteq \{1, \dots, n\}$ , 令  $f_I(x_1, \dots, x_n)$  代表  $f_{i_1}(x_1, \dots, x_n), \dots, f_{i_t}(x_1, \dots, x_n)$  组成的序列. 设  $\pi$  是计算  $f$  的有  $n$  成员参与的协议. 在  $\pi$  的一次以  $\bar{x} = (x_1, \dots, x_n)$  为输入的运行中, 第  $i$  个成员的所见记作  $VIEW_i^\pi(\bar{x})$ , 被定义为  $(x_i, r_i, m_1, \dots, m_{N_i})$  (这里  $r_i$  代表第  $i$  个成员的随机串,  $m_j$  代表他接收到的第  $j$  个消息,  $N_i$  是该成员收到的消息的数目), 对于  $I = \{i_1, \dots, i_t\}$ , 令  $VIEW_I^\pi(\bar{x}) = (I, VIEW_{i_1}^\pi(\bar{x}), \dots, VIEW_{i_t}^\pi(\bar{x}))$ , 则称  $\pi$  安全计算了  $f$ , 如果存在多项式时间的算法  $S$ , 使得对于每一个如上所述的  $I$ , 有

$$S((I, (x_{i_1}, \dots, x_{i_t}), f_I(\bar{x})))_{\bar{x} \in \{0,1\}^n} \equiv VIEW_I^\pi(\bar{x})_{\bar{x} \in \{0,1\}^n}.$$

## 3 判断集合包含关系的协议

协议的成员如果按照协议要求做运算和发送消息, 就称成员为半诚实成员; 如果协议成员不按照协议要求执行, 则称为恶意成员. 协议如果对于半诚实成员安全, 则协议在半诚实模型中是安全的; 如果协议对于恶意成员是安全的, 则协议在恶意模型中是安全的. 本文假定成员是半诚实的.

### 3.1 协议 1

假定  $A$  有一对叠加密钥  $(pk_A, sk_A)$ ,  $B$  有一对叠加密钥  $(pk_B, sk_B)$ . 设  $A$  的集合为  $S_A$ , 大小为  $N_A$ ;  $B$  的集合为  $S_B$ , 大小为  $N_B$ .  $S_A$  和  $S_B$  属于同一全集  $U$ .  $A$  定义一个  $N_A$  次多项式  $f(x) = \alpha_0 + \alpha_1 x + \dots + \alpha_{N_A} x^{N_A}$ , 使得  $f(a) = 1$  当且仅当  $a \in S_A$ , 这里  $\alpha_0, \dots, \alpha_{N_A} \in Z_q$ .  $A$  计算一个初始值  $c^* = E_{pk_A, pk_B}^{\sup, B}(1, E_{pk_A}(1))$ . 对于  $j = 1, \dots, N_B - 1$ :

第  $j.1$  步.  $A$  计算  $c = D_{sk_A}^{\sup}(c^*)$ , 然后  $A$  计算  $N_A + 1$  个密文:  $c_i^* = E_{pk_A, pk_B}^{\sup, A}(\alpha_i, c)$ ,  $i = 0, \dots, N_A$ .  $A$  把叠加密文  $c_0^*, \dots, c_{N_A}^*$  发送给  $B$ .

第  $j.2$  步.  $B$  解密  $A$  发来的密文得到  $c_0 = D_{sk_B}^{\sup}(c_0^*), \dots, c_{N_A} = D_{sk_B}^{\sup}(c_{N_A}^*)$ , 然后  $B$  计算  $c' = E_{pk_A}(0) c_0 (c_1)^{b_1} \dots (c_{N_A})^{b_{N_A}}$  以及  $c^* = E_{pk_A, pk_B}^{\sup, B}(1, c')$ .  $B$  把  $c^*$  发送给  $A$ . 如果  $j = N_B$ , 那么  $B$  不计算  $c^*$  而是计算  $c^{\text{Fin}} = c' E_{pk_A}(0)$ ,  $B$  把  $c^{\text{Fin}}$  发送给  $A$ .

输出结果.  $A$  解密  $c^{\text{Fin}}$  得到  $s = D_{sk_A}(c^{\text{Fin}})$ . 如果  $s = 1$ , 则  $A$  输出 1 (表示  $S_A$  包含  $S_B$ ); 否则  $A$  输出 0 (表示  $S_A$  不包含  $S_B$ ).

协议主要思想是在第  $j$  轮中把  $f(b_j)$  的值乘到已有的乘积中. 对于  $j = 1$ ,  $B$  计算出  $c' = E_{pk_A}(f(b_1))$ , 对于  $j > 1$ ,  $B$  计算出  $c' = E_{pk_A}(f(b_1) \dots f(b_j))$ . 在第  $N_B$  轮  $B$  计算出  $c^{\text{Fin}} = E_{pk_A}(f(b_1) \dots f(b_{N_B}))$ .

**例 1.** 采用前文提到的具体的叠加密方案, 对协议做进一步解释. 首先  $A$  令初始值为

$$c^* = E_{pk_A, pk_B}^{\sup, B}(1, E_{pk_A}(1)) = E_{pk_A, pk_B}^{\sup, B}(1, (g^r, y_A^r h)) \\ = (g^{r+r'}, g^{r''}, g^{sk_A r' + sk_B r'' + sk_A r' h}).$$

协议的第一轮.  $A$  令

$$c = D_{sk_A}^{\sup}(c^*) = (g^{r''}, g^{sk_B r'' + sk_A r' + sk_A r' h} g^{-sk_A(r+r')}) \\ = (g^{r''}, g^{sk_B r''} h).$$

$A$  计算

$$c_0^* = E_{pk_A, pk_B}^{\sup, A}(\alpha_0, c) = \dots \\ = (g^r, g^{r'a_0 + r''}, g^{sk_A r' + sk_B r'' + sk_B r'' a_0} h^{a_0}), \\ \dots, \\ c_{N_A}^* = E_{pk_A, pk_B}^{\sup, A}(\alpha_{N_A}, c) = \dots \\ = (g^r, g^{r'a_{N_A} + r''}, g^{sk_A r' + sk_B r'' + sk_B r'' a_{N_A}} h^{a_{N_A}}).$$

$A$  把  $c_0^*, \dots, c_{N_A}^*$  发送给  $B$ .  $B$  计算

$$c_0 = D_{sk_B}^{\sup}(c_0^*) = \dots = (g^r, g^{sk_A r' h^{a_0}}), \\ \dots,$$

$$c_{N_A} = D_{sk_B}^{\sup}(c_{N_A}^*) = \dots = (g^r, g^{sk_A r' h^{a_{N_A}}}).$$

$B$  接着计算

$$c' = E_{pk_A}(0) c_0 (c_1)^{b_1} \dots (c_{N_A})^{b_{N_A}} = \dots = E_{pk_A}(f(b_1)),$$

$B$  令

$$c^* = E_{pk_A, pk_B}^{\sup, B}(1, c') = (g^{R+r}, g^{r'}, y_A^r y_B^{r'} g^{sk_A R} h^{f(b_1)}),$$

$B$  发送  $c^*$  给  $A$ .

协议的第 2 轮.  $A$  令

$$c = D_{sk_A}^{\sup}(c^*) = \dots = (g^{R+r}, g^{r'}, y_A^r y_B^{r'} g^{sk_A R} h^{f(b_1)}).$$

$A$  计算

$$c_0^* = E_{pk_A, pk_B}^{\sup, A}(\alpha_0, c) = \dots \\ = (g^r, g^{r'a_0 + r''}, g^{sk_A r' + sk_B r'' + sk_B r'' a_0} h^{a_0 f(b_1)}), \\ \dots, \\ c_{N_A}^* = E_{pk_A, pk_B}^{\sup, A}(\alpha_{N_A}, c) = \dots \\ = (g^r, g^{r'a_{N_A} + r''}, g^{sk_A r' + sk_B r'' + sk_B r'' a_{N_A}} h^{a_{N_A} f(b_1)}).$$

$A$  把  $c_0^*, \dots, c_{N_A}^*$  发送给  $B$ .  $B$  计算

$$c_0 = D_{sk_B}^{\sup}(c_0^*) = \dots = (g^r, g^{sk_A r' h^{a_0 f(b_1)}}), \\ \dots,$$

$$c_{N_A} = D_{sk_B}^{\sup}(c_{N_A}^*) = \dots = (g^r, g^{sk_A r' h^{a_{N_A} f(b_1)}}).$$

$B$  接着计算

$$c' = E_{pk_A}(0) c_0 (c_1)^{b_2} \dots (c_{N_A})^{b_{N_A}} = \dots = E_{pk_A}(f(b_1) f(b_2)),$$

$B$  令

$$c^* = E_{pk_A, pk_B}^{\sup, B}(1, c') = (g^{R+r}, g^{r'}, y_A^r y_B^{r'} g^{sk_A R} h^{f(b_1) f(b_2)}),$$

$B$  发送  $c^*$  给  $A$ .

依此类推,直到第  $N_B$  轮:  $B$  计算

$c' = E_{pk_A}(0)c_0(c_1)^{b_{N_B}} \cdots (c_{N_A})^{b_{N_B}^{N_A}} = \cdots = E_{pk_A}(f(b_1) \cdots f(b_{N_B}))$   
后,  $B$  计算  $c^{\text{Fin}} = c'E_{pk_A}(0)$ ,  $B$  把  $c^{\text{Fin}}$  发送给  $A$ .

输出结果.  $A$  解密  $c^{\text{Fin}}$  得到  $s = D_{sk_A}(c^{\text{Fin}})$ . 如果  $s = h$ , 则  $A$  输出 1 (表示  $S_A$  包含  $S_B$ ), 否则  $A$  输出 0 (表示  $S_A$  不包含  $S_B$ ).

一般情况下,  $A$  只能从协议 1 获得一个比特的信息, 即  $S_A$  是否包含  $S_B$ , 因为要么  $A$  解密得到  $h$ , 要么是一个数  $y = h^x \bmod p$  ( $h$  是加密密钥的一部分). 当  $A$  解密得到的值不等于  $h$  时, 一般情况下  $A$  不能得到  $x = f(b_1) \cdots f(b_{N_B})$  的值, 因为这相当于求离散对数. 当然, 当  $x$  比较小时可以通过穷搜索找到  $x$ , 从而  $A$  可以获得关于集合  $S_B$  的信息. 比如  $A$  求出  $x$  后, 在极端的情况下如  $N_B = 1$ , 则由  $x = F(b_1)$  的值可以求出  $b_1$ , 在  $N_B > 1$  的情况下,  $A$  能知道  $S_B$  可能含有什么元素, 比如当  $N_B = 2$  时,  $x = F(b_1, \dots, b_2) = 15$ , 则有可能  $f(b_1) = 1, f(b_2) = 15$  或者  $f(b_1) = 3, f(b_2) = 5$ .

### 3.2 协议 2

上述协议需要  $N_B$  轮通信, 通过检查  $F(b_1, \dots, b_{N_B}) = f(b_1) \cdots f(b_{N_B})$  的值是否为 1 来判断集合  $S_A$  是否包含集合  $S_B$ . 改动一下协议 1:  $A$  可以把  $F(x_1, \dots, x_{N_B})$  的系数的密文发送给  $B$ ,  $B$  计算  $F(b_1, \dots, b_{N_B})$  的密文并发送给  $A$ ,  $A$  根据解密的结果判断集合  $S_A$  是否包含集合  $S_B$ . 这样协议仅需一轮通信.

函数  $F(x_1, \dots, x_{N_B}) = (\alpha_0 + \alpha_1 x_1 + \cdots + \alpha_{N_A} x_1^{N_A}) \cdots (\alpha_0 + \alpha_1 x_{N_B} + \cdots + \alpha_{N_A} x_{N_B}^{N_A})$ , 直观地看,  $F(x_1, \dots, x_{N_B})$  有  $(N_A + 1)^{N_B}$  个系数, 其实, 真正不同的系数有  $\binom{N_A + N_B}{N_B}$  个. 记这些系数为  $cf[1]$ ,  $cf[2], \dots, cf[\binom{N_A + N_B}{N_B}]$ .

**例 2.**  $N_A = 4, N_B = 2$ , 则  $F(b_1, b_2)$  为  
 $F(b_1, b_2) = (\alpha_0 + \alpha_1 b_1 + \alpha_2 b_1^2 + \alpha_3 b_1^3 + \alpha_4 b_1^4) \cdot$   
 $(\alpha_0 + \alpha_1 b_2 + \alpha_2 b_2^2 + \alpha_3 b_2^3 + \alpha_4 b_2^4)$   
 $= \alpha_0^2 + \alpha_0 \alpha_1 b_1 + \alpha_0 \alpha_2 b_1^2 + \alpha_0 \alpha_3 b_1^3 + \alpha_0 \alpha_4 b_1^4 +$   
 $\alpha_0 \alpha_1 b_2 + \alpha_1^2 b_1 b_2 + \alpha_1 \alpha_2 b_1^2 b_2 + \alpha_1 \alpha_3 b_1^3 b_2 + \alpha_1 \alpha_4 b_1^4 b_2 +$   
 $\alpha_0 \alpha_2 b_2^2 + \alpha_1 \alpha_2 b_1 b_2^2 + \alpha_2^2 b_1^2 b_2^2 + \alpha_3 \alpha_2 b_1^3 b_2^2 + \alpha_4 \alpha_2 b_1^4 b_2^2 +$   
 $\alpha_0 \alpha_3 b_2^3 + \alpha_1 \alpha_3 b_1 b_2^3 + \alpha_2 \alpha_3 b_1^2 b_2^3 + \alpha_3^2 b_1^3 b_2^3 + \alpha_4 \alpha_3 b_1^4 b_2^3 +$   
 $\alpha_0 \alpha_4 b_2^4 + \alpha_1 \alpha_4 b_1 b_2^4 + \alpha_2 \alpha_4 b_1^2 b_2^4 + \alpha_3 \alpha_4 b_1^3 b_2^4 + \alpha_4^2 b_1^4 b_2^4$   
 $= \alpha_0^2 + \alpha_0 \alpha_1 (b_1 + b_2) + \alpha_0 \alpha_2 (b_1^2 + b_2^2) + \alpha_0 \alpha_3 (b_1^3 + b_2^3) +$   
 $\alpha_0 \alpha_4 (b_1^4 + b_2^4) + \alpha_1^2 b_1 b_2 + \alpha_1 \alpha_2 (b_1^2 b_2 + b_1 b_2^2) +$   
 $\alpha_1 \alpha_3 (b_1^3 b_2 + b_1 b_2^3) + \alpha_1 \alpha_4 (b_1^4 b_2 + b_1 b_2^4) + \alpha_2^2 b_1^2 b_2^2 +$

$$\alpha_2 \alpha_3 (b_1^3 b_2^2 + b_1^2 b_2^3) + \alpha_2 \alpha_4 (b_1^4 b_2^2 + b_1^2 b_2^4) + \alpha_3^2 b_1^3 b_2^3 +$$

$$\alpha_3 \alpha_4 (b_1^4 b_2^3 + b_1^3 b_2^4) + \alpha_4^2 b_1^4 b_2^4.$$

可见关于  $b_1, b_2$  的多项式的不同系数共有  $\binom{4+2}{2} = 15$  个. 因此可以把协议 1 改变一下,  $A$  仅发送多项式系数的密文给  $B$ , 然后  $B$  计算  $F(b_1, \dots, b_{N_B})$  的密文, 由  $A$  解密得到计算结果. 下面先描述  $F(b_1, \dots, b_{N_B})$  的表达形式.

设  $I = \{(N_A, N_A, \dots, N_A), (N_A, N_A, \dots, N_A - 1), \dots, (0, 0, \dots, 0)\}$ ,  $I$  中的  $j$  个元素记作  $I_j$ . 对于第  $j$  个元素  $I_j = (i_1[j], i_2[j], \dots, i_{N_B}[j])$  来说,  $\{0, 1, \dots, N_A\}$  中的元素在  $I_j$  中出现的次数记为  $oc_j[0], oc_j[1], \dots, oc_j[N_A]$  (注意  $oc_j[0] + oc_j[1] + \cdots + oc_j[N_A] = N_B$ ).  $I_j$  可以有  $T_j$  个不同置换:

$$T_j = \frac{N_B!}{(oc_j[0])! (oc_j[1])! \cdots (oc_j[N_A])!}.$$

将  $F(b_1, \dots, b_{N_B})$  的  $\binom{N_A + N_B}{N_B}$  个不同系数按照下标排序, 即组成序列  $I, T_j$  代表系数  $\alpha_{i_1[j]} \alpha_{i_2[j]} \cdots \alpha_{i_{N_B}[j]}$  在  $F(b_1, \dots, b_{N_B})$  中出现的次数.

设  $I_j$  的一个置换为  $I_j^{(v)} = (i_1^{(v)}[j], i_2^{(v)}[j], \dots, i_{N_B}^{(v)}[j])$ , 记

$$e_j = b_1^{i_1^{(1)}[j]} b_2^{i_2^{(1)}[j]} \cdots b_{N_B}^{i_{N_B}^{(1)}[j]} + b_1^{i_1^{(2)}[j]} b_2^{i_2^{(2)}[j]} \cdots b_{N_B}^{i_{N_B}^{(2)}[j]} + \cdots +$$

$$b_1^{i_1^{(T_j)}[j]} b_2^{i_2^{(T_j)}[j]} \cdots b_{N_B}^{i_{N_B}^{(T_j)}[j]},$$

则函数

$$F(b_1, \dots, b_{N_B}) = (cf[1])^{e_1} + (cf[2])^{e_2} + \cdots +$$

$$(cf[\binom{N_A + N_B}{N_B}])^{e_{\binom{N_A + N_B}{N_B}}}.$$

假定  $A$  拥有一个普通加法同态加密方案 (相对于叠加密方案的公私钥,  $B$  知道这个方案的公钥, 协议 2 如下:

1.  $A$  把  $F(b_1, \dots, b_{N_B})$  的系数的密文  $E(cf[1]), E(cf[2]), \dots, E(cf[\binom{N_A + N_B}{N_B}])$  发给  $B$ ;

2.  $B$  计算

$$E(F(b_1, \dots, b_{N_B})) = (E(cf[1]))^{e_1} (E(cf[2]))^{e_2} \cdots$$

$$(E(cf[\binom{N_A + N_B}{N_B}]))^{e_{\binom{N_A + N_B}{N_B}}},$$

并把这个密文发给  $A$ .

3.  $A$  解密看结果是否为 1 即可判断出  $S_A$  是否包含  $S_B$ .

协议 2 使用加法同态加密方案, 实际上  $A$  最后得到的解密结果是某个数  $y = h^x$  ( $h$  是加密密钥的一部分), 其指数为  $x = f(b_1) \cdots f(b_{N_B})$ , 因此同协议 1 类似, 协议 2 中  $A$  一般只能做到判断指数是否为 1

(集合  $S_A$  是否包含集合  $S_B$ ), 也即协议 2 的输出是一个比特.

3.3 协议 3

假设  $A$  拥有一个普通加法同态加密方案(相对于叠加密方案)的公私钥,  $B$  知道其公钥.  $A$  构造  $N_A$  次多项式  $f(x) = \alpha_0 + \alpha_1 x + \dots + \alpha_{N_A} x^{N_A}$ ,  $f(a) = 0$  当且仅当  $a \in S_A$ . 协议 3 如下:

- 1.  $A$  把多项式的系数的密文  $E(\alpha_0), \dots, E(\alpha_{N_A})$  发给  $B$ ;
- 2.  $B$  计算

$$c = E(\alpha_0)(E(\alpha_1))^{(b_1+b_2+\dots+b_{N_B})}(E(\alpha_2))^{(b_1^2+b_2^2+\dots+b_{N_B}^2)} \dots$$
$$(E(\alpha_{N_A}))^{(b_1^{N_A}+b_2^{N_A}+\dots+b_{N_B}^{N_A})},$$

并计算  $c^* = c^{random}$ ,  $B$  把  $c^*$  发送给  $A$ .

- 3.  $A$  解密  $c^*$  看结果是否为 0, 如果为 0, 则表示集合  $S_A$  不包含集合  $S_B$ ; 否则集合  $S_A$  包含集合  $S_B$ .

在上述协议中,  $A$  从  $B$  处得到的密文为

$$c^* = (\alpha_0 + \alpha_1(b_1 + b_2 + \dots + b_{N_B}) + \dots + \alpha_{N_A}(b_1^{N_A} + b_2^{N_A} + \dots + b_{N_B}^{N_A}))random$$
$$= (f(b_1) + \dots + f(b_{N_B}))random.$$

因此解密得到的是  $(f(b_1) + \dots + f(b_{N_B}))random$ , 当  $f(b_1) + \dots + f(b_{N_B}) = 0$  时,  $A$  得到 0 (同样, 由于使用的是加法同态加密方案, 实际上这个 0 是位于指数上的), 当  $f(b_1) + \dots + f(b_{N_B}) \neq 0$  时,  $A$  得到的是一个未知随机数(同样位于指数位置上). 可见  $A$  仅能判断  $f(b_1) + \dots + f(b_{N_B})$  是否为 0, 因此协议 3 的输出仅包含一个比特的信息(即集合  $S_A$  是否包含集合  $S_B$ ).

当  $f(b_1) + \dots + f(b_{N_B}) \neq 0$  时, 无论  $f(b_1) + \dots + f(b_{N_B})$  的值是大还是小, 协议 3 都不泄漏关于  $S_B$  的信息, 这是协议 3 与前两个协议主要的不同之处. 协议 3 仅泄漏了集合  $S_A$  的大小, 但这不是协议本身的缺陷. 泄漏某个成员的集合的大小在用多项式表示集合的协议是不可避免的, 因为多项式的系数的个数决定了集合的大小.

4 安全和效率分析

4.1 安全性

4.1.1 协议 1 的安全性证明

**定理 2.** 假设叠加密方案具有语义安全性, 协议 1 在被动敌手模型中安全计算了判定性集合包含关系问题, 在一些特殊情况下, 协议向一方泄漏了另一方的集合可能包含的元素.

证明.

- (1) 正确性. 根据协议步骤可知,  $A$  在协议中得

到一个用  $A$  的密钥加密的密文

$$c^{Fin} = E_{pk_A}(f(b_1)f(b_2)\dots f(b_{N_B})),$$

$f$  是表示  $S_A$  的多项式,  $f(a) = 1$  当且仅当  $a \in S_A$ , 因此如果  $f(b_1)f(b_2)\dots f(b_{N_B}) = 1$ , 那么就表明  $b_i \in S_A, i = 1, \dots, N_B$ , 也即  $S_A \supseteq S_B$ . 如果  $f(b_1)f(b_2)\dots f(b_{N_B}) \neq 1$ , 那么就必定有某个  $b_i \notin S_A, i \in \{1, \dots, N_B\}$ , 也即  $S_A$  不包含  $S_B$ . 当  $S_A$  不包含  $S_B$  时, 出现  $f(b_1)f(b_2)\dots f(b_{N_B}) = 1$  的情况的概率可忽略.

(2) 隐私性. 正如前面所提到的, 在一些特殊情况下,  $A$  可以推断出  $B$  的集合可能包含什么元素. 下面主要分析一般情况下协议是否保护了成员输入的隐私性.

基于仿真的证明方法的主要思路是: 根据敌手掌握的输入和输出信息, 构造一个仿真器来仿真协议的运行, 使得仿真的结果跟实际协议运行中成员的所见计算不可区分, 也就意味着协议满足安全性的定义.

如果  $A$  是敌手, 仿真器 Simulator 以  $A$  的输入  $S_A$  和输出为输入 (Simulator 的隐式输入为  $A$  的密钥对,  $B$  的公钥), 仿真步骤如下:

Simulator 构造表示  $S_A$  的多项式  $f, f(a) = 1$  当且仅当  $a \in S_A$ . 如果  $A$  的输出结果为 1, 那么 Simulator 在  $S_A$  中选择  $N_B$  个元素构成集合  $S'_B$ ; 如果  $A$  的输出结果为 0, 那么 Simulator 在全集中选择  $N_B$  个元素, 构成集合  $S'_B$  (若  $S'_B$  是  $S_A$  的子集, 则重新选择直到  $S_A$  不包含  $S'_B$ ). 相应地,  $S'_B$  的元素记为  $b'_i$ .

Simulator 计算一个初始值  $\hat{c}^* = E_{pk_A, pk_B}^{sup, B}(1, E_{pk_A}(1))$ . 对于  $j = 1, \dots, N_B - 1$ :

仿真第  $j.1$  步.  $A$  ( $A$  是由 Simulator 仿真的) 计算  $\hat{c} = D_{sk_A}^{sup}(\hat{c}^*)$ , 然后  $A$  计算  $N_A + 1$  个密文:  $\hat{c}_i^* = E_{pk_A, pk_B}^{sup, A}(\alpha_i, \hat{c})$ ,  $i = 0, \dots, N_A$ .  $A$  把叠加密密文  $\hat{c}_0^*, \dots, \hat{c}_{N_A}^*$  发送给  $B$  ( $B$  也是由 Simulator 仿真的).

仿真第  $j.2$  步. Simulator 计算  $f(b'_1)\dots f(b'_j)$ , 然后用  $A$  的公钥加密得到  $\hat{c}' = E_{pk_A}(f(b'_1)\dots f(b'_j))$ , 然后计算  $\hat{c}^* = E_{pk_A, pk_B}^{sup, B}(1, \hat{c}')$ . 并把  $\hat{c}^*$  发送给  $A$ . 如果  $j = N_B$ , 那么 Simulator 不计算  $\hat{c}^*$  而是计算  $\hat{c}^{Fin} = \hat{c}'E_{pk_A}(0)$ , 并把  $\hat{c}^{Fin}$  发送给  $A$ .

仿真输出结果. Simulator 解密  $\hat{c}^{Fin}$  得到  $s = D_{sk_A}(\hat{c}^{Fin})$ . 如果结果为 1, 则令  $\hat{o} = 1$ ; 否则令  $\hat{o} = 0$ .

仿真的结果. Simulator 输出  $S_A$ 、第  $j.2$  步中的消息  $\hat{c}^*$  ( $j = 1, \dots, N_B - 1$ ) 以及  $\hat{c}^{Fin}$  和  $\hat{o}$ .

很明显, 仿真中的  $S_A$  和  $\hat{o}$  跟实际协议运行中的  $A$  的输入、输出相同, 又由叠加密的语义安全性可推出: 仿真中每一步中计算出的密文  $\hat{c}^*$  与实际协议运行中每一步产生的密文  $c^*$  是计算不可区分的. 因

为 Simulator 是根据实际协议中  $A$  的输出来构造仿真中  $B$  的集合  $S'_B$ , 因此仿真中的密文  $c^{\text{Fin}}$  是对正确结果的加密, 并且跟实际协议运行中的密文  $c^{\text{Fin}}$  是计算不可区分的. 可见, 仿真的结果与  $A$  在实际协议中看到的消息是计算不可区分的.

对于  $B$  是敌手的情况,  $B$  在协议中可以看到的消息包括  $N_B(N_A+1)$  个叠加密文, 仿真器的构造很简单, 仿真器只要构造  $N_B(N_A+1)$  叠加密文就可以了, 由于叠加密方案的语义安全性, 这些密文与实际协议运行中  $B$  收到的密文是计算不可区分的.

综上所述, 可见无论  $A$  是敌手还是  $B$  是敌手, 都可以构造仿真器来仿真协议的运行, 仿真的结果跟敌手在协议实际运行中收到的消息是计算不可区分的, 这就意味着, 敌手在协议运行中所能获得的额外消息都可以根据其输入和输出自己计算出来, 也即除了隐含在输入和输出中的信息外, 敌手不能从协议运行中获得更多的信息. 证毕.

#### 4.1.2 协议 2 的安全性证明

**定理 3.** 假定加法同态加密方案具有语义安全性, 协议 2 在被动敌手模型中安全计算了判定性集合包含关系问题, 在一些特殊情况下, 协议向一方泄漏了另一方的集合可能包含的元素.

证明.

(1) 正确性. 根据协议步骤可知,  $A$  在协议中得到一个密文

$$c = E_{pk_A}(f(b_1)f(b_2)\cdots f(b_{N_B})),$$

$f$  是表示  $S_A$  的多项式,  $f(a)=1$  当且仅当  $a \in S_A$ , 因此如果  $f(b_1)f(b_2)\cdots f(b_{N_B})=1$ , 那么就表明  $b_i \in S_A, i=1, \dots, N_B$ , 也即  $S_A \supseteq S_B$ . 如果  $f(b_1)f(b_2)\cdots f(b_{N_B}) \neq 1$ , 那么就必定有某个  $b_i \notin S_A, i \in \{1, \dots, N_B\}$ , 也即  $S_A$  不包含  $S_B$ .

(2) 隐私性. 与协议 1 类似, 协议 2 在特殊情况下使得  $A$  可以推断出  $B$  的集合, 下面主要讨论一般情况下协议 2 的安全性.

如果  $A$  是敌手, 仿真器 Simulator 以  $A$  的输入、输出为输入 (Simulator 的隐式输入为  $A$  的密钥对,  $B$  的公钥), 仿真步骤如下:

Simulator 构造表示  $S_A$  的多项式  $f, f(a)=1$  当且仅当  $a \in S_A$ . 如果  $A$  的输出结果为 1, 那么 Simulator 在  $S_A$  中选择  $N_B$  个元素构成集合  $S'_B$ ; 如果  $A$  的输出结果为 0, 那么 Simulator 在全集中选择  $N_B$  个元素, 构成集合  $S'_B$  (若  $S'_B$  是  $S_A$  的子集, 则重新选择直到  $S_A$  不包含  $S'_B$ ). 相应地,  $S'_B$  的元素记为  $b'_i$ .

由于仿真中用到的  $S_A$  跟实际协议中  $A$  的输入相同, 因此仿真过程中多项式  $F(b_1, \dots, b_{N_B})$  的系数跟实际协议中的一样. 仿真步骤如下:

1. Simulator 把  $F(b_1, \dots, b_{N_B})$  的系数的密文  $E(cf[1]), E(cf[2]), \dots, E(cf[\lfloor \frac{N_A+N_B}{N_B} \rfloor])$  发给  $B$ ;

2. Simulator 计算

$$E(F(b'_1, \dots, b'_{N_B})) = (E(cf[1]))^{e'_1} (E(cf[2]))^{e'_2} \cdots (E(cf[\lfloor \frac{N_A+N_B}{N_B} \rfloor]))^{e'_r} \left( \frac{N_A+N_B}{N_B} \right),$$

并把这个密文发给  $A$ .

3. Simulator 解密密文, 如果明文为 1, 则令  $\hat{o}=1$ ; 否则令  $\hat{o}=0$ .

4. Simulator 输出  $S_A, E(F(b'_1, \dots, b'_{N_B}))$  以及  $\hat{o}$ .

很显然, 仿真的结果中  $S_A$  和  $\hat{o}$  跟实际协议中  $A$  的输入和输出相同. 由于加密方案的语义安全性, 仿真中的密文  $E(F(b'_1, \dots, b'_{N_B}))$  跟实际  $A$  在协议中可以看到的消息  $c = E_{pk_A}(f(b_1)f(b_2)\cdots f(b_{N_B}))$  是计算不可区分的. 可见, 仿真的结果和  $A$  在协议中的所见是计算不可区分的.

对于  $B$  是敌手的情况,  $B$  在协议中可以看到的消息包括  $\left( \frac{N_A+N_B}{N_B} \right)$  个密文 (用  $A$  的公钥加密的), 仿真器的构造很简单, 仿真器只要构造  $\left( \frac{N_A+N_B}{N_B} \right)$  个密文就可以了, 由于加密方案的语义安全性, 这些密文与实际协议运行中  $B$  收到的密文是计算不可区分的.

综上所述, 可见无论  $A$  是敌手还是  $B$  是敌手, 都可以构造仿真器来仿真协议的运行, 仿真的结果跟敌手在协议实际运行中收到的消息是计算不可区分的. 证毕.

#### 4.1.3 协议 3 的安全性证明

**定理 4.** 假定加法同态加密方案具有语义安全性, 协议 3 在被动敌手模型中安全计算了判定性集合包含关系问题.

证明.

(1) 正确性. 根据协议步骤,  $A$  得到一个密文,

$$c^* = (\alpha_0 + \alpha_1(b_1 + b_2 + \cdots + b_{N_B}) + \cdots + \alpha_{N_A}(b_1^{N_A} + b_2^{N_A} + \cdots + b_{N_B}^{N_A})) \text{random} \\ = (f(b_1) + \cdots + f(b_{N_B})) \text{random},$$

$f$  是表示  $S_A$  的多项式,  $f(a)=0$  当且仅当  $a \in S_A$ , 因此如果  $(f(b_1) + \cdots + f(b_{N_B})) \text{random} = 0$ , 那么  $b_i \in S_A, i=1, \dots, N_B$ , 也即  $S_A \supseteq S_B$ . 当  $(f(b_1) + \cdots + f(b_{N_B})) \text{random} \neq 0$  时,  $S_A$  不包含  $S_B$  的概率可忽略. 如果  $(f(b_1) + \cdots + f(b_{N_B})) \text{random} \neq 0$ , 则必定有某

个  $b_i \notin S_A, i \in \{1, \dots, N_B\}$ , 也即  $S_A$  不包含  $S_B$ .

(2) 隐私性. 证明同定理 2 隐私性证明类似, 略.

证毕.

## 4.2 效 率

**协议 1.** 需要  $N_B$  轮通信, 通信量为  $O(kN_A N_B)$  比特, 这里  $k$  是叠加密方案的安全参数.  $A$  做  $O(N_B)$  次解密、 $O(N_A N_B)$  次加密;  $B$  做  $O(N_A N_B)$  次解密、 $O(N_A N_B)$  次加密.

**协议 2.** 需要一轮通信, 通信量为  $O(k \binom{N_A + N_B}{N_B})$  比特, 这里  $k$  是所用加密方案的安全参数.  $A$  做  $O(\binom{N_A + N_B}{N_B})$  次加密、一次解密;  $B$  做  $O(\binom{N_A + N_B}{N_B})$  次加密.

**协议 3.** 需要一轮通信, 通信量为  $O(kN_A)$  比特, 这里  $k$  是所用加密方案的安全参数.  $A$  做  $O(N_A)$  次加密、一次解密;  $B$  做  $O(N_A)$  次加密.

## 5 叠加密的替代方案

叠加密方案的算法形式比较复杂, 计算量也比一般加密方案大, 可以用更简单的加密方案——带门限解密的加法同态加密方案代替叠加密方案. 设  $E$  是加密算法,  $A$  和  $B$  各自拥有解密密钥的一部分, 解密时双方各自向对方发送一个解密值, 从两个解密值可以恢复出明文. 基于门限加密方案的协议如下:

$A$  计算一个初始值  $c = E(1)$ . 对于  $j = 1, \dots, N_B$ :

第  $j.1$  步.  $A$  计算  $N_A + 1$  个密文:  $c_i^* = c^{a_i}, i = 0, \dots, N_A$ .  $A$  把密文  $c_0^*, \dots, c_{N_A}^*$  发送给  $B$ .

第  $j.2$  步.  $B$  计算  $c^* = E(0) c_0 (c_1)^{b_j} \dots (c_{N_A})^{b_j^{N_A}}$ .  $B$  把  $c^*$  发送给  $A$ .

输出结果.  $A$  和  $B$  联合解密第  $N_B$  步产生的密文  $c^*$  得到  $s = D(c^*)$ . 如果  $s = 1$ , 则  $A$  和  $B$  输出 1 (表示  $S_A$  包含  $S_B$ ); 否则  $A$  和  $B$  输出 0 (表示  $S_A$  不包含  $S_B$ ).

上述基于门限加密方案的协议安全性证明与协议 1、协议 2 的安全性证明类似, 略.

门限加密方案的加密算法比叠加密方案的加密算法简单, 计算量小, 因此, 跟协议 1 比, 上面的协议在计算效率上有提高. 同时由于门限解密需要一方给另一方发送解密值 (如果双方都要得到计算结果的话则需要双方交换解密值), 在输出结果阶段需要多一轮通信. 文献[5]中的判断集合是否不相交的协议 PIPE #1 也可以改造成基于门限加法同态加密方案的协议.

以加法同态 ElGamal 加密方案为例, 加密消息  $m$ :  $E(m) = (g^r, y^r h^m)$ , 需要两个模幂运算; 解密密文  $c = (G, H)$ :  $m = H/G^x$ , 需要一个模幂运算. 叠加密算法加密一个消息  $m$  和一个密文  $c = (G, H)$ :  $E_{pk_A, pk_B}^{sup, A}(m, (G, H)) = (g^r, G^m g^{r'}, y_X^r y_X^{r'} H^m)$ , 需要 3 个模幂运算, 解密密文  $c = (G_A, G_B, Y)$ , 双方的解密算法在计算量上跟 ElGamal 的解密算法一样, 都是一个模幂运算.

协议 1 第  $j$  轮,  $A$  做一次解密和  $N_A + 1$  次叠加密, 需要的计算量为  $3N_A + 4$  次模幂运算,  $B$  做  $N_A + 1$  次解密和  $N_A + 1$  次叠加密, 需要的计算量为  $3(N_A + 1)$  次模幂运算, 每一轮共  $6(N_A + 1) + 1$  次模幂运算.

基于普通加法同态 ElGamal 方案的协议第  $j$  轮,  $A$  做  $N_A + 1$  次加密, 需要  $2(N_A + 1)$  次模幂运算,  $B$  做  $N_A + 1$  次加密, 需要  $2(N_A + 1)$  次模幂运算, 每一轮共  $4(N_A + 1)$  次模幂运算.

所以, 改造后的协议比协议 1 节省了  $2N_B(N_A + 1)$  次模幂运算. 此外, 叠加密的密文长度是普通 ElGamal 密文的 1.5 倍, 因此改造后的协议交换的消息量是协议 1 的  $2/3$ .

需要注意的是, 在获知计算结果阶段, 改造后的协议需要成员交互一次, 交换解密值或者一方给另一方发送解密值. 因此改造后的协议比协议 1 多了一轮通信.

## 6 结 论

本文讨论了判断集合包含关系的问题, 给出了 3 个具有不同效率的协议. 协议的基本思想是利用多项式来表示集合, 从而通过计算多项式在集合各元素处的值来判断两个集合是否具有包含关系. 协议 1 基于叠加密方案或支持门限解密的加法同态加密方案, 协议 2 和协议 3 基于普通的加法同态加密方案. 在半诚实模型中, 采用基于仿真的方法证明了协议的安全性.

文献[5]指出, 协议 PIPE #2 和 PIPE #3 可以转化为具有主动安全性的协议, 本文的协议 1 和协议 2 分别与 PIPE #2 和 PIPE #3 类似, 因此采用文献[5]提到的技术手段, 协议 1 和协议 2 也可以转化为具有主动安全性的协议. 根据 Goldreich<sup>[8]</sup>指出的, 任何被动安全的协议都可以转化为主动安全的协议, 协议 3 也可转化为具有主动安全性的协议.

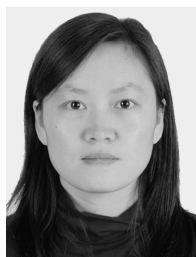


## 参 考 文 献

- [1] Yao A C. Protocols for secure computation//Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science. Chicago, Illinois, 1982; 160-164
- [2] Goldreich O, Micali S, Wigderson A. How to play any mental game or a completeness theorem for protocols with honest majority//Proceedings of the 19th Annual ACM Symposium on Theory of Computing. New York, USA, 1987; 218-229
- [3] Kissner L, Song D. Privacy-preserving set operations//Proceedings of the 25th Annual International Cryptology Conference (CRYPTO 2005). Santa Barbara, California, USA, 2005, LNCS 3621; 241-257
- [4] Freedman M J, Nissim K, Pinkas B. Efficient private matching and set intersection//Proceedings of the EUROCRYPT 2004. Interlaken, Switzerland, 2004, LNCS 3027; 1-19
- [5] Kiayias A, Mitrofanova A. Testing disjointness of private

datasets//Proceedings of the 9th International Conference on Financial Cryptography and Data Security (FC'05). Roseau, Dominica, 2005, LNCS 3570; 109-124

- [6] Li Shun-Dong, Si Tian-Ge, Dai Yi-Qing. Secure multi-party computation of set-inclusion and graph-inclusion. Journal of Computer Research and Development, 2005, 42(10): 1647-1653(in Chinese)  
(李顺东, 司天歌, 戴一奇. 集合包含与几何包含的多方保密计算. 计算机研究与发展, 2005, 42(10): 1647-1653)
- [7] Yao A C. How to generate and exchange secrets//Proceedings of the 27th IEEE Symposium on Foundations of Computer Science. Toronto, Ontario, Canada, 1986; 162-167
- [8] Goldreich O. Foundations of Cryptography: Volume II, Basic Applications. Beijing: Publishing House of Electronics Industry (Originated from Cambridge University Press), 2005
- [9] Goldwasser S, Micali S. Probabilistic encryption. Journal of Computer and System Sciences, 1984, 28; 270-299



**LI Rong-Hua**, born in 1977, Ph.D.. Her research interests include cryptography and cryptographic protocols.

**WU Chuan-Kun**, born in 1964, professor, Ph. D. supervisor. His research interests include cryptography and network security, electronic-commerce, and information hiding etc.

**ZHANG Yu-Qing**, born in 1966, professor. His current research interests include network and system security and cryptography.

## Background

Secure Multiparty computation is that  $n$  distrustful parties, with each holding a secret input, want to compute an agreed function of their inputs. But they don't wish to leak anything else except the outputs of the function. A protocol for multiparty computation should satisfy correctness and privacy. That is to say, the results of the computation should be correct and the privacy of parties' inputs should be protected even if an adversary corrupts some of the parties. The set inclusion problem can be seen as a special case of secure multiparty computation problem. Thus, the set inclusion problem can be solved using the general protocols for secure multiparty computation. However, general protocols are not very efficient.

Li, Si, and Dai studied the set inclusion problem, and gave a solution to it. Actually, their protocol computes the cardinality intersection problem, not the set inclusion problem. That is to say, parties compute the size of the intersection set, and then decide if one set includes the other by comparing the sizes of the intersection and the sizes of their sets. Kissner and Song also studied the set inclusion problem. They presented a protocol based on threshold homomorphic encryption scheme. In their protocol,  $S_A$  is represented as a polynomial  $f(x)$ , such that  $f(a)=0$  if and only if  $a \in S_A$ . The set  $S_A$  includes the set  $S_B$  if  $\sum f(S_B[i])=0$ , where

$S_B[i]$  denotes the  $i$ th element of  $S_B$ .

This paper proposes three protocols for the set inclusion problem. The first protocol can be based on superposed encryption or threshold encryption. The second protocol is a modified version of the first protocol, which can be based on the traditional homomorphic encryption (in contrast to superposed encryption and threshold encryption). These two protocols use a set representation method different from the one that Kissner and Song's protocol uses. Here,  $S_A$  is represented as a polynomial  $f(x)$  such that  $f(a)=1$  if and only if  $a \in S_A$ , and  $S_A$  includes  $S_B$  if  $\prod f(S_B[i])=1$ . Such a method of representing sets may be of independent use. The third protocol uses the same method as Kissner and Song's protocol; however, it is based on homomorphic encryption instead of threshold encryption, and is more efficient.

This paper is supported by the National Natural Science Foundation Nos. 60573048, 60773135. The project No. 60573048 is about survivability of networks. The project No. 60773135 is about P2P trust management. The group has some results on security vulnerabilities, vulnerability rating methods, survivability of networks, etc. Secure multiparty computation is a part of these projects, and considers cryptographic protocols that are required to protect privacy of inputs.