**BGW安全多方计算算法——乘法门**

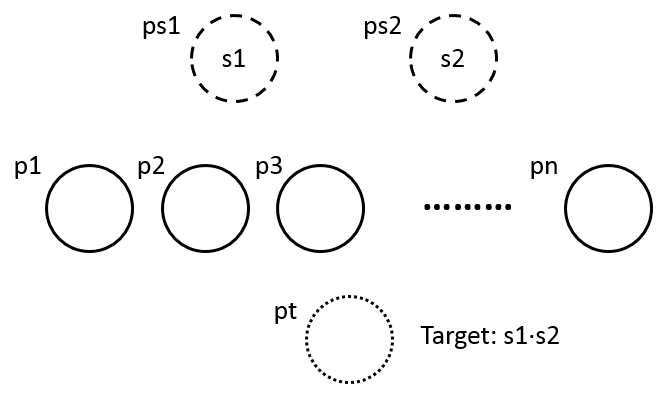
# 场景

对于两个秘密拥有者实体ps1与ps2，两者分别拥有秘密*s1*与*s2*。此时某第三方*pt*希望获得两个秘密的乘积*s1s2*。这个过程需要一组参与实体*p*1, *p*2, …, *pn*协助进行秘密分享与计算，并且整个过程满足以下条件：

* 第三方*pt*获得两个秘密的乘积*s1s2*。
* 在*p*1, *p*2, …, *pn*中不超过t人共同作恶的前提下，*pt*外其余所有参与实体均不能获得任何关于乘积*s1s2*的信息。
* 在*p*1, *p*2, …, *pn*中不超过t人共同作恶的前提下，仅ps1能知道*s1*的值，仅ps2能知道*s2*的值。

其中，秘密拥有者实体ps1与ps2或者第三方*pt*可以包括在参与实体*p*1, *p*2, …, *pn*之中，只要其符合逻辑上的关系（譬如ps1与*pt*不能为同一人，否则其就能轻易推断出不能被其知道的*s2*）。

本质上，这是一个二元安全多方计算的乘法门（拥有两个输入与一个输出，三者符合乘法关系）。拥有之前的线性安全多方计算（加法与常数乘法）与这里的二元安全多方计算乘法门，理论上任何仅包含乘法与加法的函数都能被顺利多方计算出来。



# 前提：Shamir秘密分享

为了进行安全多方计算的乘法门，首先要采用Shamir的秘密分享方案，将两个秘密*s1*与*s2*分为*n*份交给参与实体*p*1, *p*2, …, *pn*。这一部分先阐述关于Shamir秘密分享的内容，在后面章节会用到。

## Shamir的秘密分享方案

Shamir的秘密分享方案是一个(*k*, *n*)门限秘密分享方案，即将秘密*D*分解为*n*个分片，使得：

1. 拥有任意*k*个或更多的分片，可以快速计算*D*；
2. 仅拥有*k* - 1个或更少的分片，*D*完全不可判断，对于所有取值概率相同。

方案描述如下：

1. 方案基于一个*k* – 1次一元多项式，该多项式在有限域*Fp*上，其中*p*是素数，且*p* > *D*，多项式如下：

*q*(*x*) = *d*0 + *d*1*x* + … + *dk*-1*xk*-1 mod *p*

1. 分享秘密：
   1. 随机选择*k*-1个随机数*d*1, *d*2, …, *dk*-1*Fp*，令*d*0 = *D*，构建上述多项式；
   2. 计算*q*(1), *q*(2), …, *q*(*n*)，构造*n*个分片(*i*, *q*(*i*))，其中1≤*i*≤*n*。
   3. 这里b)中的多项式变量*x*取值为1到*n*，实际也可以是任意*n*个*Fp*内的互不相等的随机数如，是等价的。
2. 通过*k*个分片还原秘密：*k*个分片，就有*k*个线性方程，未知数是*d*0, *d*1, …, *dk*-1，通过解方程可得所有未知数，其中*d*0就是还原出来的秘密*D*。

# 安全多方计算乘法门基本流程概览

为了计算多方计算乘法门，基本的步骤可以归结如下：

1. 准备阶段：各方共识一个素数*p*，以及一组随机数。
2. 秘密分发阶段： 两个秘密拥有者实体ps1与ps2采用Shamir的秘密分享方案，将两个秘密*s1*与*s2*分为*n*份交给参与实体*p*1, *p*2, …, *pn*。也即是，若ps1与ps2在Shamir秘密分享中采用的多项式分别为*t*次多项式*q*1(*x*)与*q*2(*x*)，则*p*i获得秘密分片*q1(ai)*与*q2(ai)*。
3. 协商阶段： 参与实体*p*1, *p*2, …, *pn*协商一个*2t*次且常数项为0的多项式*φ(x)*，每个实体*p*i仅拥有此多项式的一个点*φ(ai)*。
4. 降次阶段：参与实体*p*1, *p*2, …, *pn*分别计算*u(ai)= φ(ai)+ q1(ai) q2(ai)*。之后，利用*n*次线性安全多方计算进行矩阵乘法，从而最终使得每个实体获得常数项为*s1s2*的*t*次多项式*tu(x)*上的一个点*tu(ai)*的值（详见4.4）。
5. 输出阶段：所有*n*个实体*p*1, *p*2, …, *pn*发向第三方*pt*发送*tu(ai)*，*pt*就可以利用其中任意*t*+1个*tu(x)*上的点，运行Shamir的秘密还原算法，计算出最终结果即两个秘密的乘积*s1s2*。

# 二元乘法门计算

指*f*(*s*1, *s*2) = *s*1*s*2，即一个二元安全多方计算的乘法门。

## 准备阶段

**所有实体共识一个素数*p*，以及一组随机数。**

这里一提，素数*p*的取值应当大于计算结果可能的最大值。

## 秘密分发阶段

初始时，对于两个秘密拥有者实体*ps1*与*ps2*，两者分别拥有秘密*s1*与*s2*。

**对于任意*psi*，随机选择一个*t*次多项式*qi*(*x*) = *si* + *qi*1*x + qi*2*x*2+…+ *qitxt* mod *p*，其中。**

**每个秘密拥有者实体*psi*计算*qi*(*a*1)，*qi*(*a*2)，…，*qi*(*an*)，并将*qi*(*ak*)发给实体*pk*(1≤*k*≤*n*)；**

此时每个实体*pi*拥有的信息是*q*1(*ai*), *q*2(*ai*)。经此一步，相当于两个秘密*s1*与*s2*已经被分为了*n*份给*p*1, *p*2, …, *pn*。

## 协商阶段

协商阶段的最终目的与结果：参与实体*p*1, *p*2, …, *pn*共识一个*2t*次且常数项为0的多项式*φ(x)*，且每个人*p*i仅拥有此多项式的一个点*φ(ai)*。

实际操作具体流程如下：

**对于*p*1, *p*2, …, *pn*中任意一个实体，随机选择一个*2t*次的多项式，且此多项式的常数项必须为0。**也即是，对于实体*pi*，其需要随机选择一个形式为如下的多项式*φi(x)*：

*φi(x)= ri*1*x + ri*2*x*2+…+ *ri,2tx2t* mod *p*，其中。

**接下来，对于每一实体*pi*，其需要计算*φ*i(*a1*), *φ*i(*a2*), …, *φi*(*an*)，并且将*φi*(*ak*)分发给*pk* (1≤*k*≤*n*)。**

此时每个实体*pi*拥有的信息是*φ*1(*ai*), *φ*2(*ai*), …, *φn*(*ai*)。**不妨定义，那么显然*pi*就可以计算出*φ(ai)***，计算公式显然是：

值得注意的是，*φ(x)*是一个常数项为0的*2t*次的多项式，因为每一个*φi(x)*都是常数项为0的*2t*次的多项式。并且，*φ(x)*的每个系数都是满足在域内均匀分布的。

协商阶段完毕后每个实体*pi*拥有的信息：*φ(ai)*，*q*1(*ai*)，*q*2(*ai*)。

## 降次阶段

首先，参与实体*p*1, *p*2, …, *pn*分别计算*u(ai)= φ(ai)+ q1(ai) q2(ai)*。

值得注意的是，*u(x)*是一个常数项为两个秘密的乘积*s1s2*的*2t*次的多项式，因为*φ(x)*是一个常数项为0的*2t*次的多项式，而*qi*(*x*)是一个常数项为*si*的*t*次多项式。

此时，每个实体*pi*拥有的信息：*u(ai)*。

目前*u(x)*是一个常数项为两个秘密的乘积*s1s2*的*2t*次的多项式，我们需要将其降低为一个*t*次多项式*tu(x)*，且*t*次及其以下的系数保持不变。即，假设目前*u(x)*可以如下表示：

*u(x)= s1s2 + u1x + u*2*x*2+…+ *u2tx2t* mod *p*，其中

那么降次后*t*次多项式*tu(x)*应当为：

*tu(x)= s1s2 + u*1*x + u*2*x*2+…+ *utxt* mod *p*

我们希望实体*pi*能将手中的*u(ai)*替换为*tu(ai)*，如此一来当*pt*得到*t*+1个*tu(ai)*，其中(1≤*i*≤*n*)时，*pt*只需再通过Shamir的秘密还原方式，即可得到*s1s2*的值。所有*tu(ai)*的具体计算方式如下：

定义对应于随机数的Vander-monde矩阵如下所示：

并且定义为长度为*n*的一个向量，并且其中*u0= s1s2*。那么显然有：

由于可逆，则有：

同理若定义为长度为*n*的一个向量，并且其中*u0= s1s2*，那么同理显然有：

而如果构造矩阵，其中：

也就有：，那么上述式子代入即可得到：

如果我们记矩阵，那么代入上式，显然对于任意(1≤*k*≤*n*)都有：

矩阵是一个常数矩阵，*u(ai)*则是每个实体*pi*独有的信息，需要计算出*u(ai)*们的加权和*tu(ak)*，这正是线性安全多方计算的模式。因此我们可以通过*n*次的线性安全多方计算，使得每个实体*pk*计算出*tu(ak)*。这部分内容请参见《线性安全多方计算算法》，实现时调用相关子模块即可。

下表列出在计算*tu(ak)*的过程中，相关的计算对应关系。

|  |  |
| --- | --- |
| **“线性安全多方计算”中的变量** | **本文中的实际变量** |
| 参与实体*p*1, *p*2, …, *pn* | 参与实体*p*1, *p*2, …, *pn* |
| 各*pi*的秘密*xi* | *u(ai)* |
| 最终结果接收者（输出阶段接收β者） | 实体*p*k |
| 计算结果*f*(*x*1, *x*2, …, *xn*) | *tu(ak)* |
| *f*(*x*1, *x*2, …, *xn*)中各常数项 | 矩阵B中第*k*行的常数项 |

上述过程一共进行*n*次(1≤*k*≤*n*)，即每一次计算出一个*tu(ak)*给实体*pk*。

**降次阶段总结：**

**1. 参与实体*p*1, *p*2, …, *pn*分别计算*u(ai)= φ(ai)+ q1(ai) q2(ai)*。此时，每个实体*pi*拥有的信息：*u(ai)*。**

**2. 所有实体分别本地计算矩阵B，其中。**

**3. 通过*n*次的线性安全多方计算，使得每个实体*pk*计算出*tu(ak)*。这部分内容请参见《线性安全多方计算算法》，实现时调用相关子模块即可。**

**4. 降次阶段结束时，每个实体*pi*拥有的信息：*tu(ai)*。**

## 输出阶段

输出阶段中，**所有*n*个实体*p*1, *p*2, …, *pn*发向第三方*pt*发送*tu(ai)*，*pt*就可以利用其中任意*t*+1个*tu(x)*上的点，运行Shamir的秘密还原算法，计算出最终结果即两个秘密的乘积*s1s2*。**

所有*tu(ai)*，其中(1≤*i*≤*n*)得到之后，我们相当于得到了一个常数项为*s1s2*的*t*次多项式*tu(x)*上的*n*个点。因此我们只需要采用Shamir的秘密还原方案，利用多项式*tu(x)*中任意*t+1*个点，即可还原该多项式。

具体而言，可以理解为该*t*次多项式所有的*t+1*个系数为*t+1*个未知数，利用*t+1*个点即可构成关于这些未知数的线性方程组，解出这些系数。而这些系数中的常数项，即为我们所需要的*s1s2*。

# 补充说明

这里有额外的一些说明：

## 多元乘法与后续计算

本质上，降次阶段中最后获得的，本质上也就是将*s1s2*乘积进行Shamir的(*t+1,n*)门限切分后的结果。这些秘密分片分别属于实体*p*1, *p*2, …, *pn*，正好可以进行后续进一步的安全计算。

例如：假设需要计算*f*(*s*1, *s*2, *s*3) = *s*1*s*2*s*3，那么可以先对秘密*s1*与*s2*进行乘法多方计算，降次阶段之后，就得到了*s1s2*乘积的(*t+1,n*)门限秘密分享结果（即上面的各个*tu(ai)*），*n*个分片分别属于实体*p*1, *p*2, …, *pn*。

此时，我们因为不需要知道*s*1*s*2的结果，而想要知道*s*1*s*2*s*3的结果，因此我们将**不进行输出阶段，而只让*n*个实体先暂存各个*tu(ai)***。

取输出阶段而代之的，是将***s*3进行(*t+1,n*)门限Shamir秘密分享**。即，秘密*s*3拥有者实体ps3随机选取常数项为*s*3的*t*次多项式*q3(x)*，并将*n*个分片*q3*(*a*1)，*q3*(*a*2)，…，*q3*(*an*)分配给实体*p*1, *p*2, …, *pn*。**此时每个实体*p*i拥有*q3(ai)*与*tu(ai)*，只需要将这两者分别看作4.2章“秘密分享阶段”结束时的*q*1(*ai*)与*q*2(*ai*)，进而从协商阶段开始继续完成整个乘法流程即可。**

又例如：假设需要计算*f*(*s*1, *s*2, *s*3) = *s*1*s*2+*s*3，同样思路，先多方计算属于*s*1*s*2的*n*个分片（到降次阶段停止），之后再分发秘密*s*3，然后每个实体*p*i将拥有的*q3(ai)*与*tu(ai)*直接相加，则每个人就得到了一个常数项为*s*1*s*2+*s*3的*t*次多项式的一个分片，之后再进行输出阶段即可。

## 关于4.4降次阶段中的通信与详细步骤

在降次阶段中，我们需要为每一个实体*p*i计算出*tu(ai)*。其中，对于任意的(1≤*k*≤*n*)，都有：

因此，我们总共需要进行*n*次线性安全多方计算。在《线性安全多方计算算法》中，我们分析过，一次线性安全多方计算的通信复杂度为O(n2)，那么*n*次的复杂度是否就变成O(n3)呢？不是。实际上，通过Shamir秘密数据的复用，总通信复杂度仍旧可以保持在O(n2)。

“*n*次线性安全多方计算”的详细步骤如下：

1. 初始时每个实体*pi*拥有的信息：*u(ai)*。

2. 对每个实体*pk*，随机选择一个常数项为*u(ak)*的*t*次的多项式*qk*(*x*)，并计算*qk*(*a*1)，*qk*(*a*2)，…，*qk*(*an*)，并将*qk*(*aj*)发给实体*pj*(1≤*j*≤*n*)。此时每个实体*pj*拥有的信息是*q*1(*aj*), *q*2(*aj*), …, *qn*(*aj*)。

3. 对于每个实体*pj*，分别计算如下*n*个值，即对1≤*i*≤*n*的所有βij。

此时每个实体*pj*已经拥有βij，其中1≤*i*≤*n*。

4. 实体*pj*将βij发给实体*pi*，此时每个实体*pi*拥有信息为βi1, βi2,…, βin。

5. 每个实体*pi*取βi1, βi2,…, βin中*t*+1份运行Shamir秘密还原，得到的结果就是*tu(ai)*。至此，“*n*次线性安全多方计算”完成，此时每个实体*pi*拥有的信息：*tu(ai)*。