安全内积协议

安全内积协议功能介绍：

Input：

Alice owns

$$X = (x_1, x_2, \ldots, x_n)$$

Bob owns

$$Y = (y_1, y_2, \ldots, y_n)$$

Output：

Alice outputs

$$X \cdot Y = (x_1, x_2, \ldots, x_n) \cdot (y_1, y_2, \ldots y_n) = x_1 * y_1 + x_2 * y_2 + \ldots + x_n * y_n$$

Bob outputs empty;

说明：

此协议用于算术电路中乘法三元组生成。

此协议Alice最终获得内积结果。

乘法三元组介绍：

乘法三元组用于算术电路乘法计算，是独立于算术电路且满足特定关系的一组数字，关系如下：

$$P_0 : a_0, b_0, c_0;$$
$$P_1 : a_1, b_1, c_1;$$
$$\text{其中：}\ a_0 + a_1 = a; b_0 + b_1 = b; c_0 + c_1 = c; \text{且} c = a * b$$

安全内积协议：

1. Alice选择随机数向量

$$R = (r_1, r_2, \ldots, r_n), r_i \in F_p$$

计算

$$Z = X - R = (x_1 - r_1, x_2 - r_2, \ldots, x_n - r_n) = (z_1, z_2, \ldots, z_n), z_i = x_i - r_i \in F_p$$

在选择一个随机数

$$r \in F_p$$

计算

$$W = r * Z = （r * z_1, r * z_2, \ldots, r * z_n）, w_i = r * z_i \in Fp$$

发送（R，W）给Bob。

2. Bob计算

$$u = R \cdot Y = （r_1 * y_1 + r_2 * y_2 + \ldots + r_n * y_n）, u \in F_p$$

计算

$$v = W \cdot Y = (w_1 * y_1 + w_2 * y_2 + \ldots + w_n * y_n), v \in F_p$$

发送（u，v）给Alice。

3. Alice计算

$$u + v/r$$

就可以获得

$$X \cdot Y$$

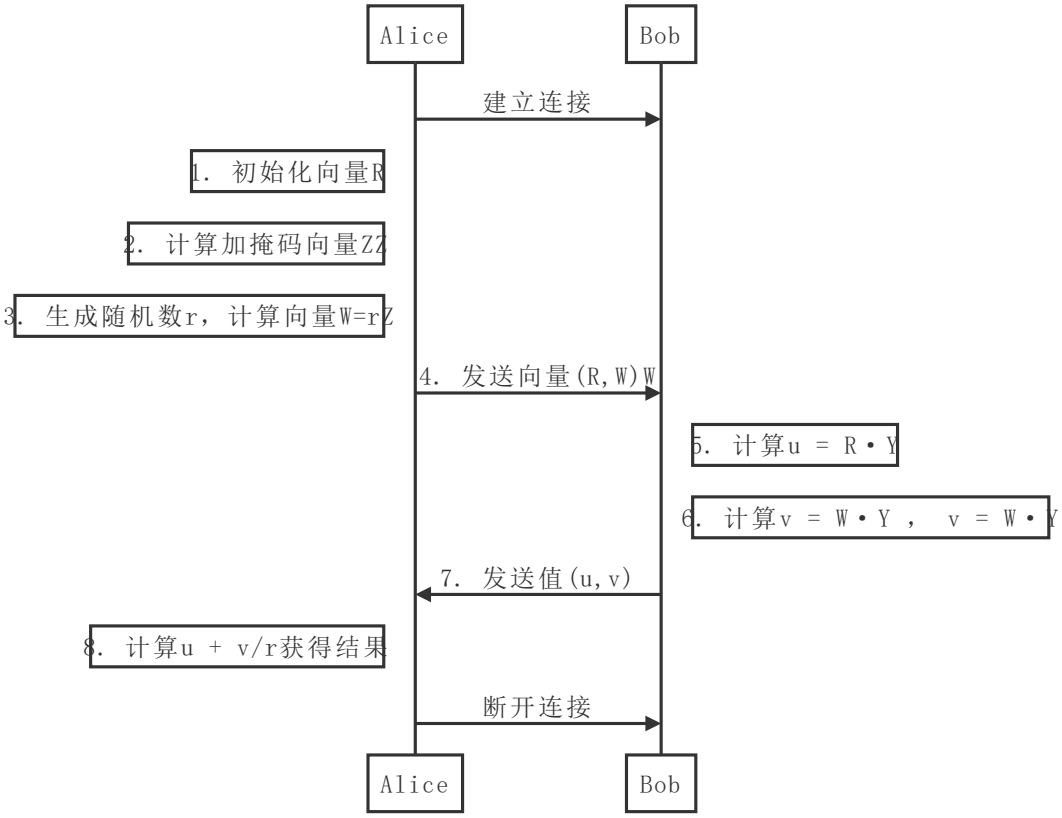
使用安全内积协议构造乘法三元组：

$$\begin{gathered}
P_0 : a_0, b_0; \\
P_1 : a_1, b_1, c_1; \\
c_0 = (a_0 + a_1)(b_0 + b_1) - c_1 \\
= a_0 b_0 + a_0 b_1 + a_1 b_0 + a_1 b_1 - c_1 \\
= (1, b_0, a_0, a_0 b_0, -1)(a_1 b_1, a_1, b_1, 1, c_1)
\end{gathered}$$

然后使用安全内积协议。

流程定义：

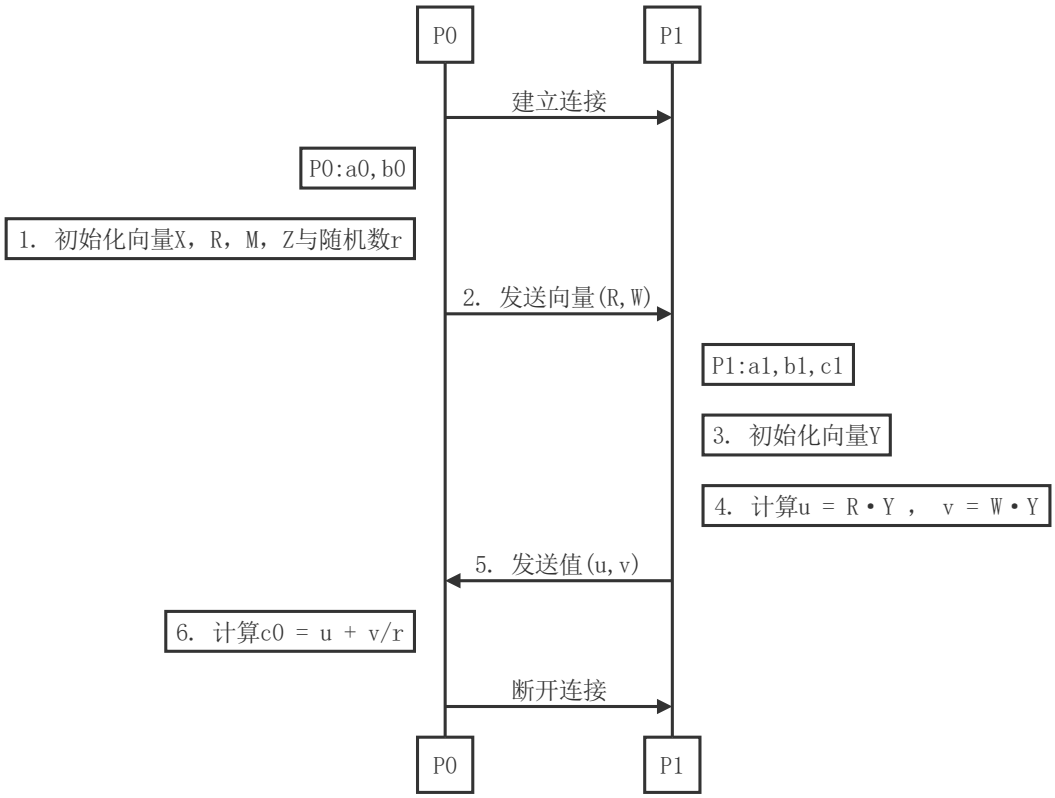
安全内积协议流程图：



流程描述：

1. 初始化生成掩码向量R。
2. 计算加掩码向量Z。
3. 生成随机数r，计算向量W=rZ。
4. 发送（R，W）。
5. Bob计算u = R·Y。
6. Bob计算v = W·Y。
7. 发送值(u，v)。
8. Alice计算u+v/r即可获得结果。

利用该协议协商乘法三元组流程图：



流程描述：

1. 初始化向量：

$$X = (1, b_0, a_0, a_0b_0, -1)$$
$$R = (r_1, r_2, r_3, r_4, r_5) \quad r_i \in F_p$$
$$Z = X - R = (1 - r_1, b_0 - r_2, a_0 - r_3, a_0b_0 - r_4, -1 - r_5) = (z_1, z_2, \ldots, z_5), z_i = x_i - r_i \in F_p$$
$$random \quad r \quad (r^{-1} \in F_p)$$
$$W = r * Z = (r * z_1, r * z_2, \ldots, r * z_n), w_i = r * z_i \in Fp$$

2. 发送向量R与W。

3. 初始化向量：

$$Y = (a_1b_1, a_1, b_1, 1, c_1)$$

4. 计算值u，v：

$$u = R \cdot Y = (r_1 * a_1b_1 + r_2 * a_1 + r_3 * b_1 + r_4 * 1 + r_5 * c_1) \quad u \in Fp$$
$$v = W \cdot Y = (w_1 * a_1b_1 + w_2 * a_1 + w_3 * b_1 + w_4 * 1 + w_5 * c_1) \quad v \in Fp$$

5. 发送值u与v。

6. P1计算c0：

$$c_0 = u + r^{-1}v \quad c_0 \in Fp$$

接口定义：

位置1：

```
ABY/src/abycore/sharing/arithsharing.cpp
    void ArithSharing<T>::InitMTs()
    //此处代码主要是初始化乘法三元组值，需要初始化赋值。

接口定义：
InitMTs(); //初始化乘法三元组参数值
```

位置2：

```
ABY/src/abycore/aby/abysetup.h
    //此位置为定义结构体，重新定义新结构体

结构体定义：
struct MTsGenVals {
    CBitVector* A;
    CBitVector* B;
    CBitVector* C;
    uint32_t numMTs;
    uint32_t sharebitlen;
}
```

位置3：

```
ABY/src/abycore/sharing/arithsharing.cpp
    void ArithSharing<T>::PrepareSetupPhase(ABYSetup* setup)
    //此处修改结构体赋值

MTsGenVals* pgentask = (MTsGenVals*) malloc(sizeof(MTsGenVals));
    pgentask->A = &(m_vA[0]);
    pgentask->B = &(m_vB[0]);
    pgentask->C = &(m_vC[0]);
    pgentask->numMTs = m_nMTs;
    pgentask->sharebitlen = m_nTypeBitLen;
    setup->AddPKMTGenTask(pgentask);
```

位置4：

```
/ABY/src/abycore/DJN/djnparty.cpp
    //此处代码是计算乘法三元组。
接口定义
    void DJNParty::computeArithmeticMTs(BYTE * A, BYTE * B, BYTE * C, BYTE * A1,
BYTE * B1, BYTE * C1, uint32_t numMTs, channel* chan)  //计算乘法三元组
```