



北京邮电大学

Beijing University of Posts and Telecommunications

安全多方计算简介

石瑞生

网络空间安全学院

保密声明 (Confidential Statement)



- 本课件属于北京邮电大学专有机密，所有权属于北京邮电大学网络空间安全学院大数据安全教学团队。
- 课件所涉及的内容和资料只限于同学进行内部查看使用，禁止外传至任何渠道，包括但不限于网络。一经发现，必究责任！！！！
- 收到本PPT后，收件人或知情人应遵守以下规定：
 1. 本PPT仅限本人使用，禁止网络私自传播，否则后果自负。
 2. 在没有取得北邮网安院大数据安全教学团队负责人同意前，收件人或知情人不得将本PPT全部或部分地予以复制、传递给他人，影印、泄漏或散步给他人。
 3. 北京邮电大学享有追究刑事责任的权利。

- 1. 概念 & 模型
 - 示例：百万富翁问题
- 2. 基础协议介绍
 - 2.1 GC
 - 2.2 OT
 - 2.3 SS
- 3. 应用与挑战
 - 3.1 安全多方计算通用框架
 - 3.2 性能问题 & 安全问题

安全多方计算的概念与模型

什么是安全多方计算（Secure Multi-Party Computation）？

安全多方计算是**无可信第三方**的保护隐私计算协议。

安全多方计算（Secure Multi-Party Computation, MPC），主要研究在**无可信第三方**的情况下，多个参与者如何**安全地计算一个约定函数**的问题。

安全多方计算（Secure Multi-party Computing, SMC）有多个参与方，**每一个参与方拥有自己的秘密信息**，他们希望利用**这些秘密信息作为输入**，共同计算一个函数。

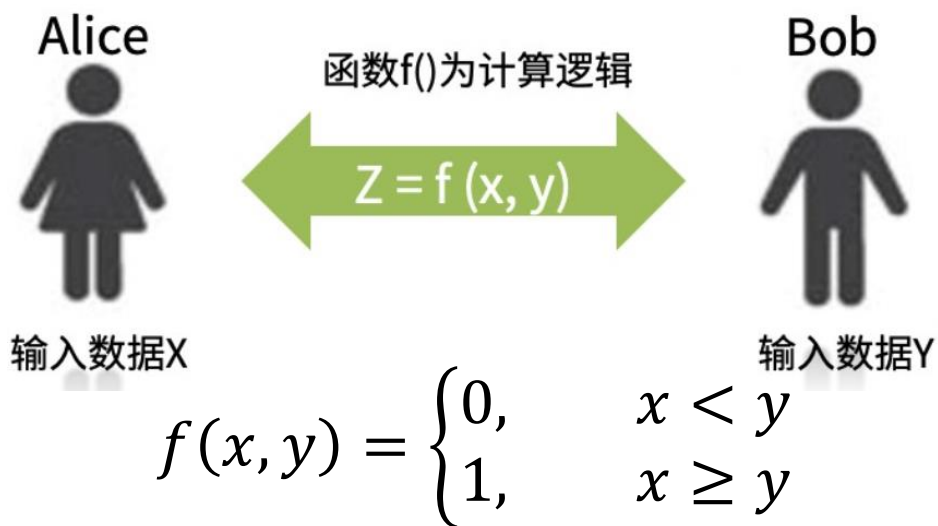
安全多方计算的一般概念：**m个参与方**希望共同计算一个函数 **$f(x_1, x_2, \dots, x_m)$** ，其中 x_i 是第i个参与方所掌握的私有输入，安全多方计算使得函数f可以被正确计算（**正确性**），同时每个参与方不能得知其他参与者的私有输入（**安全性**）。

例如，一个协会希望知道协会成员的平均收入，每个人又不希望泄漏自己的收入信息。通过安全多方计算技术，即实现了数据的共享，又保护了参与方的隐私信息，是大数据服务中实现安全与隐私保护的有力工具。

起源



- 安全多方计算问题最早是由著名的计算机科学家、2000年图灵奖获得者姚期智教授提出的，即百万富翁问题。
 - 1982年，姚期智首先介绍了安全多方计算的概念，并提出了著名的百万富翁问题。



在没有第三方参与的情况下，两个百万富翁能够在互相不暴露自己的财产数额的情况下，比较谁更富有。

1986年，姚期智提出了解决**两方安全计算**通用解决方案：姚氏混淆电路。

随后，Goldreich, Micali（2012图灵奖得主），Wigderson对该问题进行了推广，提出了**多方**安全计算协议**GMW协议**，可以用来计算任意函数。

1988年，Goldwasser, Chaum等人从理论上证明了安全多方计算的可解性。（BGM88, CCD88）

安全多方计算的理论研究主要以以色列学者Goldreich等人的工作为主，研究工作已经得到了一般安全多方计算问题都是可解的结论。

- 安全多方计算模型有两种：半诚实模型和恶意模型。
 - 半诚实模型：如果所有参与者都是诚实的或者半诚实的，称此模型为半诚实模型，其中的攻击者是被动的。
 - 恶意模型：存在恶意参与者的模型称为恶意模型，其中的攻击者是主动的。
 - 隐蔽的攻击模型：上述的半诚实对手模型过于脆弱，但恶意对手模型在安全性要求下导致协议效率太低。为了克服这些困难，提出了一种隐蔽敌手模型。
 - 一个隐蔽的对手可能表现出恶意行为，但它有一定的概率被诚实的参与者发现作弊。
 - 这一模型代表了许多金融或政治环境，在这些环境中，诚实的行为是不可能假设的，但所涉及的公司和机构不能承受与被发现作弊有关的名誉损失。在这个模型中，对手必须权衡被抓住的风险和作弊的好处。

- 参与者模型

- 诚实参与者：在协议执行过程中，诚实参与者完全按照协议的要求完成协议的各个步骤，同时保密自己的所有输入、输出及中间结果。
- 半诚实参与者：在协议执行过程中，半诚实参与者完全按照协议的要求完成协议的各个步骤，但同时~~可能将自己的输入、输出及中间结果泄漏给攻击者~~，也可以根据自己的输入、输出及中间结果推导其他参与者的信息。
- 恶意参与者：在协议执行过程中，恶意参与者完全按照攻击者的意志执行协议的各个步骤，不但将自己的所有输入、输出及中间结果泄露给攻击者，还可以根据攻击者的意图改变输入信息、中间结果信息，甚至终止协议。

- 攻击者模型

- 攻击者是指企图破坏协议安全性和正确性的人。对攻击者进行分类时，可以有不同的分类准则，这些分类准则主要有攻击者的计算能力、网络同步状态、对恶意参与者的控制程度和动态性。
- 按照计算能力分类：按照攻击者的计算能力可以将攻击者分为拥有无限计算能力的攻击者和拥有有限计算能力的攻击者。
 - 对于拥有无限计算能力的攻击者而言，不存在诸如大素数分解困难等数学难题。在无限计算能力的攻击者模型下的安全的多方计算协议为**信息论安全**的多方计算协议。
 - 对于拥有有限计算能力的攻击者而言，破解目前公认的数学难题是不可能的。在有限计算能力的攻击者模型下的安全的多方计算协议为**密码学安全**的多方计算协议。

百万富翁问题

- 问题

- 在没有第三方参与的情况下，两个百万富翁能够在互相不暴露自己的财产数额的情况下，比较谁更富有。

- 解决方案

1982 年,姚期智教授提出的百万富翁协议是解决百万富翁问题的最早方案。在该协议中,假设 Alice 的秘密输入为 a , Bob 的秘密输入为 b , 满足 $1 \leq a < b \leq n$ (原协议的要求是 $1 \leq a < b \leq 10$, 为了保证通用性, 这里假设 $1 \leq a < b \leq n$, n 是大于 1 的某个正整数)。令 M 是所有 N bit 非负整数的集合, Q_N 是所有从 M 到 M 的一一映射函数的集合。令 E_A 是 Alice 的公钥, 它是从 Q_N 中随机抽取的。该协议的具体描述如下。

输入: Alice 有一个秘密输入 a , Bob 有一个秘密输入 b 。

输出: Alice 和 Bob 得到 a 和 b 的大小关系。

百万富翁问题 – 解决方案

- ① Bob 随机选取一个 N bit 的整数 x , 秘密计算 $E_A(x)$ 的值, 并把该值记为 k 。
- ② Bob 将 $k-b+1$ 发送给 Alice。
- ③ Alice 秘密地计算 $y_u = D_A(k-b+u)$ 的值 ($u=1, 2, \dots, n$)。
- ④ Alice 产生一个 $N/2$ bit 的随机素数 p , 对所有 u 计算 $z_u = y_u \pmod p$ 。如果所有的 z_u 在模 p 运算下至少相差 2, 则停止。否则, 重新产生一个随机素数 p 重复上面的步骤, 直到所有的 z_u 至少相差 2。用 $p, z_u (u=1, 2, \dots, n)$ 表示最终产生的这些数。
- ⑤ Alice 将素数 p 以及下面的 n 个数都发送给 Bob (下列数都在模 p 运算下):
$$z_1, z_2, \dots, z_a, z_{a+1} + 1, \dots, z_n + 1$$
- ⑥ Bob 检验由 Alice 传送过来的不包括 p 在内的第 b 个值, 若它等于 $x \pmod p$, 则 $a \geq b$, 否则 $a < b$ 。
- ⑦ Bob 把结论告诉 Alice。

百万富翁问题 – 分析

正确性分析

$$z_1, z_2, \dots, z_a, z_{a+1} + 1, \dots, z_n + 1$$

上述协议能正确判断出 a 和 b 的大小关系, 因为

$$z_u = D_A [E_A(x) - b + u] \bmod p$$

特别地, 有

$$z_b = D_A [E_A(x) - b + b] \bmod p = D_A [E_A(x)] \bmod p = x \bmod p$$

如果 $a \geq b$, 那么第 b 个值为 $z_b = x \bmod p$, 否则为 $z_b + 1 = (x \bmod p) + 1 \neq (x \bmod p)$ 。所以, Bob 通过检验由 Alice 传送的不包括 p 在内的第 b 个值, 可以判断 a 和 b 的大小。

安全性分析

协议能够保证 Alice 和 Bob 都不能得到有关对方财富的更多的信息。

首先, 除了当 Bob 告诉 Alice 最后的结论后, Alice 能够推测出 b 的范围以外, Alice 将不知道 Bob 财富的任何信息, 因为她从 Bob 那里仅仅得到一个值 $k - b + 1$, 由于 k 的存在, 使 Alice 不能从中得知 b 。

其次, Bob 知道 y_b (即 x) 的值, 因此他也知道 z_b 的值。然而他不知道其他 z_u 的值, 而且通过观察 Alice 发送给他的数列, 他也无法辨认出哪个是 z_u , 哪个是 $z_u + 1$ 。这一点是由两两 z_u 之间至少相差 2 来保证的。

解决方案的局限性

但是,协议仍然存在一些可以被攻击的地方。例如,Bob 有可能选择随机数 t ,并验证 $E_A(t) = k - b + n - 1$ 是否成立。如果他尝试成功了,他便知道 $y_{n-1} = t$,从而他也得知了 z_{n-1} 的值。因此他能够得知 a 是否大于等于 $n-1$ 。而且,在协议的最后一步,Bob 可能欺骗 Alice,告诉 Alice 一个错误的结论。所以,该协议不能对抗主动攻击,即不能使用在恶意模型下。

另外,该协议的复杂度是指数级的,效率非常低,不实用。但是,该协议的设计思想是简单的,实际上它主要将 a 在一个有序数中的位置做了隐藏的“标记”,在 a 之前的数使用一种标记,在 a 之后的数使用另外一种标记,然后让 b 揭示它所在位置的“标记”,如果是前一种,则 b 比 a 小,否则 b 比 a 大。由于简单的设计原理,该协议成为百万富翁协议中较为经典的一个。



阅读材料 - MPC



- 1) 百万富翁问题: Andrew C. Yao, **Protocols for Secure Computations**, 1982
- 2) Yao, Andrew Chi-Chih. "**How to generate and exchange secrets**." In *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pp. 162-167. IEEE, 1986. 【GC】
- 3) Goldreich, O., S. Micali, and A. Wigderson. 1987. "How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority". In: *19th Annual ACM Symposium on Theory of Computing*. Ed. by A. Aho. ACM Press. 218–229. 【GMW】

OT

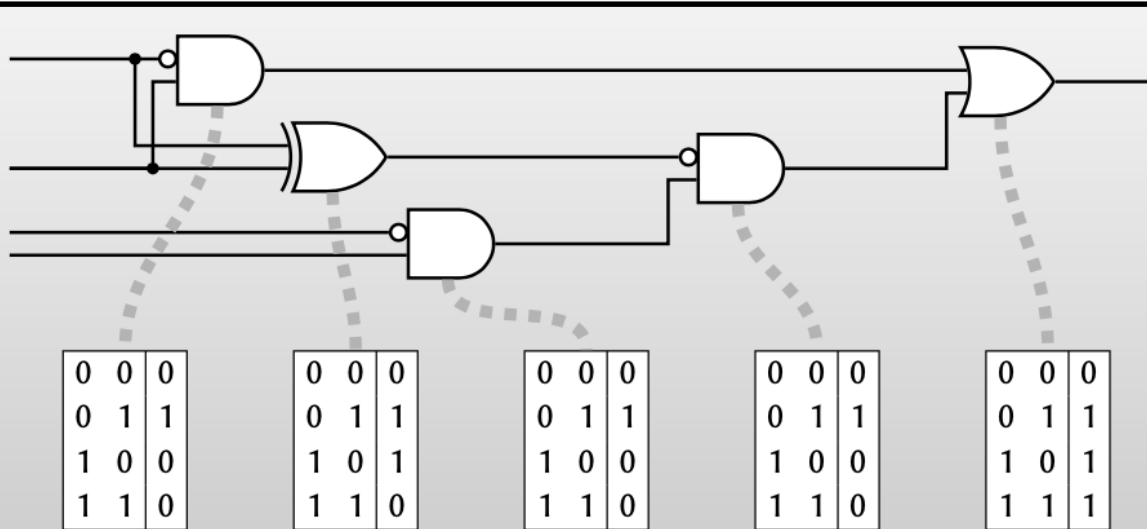
- 1) Rabin M O . How to Exchange Secrets by Oblivious Transfer[J]. Technical Memo TR-81, 1981.
- 2) Even, Shimon, Oded Goldreich, and Abraham Lempel. "**A randomized protocol for signing contracts**." *Communications of the ACM* 28, no. 6 (1985): 637-647.
- 3) Beaver, Donald. "Correlated pseudorandomness and the complexity of private computations." In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pp. 479-488. 1996.
- 4) Ishai Y , Kilian J , Nissim K , et al. **Extending Oblivious Transfers Efficiently**[C]// 23rd Annual International Cryptology Conference. CiteSeer, 2003.
- 5) Kolesnikov V , Kumaresan R . Improved OT Extension for Transferring Short Secrets[M]. Springer Berlin Heidelberg, 2013.
- 6) Asharov G , Lindell Y , Schneider T , et al. More efficient oblivious transfer and extensions for faster secure computation[C]// Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM, 2013.

基础MPC协议简介

- 2.1 概述
- 2.2 GC (混淆电路) 的工作原理
- 2.3 OT (不经意传输) 模型与工作原理
- 2.4 SS (秘密共享)

混淆电路 (GC, Garbled Circuit)

- 乱码电路 (GC, Garbled Circuit) 是半诚实模型下的两方安全计算协议, 针对布尔电路。

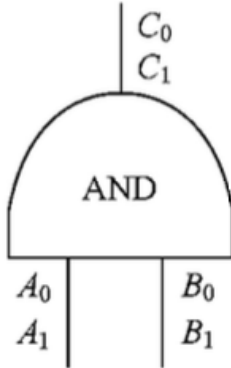


由于对每个功能函数 f , 都存在一个与其等价的电路 C , 因此最早的 Yao 协议^[1] 就是首先将功能函数转化为一个电路, 然后针对电路的每个电路门进行混淆, 最后逐次计算每个混乱门电路来实现对任意功能函数的安全多方计算.

混淆电路 (GC, Garbled Circuit)

- GC协议的两个参与方扮演不同的角色，承担不同的工作

Garbler P1	Evaluator P2
安全多方计算协议：将计算逻辑编译成布尔电路，然后将布尔电路中的每一个门进行加密并打乱加密顺序完成混淆操作，将需要计算的函数f表示为一个混淆电路。	对混淆电路进行解密获取计算结果



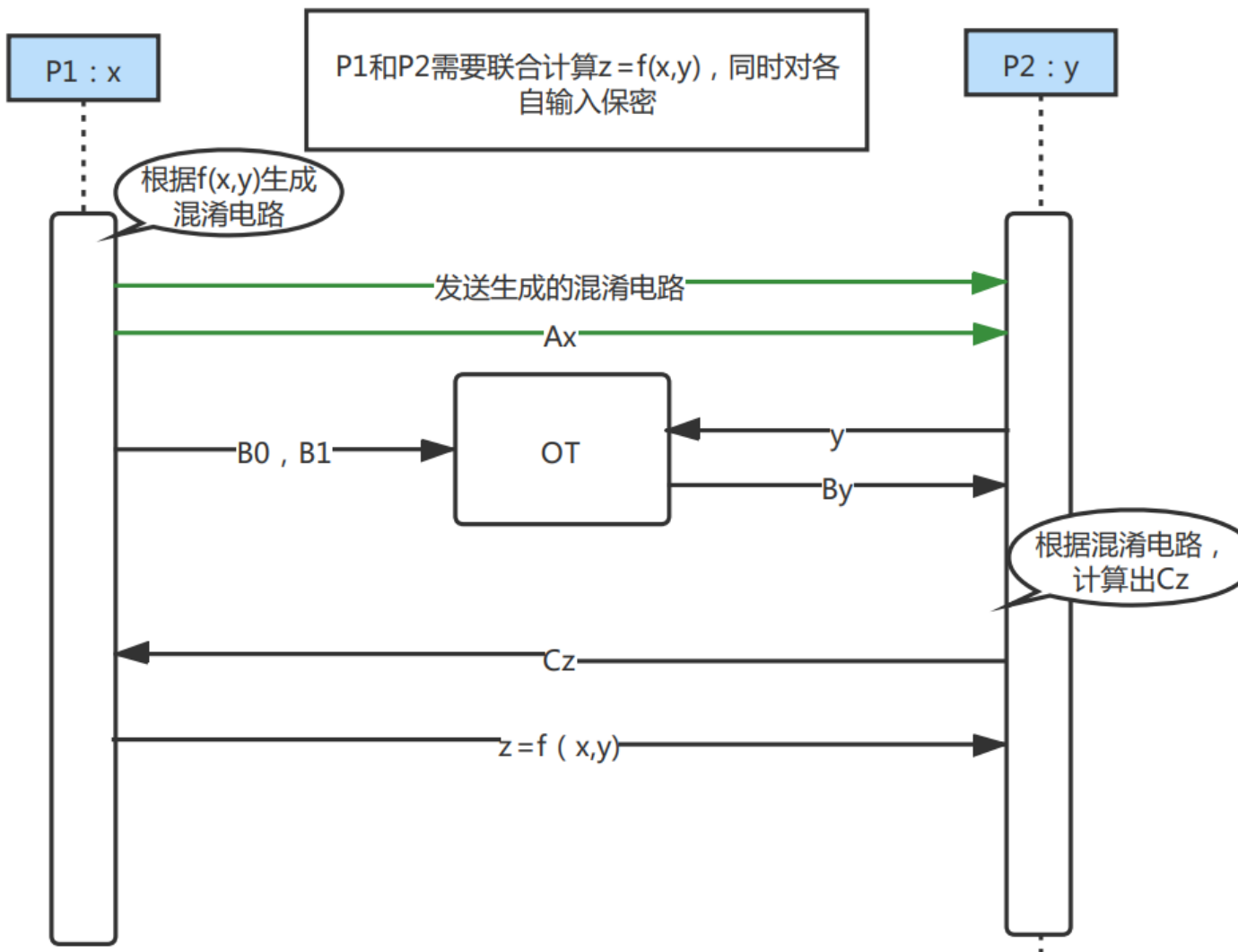
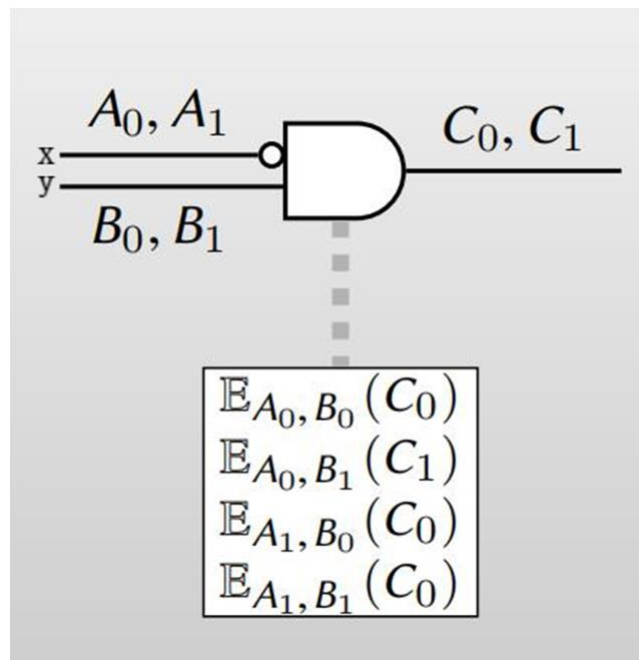
(a) Garbled AND Gate

τ	σ	$\tau\sigma$	Garbled Value
0 A_0	0 B_0	0 C_0	$E_{A_0}(E_{B_0}(C_0))$
0 A_0	1 B_1	0 C_0	$E_{A_0}(E_{B_1}(C_0))$
1 A_1	0 B_0	0 C_0	$E_{A_1}(E_{B_0}(C_0))$
1 A_1	1 B_1	1 C_1	$E_{A_1}(E_{B_1}(C_1))$

(b) Garbled computation table

加密方：将需要计算的函数f表示为一个混淆电路，并发给计算方
 计算方：完成密文电路计算

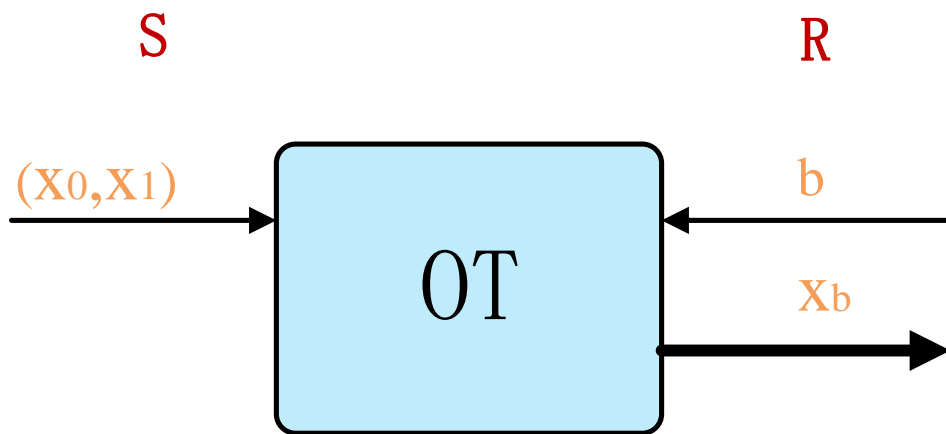
GC协议



OT协议定义及模型

OT 协议：不经意传输（Oblivious Transfer，OT），是安全多方计算（Secure Multi-Party Computation，MPC）的重要构造模块，属于基础协议。发送方将潜在的许多信息发送给一个接收者，发送方无法知道接收方接收到的是那个信息。

参与方：发送方 S \rightarrow 两个秘密 (X_0, X_1)
接收方 R \rightarrow 选择比特 $b \in (0, 1)$ ，收到 X_b



- If $b = 0$, $X_b = X_0$
- If $b = 1$, $X_b = X_1$

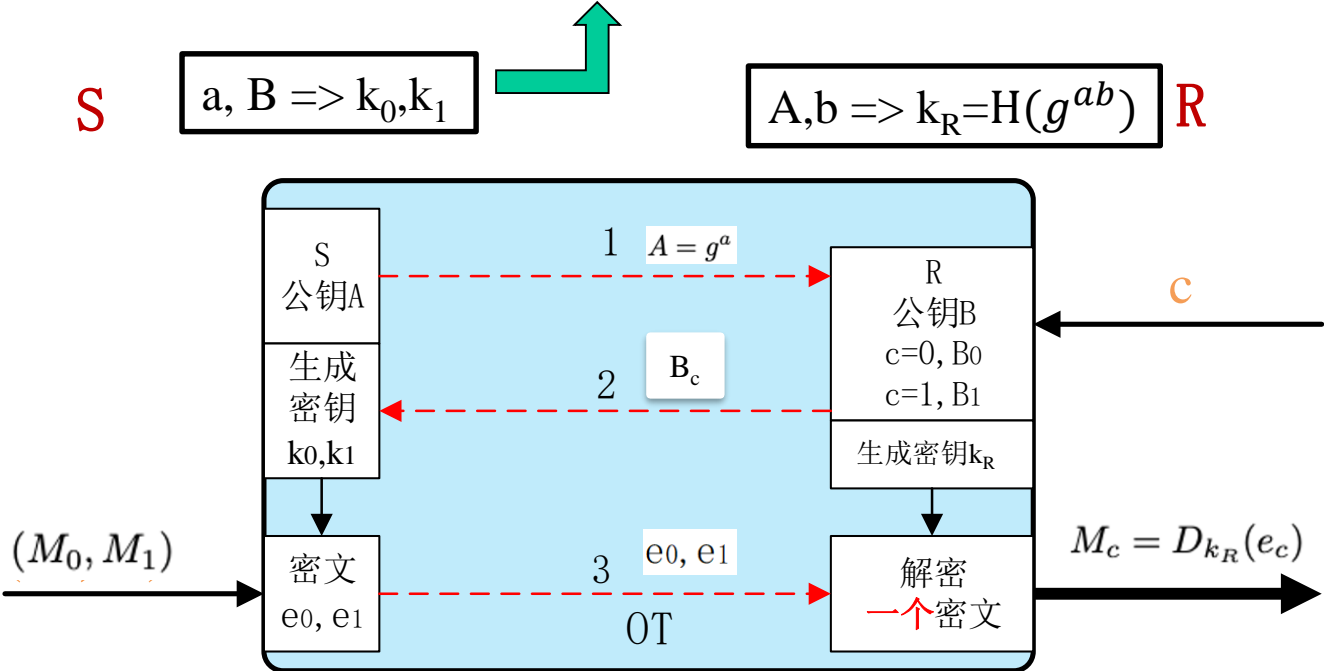
工作原理 - 2选1OT协议



Algorithm 1 基础 2 选 1OT 协议

- 发送方 \mathcal{S} 输入: (M_0, M_1)
接收方 \mathcal{R} 输入: 选择向量 c
共同输入: 素数 p , 生成元 $g \in \mathbb{Z}_p$, 哈希函数 H
- $\mathcal{S} \rightarrow \mathcal{R}$, $a \leftarrow \mathbb{Z}_p$
 - 将 $A = g^a$ 发送给 \mathcal{R} , 由于离散对数难题, \mathcal{R} 无法破译 a
 - $\mathcal{R} \rightarrow \mathcal{S}$, $b \leftarrow \mathbb{Z}_p$
 - 如果 $c = 0, B = g^b$; 如果 $c = 1, B = Ag^b$, 将 B 发送给 \mathcal{S}
 - 计算 $k_R = H(A^b)$
 - \mathcal{S} 计算密钥, 并加密消息, 发送给 \mathcal{R} :
 - $k_0 = H(B^a)$, $e_0 \leftarrow E_{k_0}(M_0)$
 - $k_1 = H((\frac{B}{A})^a)$, $e_1 \leftarrow E_{k_1}(M_1)$
 - \mathcal{R} 解密获得 $M_c = D_{k_R}(e_c)$

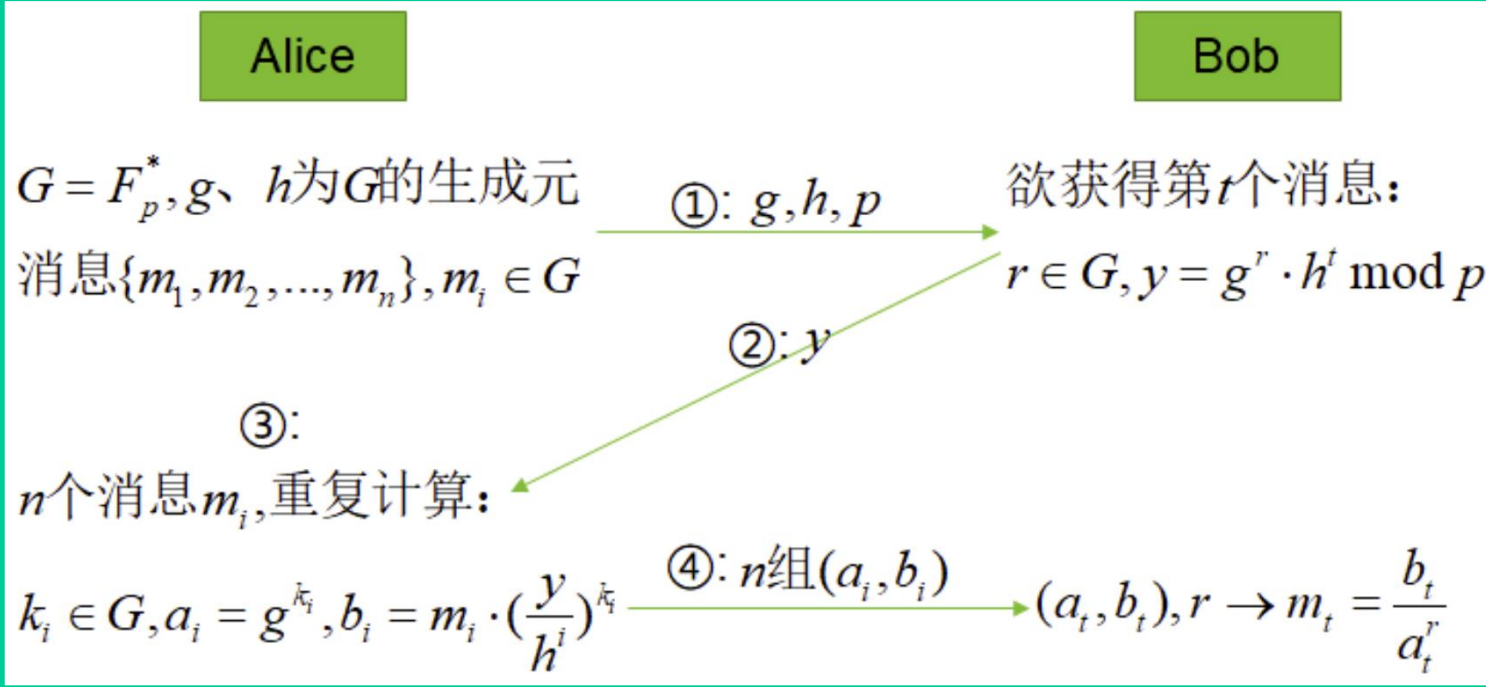
		K_0	K_1
$c=0$	$B_0 = g^b$	$H(B^a) = H(g^{ab})$	$H((\frac{B}{A})^a) = H(g^{(b-a)a}) = H(g^{ab-a^2})$
$c=1$	$B_1 = Ag^b = g^{a+b}$	$H(B^a) = H(g^{(a+b)a})$	$H((\frac{B}{A})^a) = H((g^b)^a) = H(g^{ab})$



$k_R = H(A^b) = H(g^{ab})$
如果 $c = 0$, $k_0 = H(g^{ab})$, $k_1 = H(g^{ab-a^2})$
如果 $c = 1$, $k_0 = H(g^{(a+b)a})$, $k_1 = H(g^{ab})$
由于离散对数难题, \mathcal{R} 只能获得选择向量对应密钥, 并解密获得 M_c 。

基于离散对数实现n选1的OT协议

- 协议执行过程分为4个步骤:
1. Alice有n条消息 $\{m_1, \dots, m_n\}$, $m_i \in G$ (G 代表素数域 F_p^*), 挑选 G 的两个生成元 g 和 h , 将 g 、 h 、 p 发送给Bob。
 2. 假定Bob想获得第 t 条消息, 则Bob随机选择大整数 $r \in G$, 并计算 $y = g^r \cdot h^t \bmod p$, 将 y 发送给Alice。
 3. Alice利用 y 、 g 、 h , 分别对n条消息, 重复执行图中左下角的计算步骤, 每一次执行都会随机选择大整数 $k_i \in G$, 并结合消息 m_i 计算 a_i 和 b_i 。然后将n组 (a_i, b_i) 发送给Bob。
 4. Bob拿到n组 (a_i, b_i) 后, 利用掌握的大整数 r , 计算想要的第 t 条消息 $m_t = b_t / (a_t)^r$ 。



$$\frac{b_x}{(a_x)^r} = \frac{m_x \cdot (\frac{y}{h^{k_x}})^{k_x}}{(g^{k_x})^r} = \frac{m_x \cdot (\frac{g^r \cdot h^t}{h^{k_x}})^{k_x}}{(g^{k_x})^r} = m_x \cdot h^{(t-k_x) \cdot k_x}$$



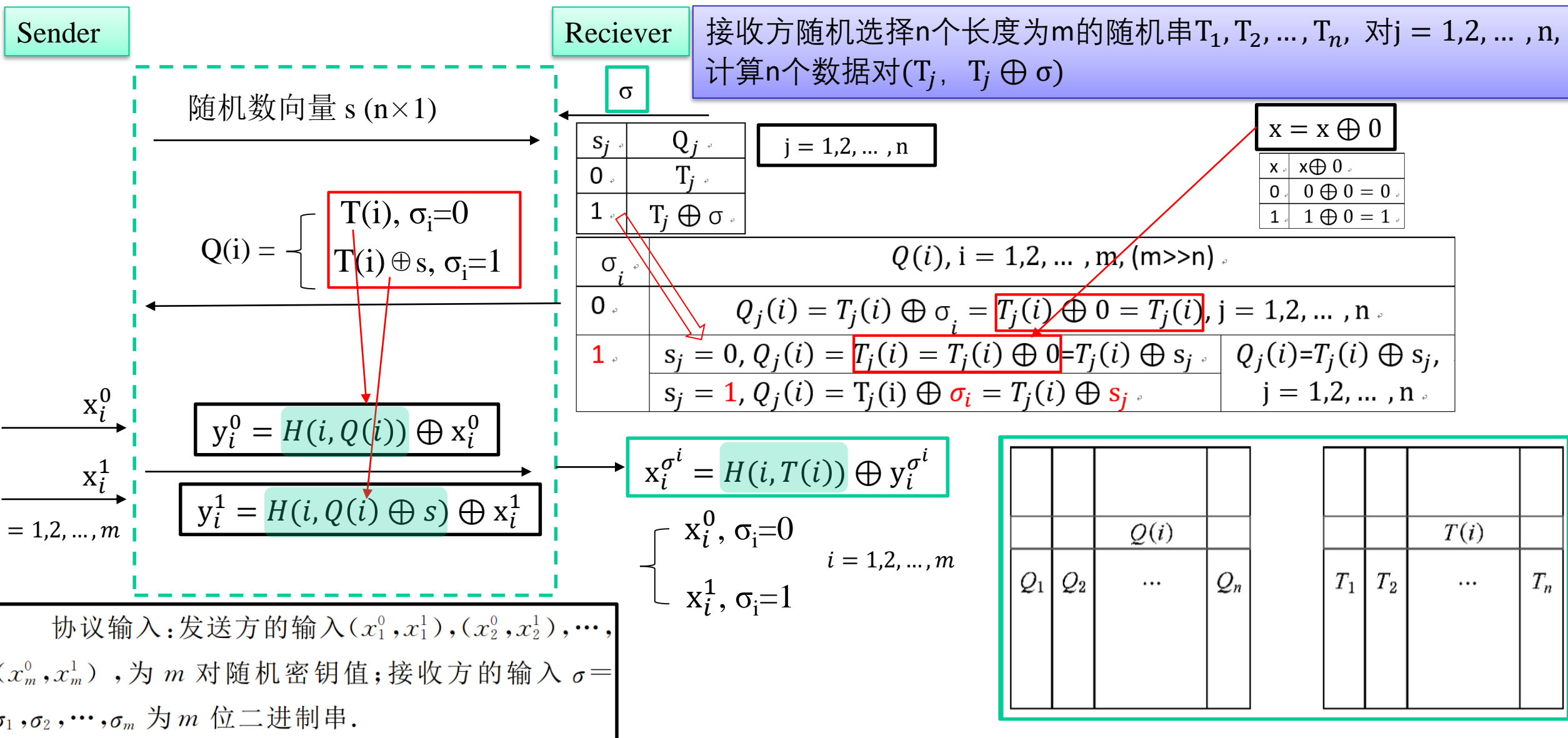
思考：利用OT协议解决百万富翁问题



- OT 协议是所有安全多方计算协议中的最基本工具
 - 在基于Yao混乱电路构造的SMPC协议中, 参与方每bit输入需要一个OT协议,
 - 而在基于GMW范式构造的SMPC协议中, 布尔电路的每一个And门需要至少一个OT协议。
- 在恶意敌手模型下的实际的安全多方计算协议所需要执行的OT次数需要数百万次。
 - 举例来说, 利用TinyOT技术, 当计算AES电路时, 需要使用 2^{19} 次OT协议,
 - 而计算隐私集合求交(private set intersection, PSI)电路时, 需要使用 2^{30} 次OT协议。
- 因此, OT协议执行的效率成为了影响安全多方计算协议效率的最重要的因素。

OT - IKNP[03]

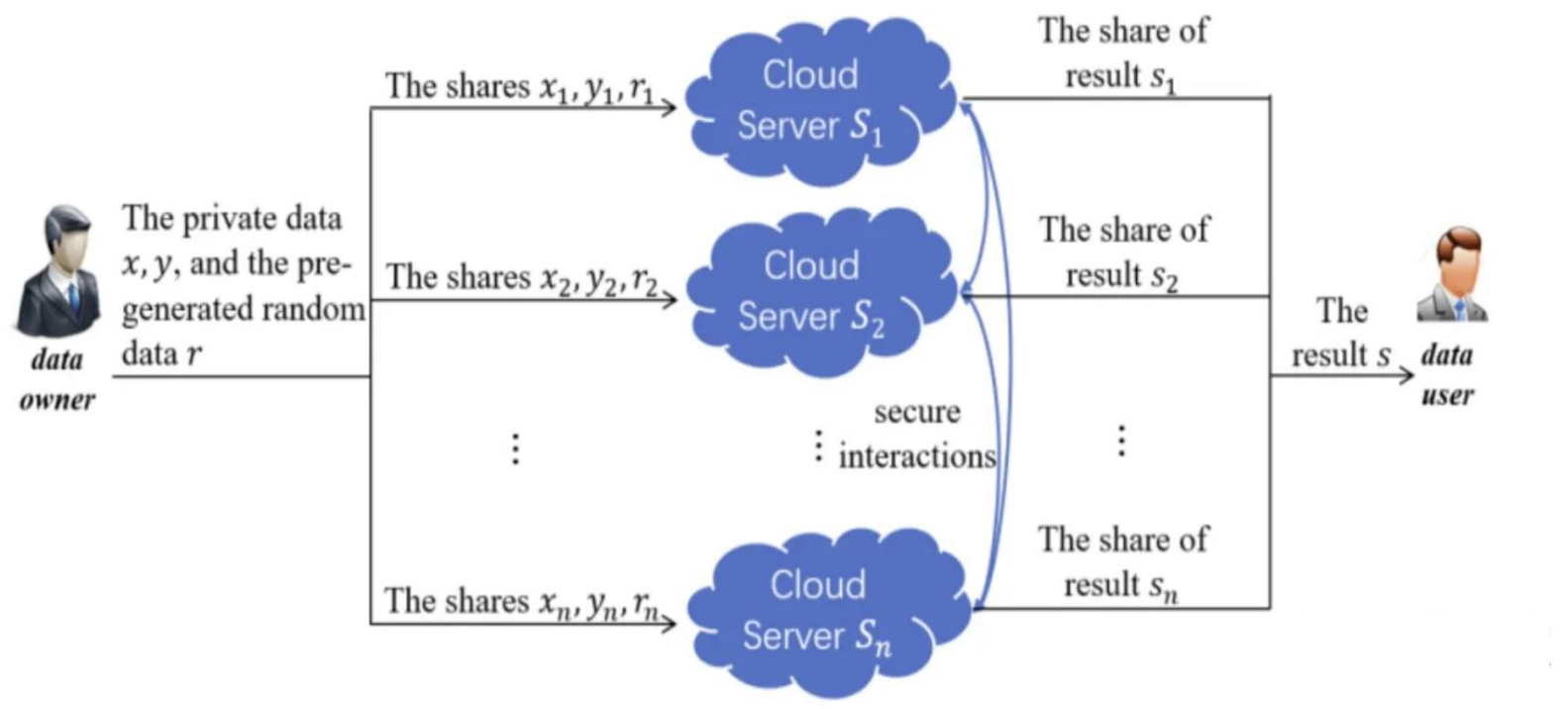
提出了一个半诚实敌手模型下安全的OT扩展协议，是第 1 个高效地将 n 个基础OT协议扩展为 m ($\gg n$) 个OT协议的工作
通过运行少量的OT协议(例如几百个)作为基础，来获得大量(例如几百万个)OT协议执行的效果



秘密共享

Secret Sharing 翻译成中文就是“秘密分享”或者“秘密共享”，通常大家都称为秘密分享。秘密分享是在一组参与者中分享秘密的技术，它主要用于保护重要信息，防止信息被丢失、被破坏、被篡改。它源于经典密码理论，最早由Sharmir和Blakley在1979年提出。简单来说，秘密分享就是指分享的秘密在一个用户群体里进行合理分配，以达到由所有成员共同掌管秘密的目的。可以将秘密分享分为两类：

- 1) 严格的秘密分享：需要所有人一起解密；
- 2) 阈值的秘密分享：不需要所有人，只需要满足一定人数，就可以解密

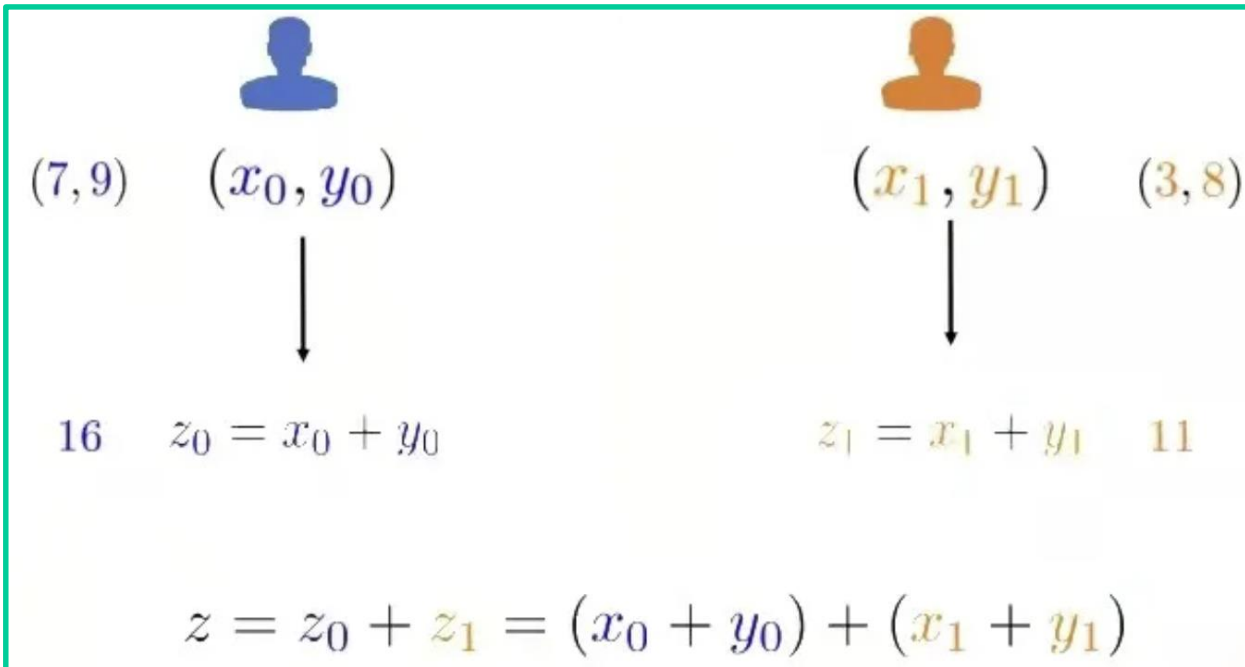


秘密分享 (Secret Sharing)

不同的操作都是以share为输入,以share为输出

由x, y的share为输入,如何得到 $z=x+y$ 的share?

由x, y的share输入, 如何得到 $z=x*y$ 的share?



Beaver's Triple : 满足一定关系的随机数

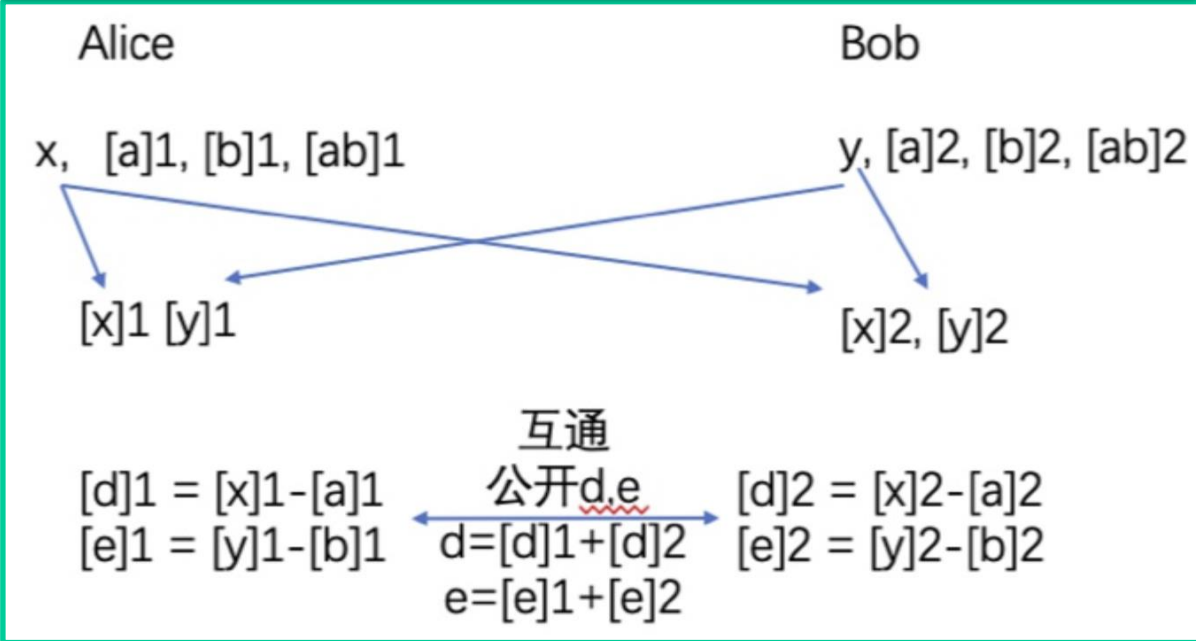
$$(a_0, b_0, c_0) \quad (a_1, b_1, c_1)$$

$$(a_0 + a_1) \cdot (b_0 + b_1) = c_0 + c_1$$

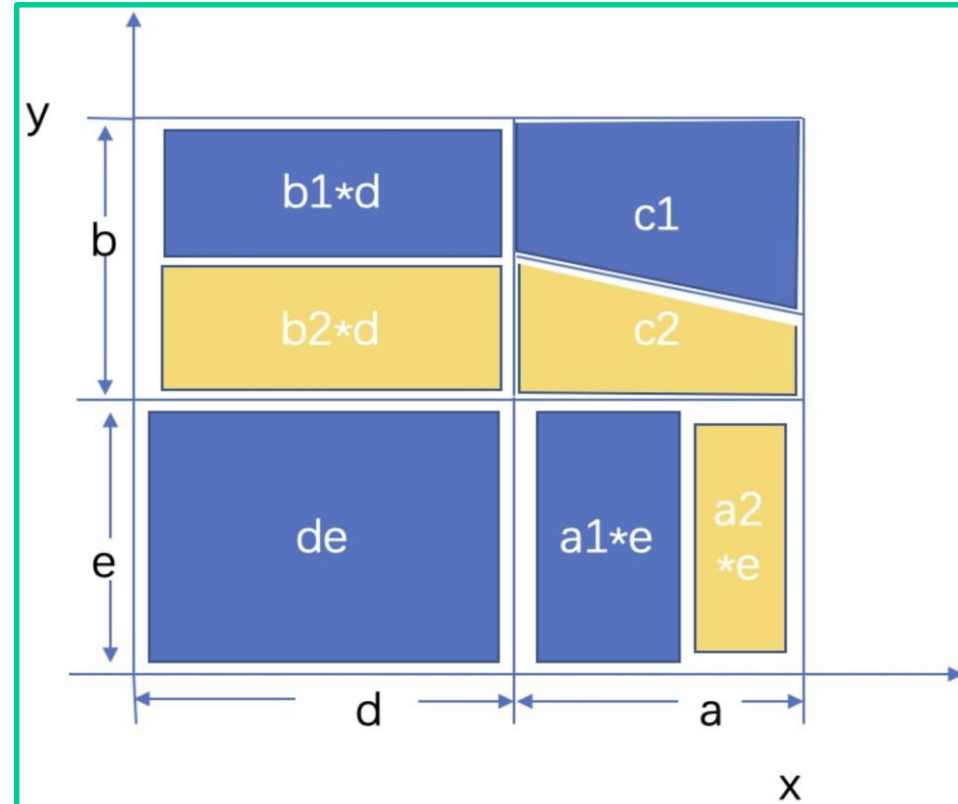
$$(2, 1, 12) \quad (4, 2, 6)$$

乘法运算

乘法运算的分享依赖于一个被分享的三元组 $\langle a, b, ab=c \rangle$



$$z_1 = de + d[b]_1 + [a]_1 e + [ab]_1, \quad z_2 = d[b]_2 + [a]_2 e + [ab]_2$$



Alice和Bob分别持有 x 和 y , 他们得到了一对随机乘法三元组 $\langle a, b, ab \rangle$ 的分片。它们采用如下方法计算 x, y 乘积 (的分片) :

1) 各自将己方数据分享一个分片给对方。

2) 各自通过加法分享并还原, 完成盲化, 分别得到 x 的盲化 $d=x-a$, y 的盲化 $e=y-b$, 并把 d, e 公开给对方。

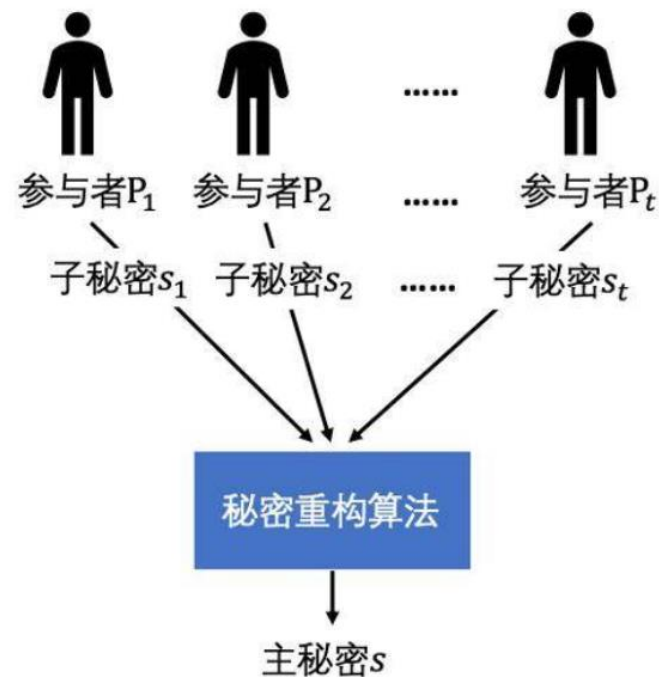
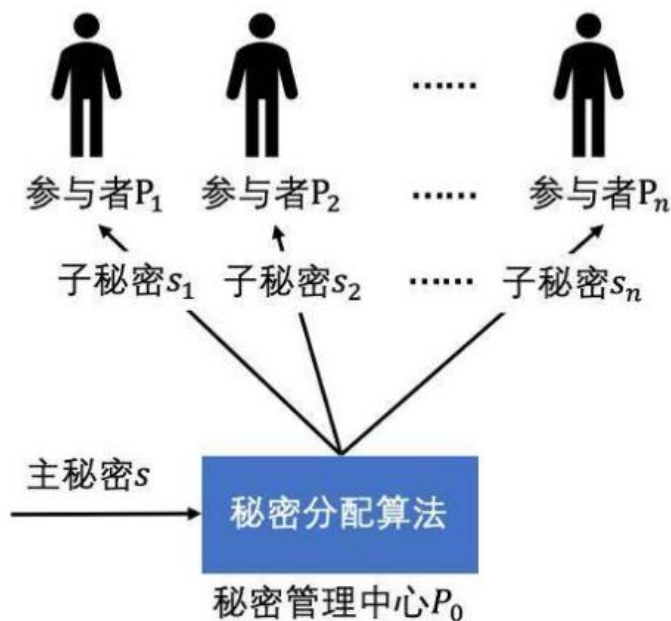
注意在此过程中, x, y, a, b 均没有泄露。

3) 此时 $xy=(d+a)(e+b)=de+[b]d+[a]e+[ab]$, 处于分享态的三元组放在了中括号内, 可以看出问题已被转化为加法问题。

如同一块长方形的瓷砖被分成了7块, 一方计算4块, 另一方计算3块, 如此就完成了乘法运算的加性分享。

门限秘密分享 - Shamir 秘密共享机制

- N 个人分别持有秘密 s 的一部分，当且仅当 t 或以上个人在尝试才可以恢复出秘密 s

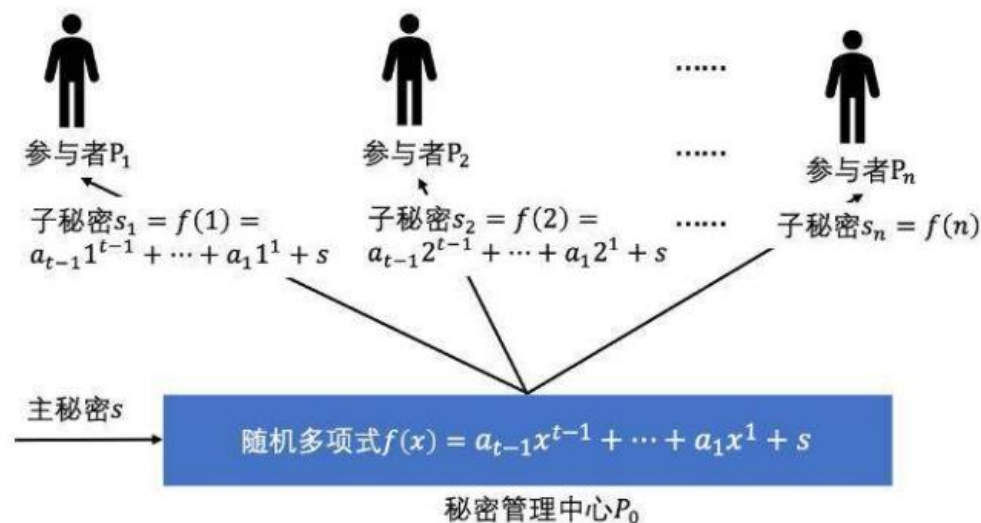


Shamir 秘密共享机制

- 首先, 对于秘密 s , 给定一个值 p , 构造一个多项式如 $f(x)=(s+a_1x+ a_2 x^2 +...+ a_{t-1} x^{t-1})\text{mod}(p)$
- 给定 n 个随机值, 每个随机值称为 r_i , 求得 $f(r_i)$, 然后将 $(r_i, f(r_i))$ 发送给第 i 个参与者。分发完成后, 多项式被销毁, 秘密 s 被隐藏
- 根据数学原理, 当我们拥有 t 个参与者持有的 $(r_i, f(r_i))$ 时, 就能够重建原 t 次多项式, 并得到秘密 s 。重建方式使用多项式插值法

$$f(x) = \sum_{i=1}^t \left(\prod_{1 \leq j \leq t, j \neq i} (x - r_j) (r_i - r_j)^{-1} \right) f(r_i) \pmod{p}$$

- $s = f(0)$



应用与挑战

- 虽然二十世纪八十年代，密码学家就提出了MPC算法，但**一直以来MPC协议的效率都很低，无法在实际中使用**，直到2004年“公平参与（Fairplay）”编译器的提出改变了这一现状。
 - “公平参与”是第一个通用MPC框架，可以通过MPC执行任意函数的计算。
 - 在接下来的10年，这一框架掀起了学者们**针对MPC协议性能优化的浪潮**。直至现在，无论从算法角度还是从实现层面，MPC协议得到了巨大的优化。
- 从2004年到2019年，MPC协议的执行效率至少被提高了五个数量级。

Malkhi, D., N. Nisan, B. Pinkas, and Y. Sella. 2004. “**Fairplay** - Secure Two-Party Computation System”. In: *USENIX Security Symposium*.

安全多方计算通用框架（SP 2019）

- 《SoK: General Purpose Frameworks for Secure Multi-Party Computation》这篇论文非常详尽的分析了现有的安全多方计算通用框架，从各种维度评估了各个框架的优缺点，在不同场景下给出了使用建议。
 - 特别地，作者成功构建了所有的通用框架，并把构建环境打包在Docker中。
- 这篇文章特别适合准备研究安全多方计算协议，或者准备在实际中使用安全多方计算的研究人员。

- 计算的可验证性
 - 很多方法忽略了对参与方输入、输出的验证，其正确性都依赖于计算参与方完全诚实遵循协议进行计算。计算的可验证性是一个非常重要，却被长期忽略的问题。
 - 通用GMW编译器使用零知识证明以及承诺等工具，迫使参与方必须依照协议来执行，从而可以将半诚实敌手下安全的协议编译为可抵抗恶意敌手攻击的协议。
- 计算开销大
 - 这些协议通常计算开销都非常大，是其应用于实际系统的主要障碍。
 - 1) 简化混淆电路的规模
 - 2) 对于恶意的敌手，引入 cut-and-choose 技术
 - 3) 专用型MPC：为解决特定问题所构造出的特殊MPC协议。由于是针对性构造并进行优化，专用算法的效率会比基于混淆电路的通用框架高很多。

如何提高混淆电路的性能？

1、简化混淆电路的规模

对于任意一个给定的计算任务,若将其转化为电路,其电路门的数量将是一个巨大的数字,而混乱电路将每一位通过一个随机数(属于某对称密码算法的密钥空间)及两重对称加解密运算进行混乱,计算负荷巨大,因此简化混乱电路的规模是提高安全多方计算协议效率直接有效的方法.

2、对于恶意的敌手, 引入 cut-and-choose 技术

通用GMW编译器使用零知识证明以及承诺等工具, 迫使参与方必须依照协议来执行, 从而可以将半诚实敌手下安全的协议编译为可抵抗恶意敌手攻击的协议.

但也因为它大量地使用零知识证明以及承诺等协议, 需要大量的公钥操作, 非常低效.

cut-and-choose技术有效地减少了零知识证明等低效操作. 基于cut-and-choose技术应用的层次不同, 这些工作大致可以分为 2 类:

- 1) 对整个混乱电路来实施cut-and-choose技术;
- 2) 针对混乱电路的电路门实施 cut-and-choose 技术.

专用型MPC

- 根据支持的计算任务MPC可分为专用场景和通用场景两类。
- 通用型MPC
 - 通用型MPC算法一般由混淆电路（GC）实现，具有完备性，理论上可支持任何计算任务。具体做法是将计算逻辑编译成电路，然后混淆执行，但对于复杂计算逻辑，混淆电路的效率会有不同程度的降低，与专用算法相比效率会有很大的差距。
- 专用型MPC
 - 专用型MPC是指为解决特定问题所构造出的特殊MPC协议，由于是针对性构造并进行优化，专用算法的效率会比基于混淆电路的通用框架高很多，当前MPC专用算法包含四则运算，比较运算，矩阵运算，隐私集合求交集，隐私数据查询等。
 - 虽然专用型MPC与通用型MPC相比效率更高，但同样存在一些缺点，如只能支持单一计算逻辑，场景无法通用；另外专用算法设计需要领域专家针对特定问题精心设计，设计成本高。



阅读材料 – 基础知识与应用



- 1) Hastings, Marcella, Brett Hemenway, Daniel Noble, and Steve Zdancewic. "Sok: General purpose compilers for secure multi-party computation." In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 1220-1237. IEEE, 2019.
- 2) <https://github.com/MPC-SoK/frameworks>
- 3) David Evans, Vladimir Kolesnikov and Mike Rosulek, *A Pragmatic Introduction to Secure MultiParty Computation*. NOW Publishers, 2018. (This version: April 15, 2020)
- 4) 2021_CACM_Secure Multiparty Computation (MPC)
- 5) 2022_冯登国_Concretely efficient secure multi-party computation protocols-survey and more

SMPC在机器学习隐私保护的应用

两方

- 半诚实
- 构造用于两方计算环境的基于算术共享、布尔共享和Yao混淆电路的安全计算方案，使用Beaver乘法三元组

ABY

- 半诚实
- 提出新的定点乘法计算协议，并针对乘法计算溢出问题提出高效的截断协议

SecureML

多方

- 半诚实
- 利用半诚实的第三方代替乘法三元组中的可信第三方，基于加法秘密共享、GMW、GC设计安全协议，降低了通信成本

Chameleon

- 半诚实或恶意（至多允许一个恶意腐化方）
- 将ABY扩展至三方模型中，在SecureML基础上设计三方截断协议

ABY3

- 半诚实或恶意（至多允许一个恶意腐化方）
- 利用秘密共享的不对称性，放弃SecureML和ABY 3中一些消耗较高的混淆电路或并行前缀加法器，提出新的安全比较协议

ASTRA

- 半诚实或恶意（至多允许一个恶意腐化方）
- 只使用算术秘密共享而避免使用转换协议(在算术、布尔和混淆电路之间)来实现针对非线性运算的恶意安全协议

FALCON



北京邮电大学

Beijing University of Posts and Telecommunications

感谢聆听！
