* **基础概念**

什么是协议：

协议是指双方或者多方为了完成一项任务所进行的一系列步骤，而每一步必须依次执行，在前一步完成前，后面的步骤是不能进行的。

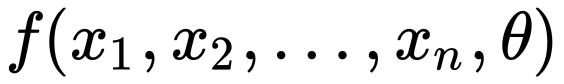
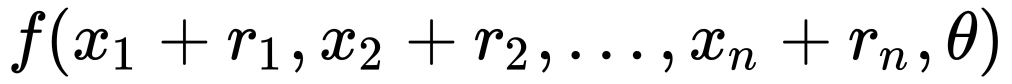
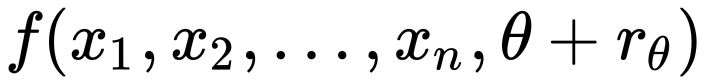
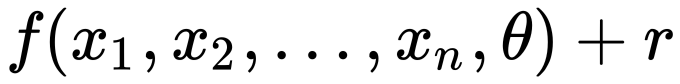
什么是密码协议：

密码协议是以密码算法为基础的协议，也称作安全协议。它是使用密码技术完成某项任务并且满足安全需要的协议。

* **差分隐私**

安全计算这个领域虽然很小众，但其实理论界很早就开始关注它了。研究计算理论的学者们四十多年当中已经提出了很多种实现安全计算的方法。总的来说，大致可归为两类：一类是基于噪音的，另一类不是基于噪音的。当然也有人认为基于噪音的不应算在安全计算当中，但这就纯粹是抠字眼的文字游戏了，我们这里先不管这些，为方便以后的叙述，暂且把“保护数据隐私的多方计算方法”都归入到安全计算当中。

基于噪音的安全计算方法，最主要代表是目前很火的差分隐私（differential privacy）。这类方法的思想是，对计算过程用噪音干扰，让原始数据淹没在噪音中，使别有用心者无法从得到的结果反推原始数据。这就好像我们拿到一张打了马赛克的图片，虽然可能可以猜出马赛克后面大概长啥样，但很难知道马赛克后面的所有细节。

有一点值得注意：这个干扰既可以是数据源，也可以是模型参数和输出。也就是说，参与者既可以对自己的原始数据加噪音使得原始数据从来没在计算过程中出现过，也可以在模型训练的时候改变通过改变模型参数影响输出结果，也可以直接在输出暴露前在输出上加噪音从而使得从计算结果无法反推输入。比如我们要计算一个函数  ，那么对输入进行干扰后得到的结果便是  ，对参数进行干扰后得到的结果为  ，对输出进行干扰后的结果是  。

差分隐私正是为了应对差分攻击而生，简单来说就是给查询结果加噪声（实际上，也可以对源数据/模型参数加噪声），但是一定要恰到好处。噪声加太大，数据集得出的统计值就没有意义了；而噪声加太小，还是很容易被差分攻击攻破，无法保证隐私。

目前常用的添加噪音的机制有**拉普拉斯机制**和**指数机制**。其中拉普拉斯机制用于保护数值型的结果，指数机制用于保护离散型的结果。其中，拉普拉斯机制和随机化回答是两个经典的差分隐私算法，还有许多其他不同的算法，一般而言，不同的应用场景、不同的数据集、不同的输出往往需要不同的算法设计。在神经网络中，计算梯度中加入噪声。

* **不经意传输协议(Oblivious Transfer, OT)**

       OT协议是Rabin于1981年提出的一种两方计算协议，是隐私求交的一种基础协议，现在被广泛应用于安全多方计算领域。

**1-out-of-N OT**

设A方有一个列表，B方选 ，B通过𝑖从A列表中选择一个元素，但A不能得到关于𝑖的信息，B也只能得到

如果 𝑛=2，就是典型的 1-out-of-2 不经意传输协议。

典型的 1-out-of-2 OT 协议：

文本

中度可信度描述已自动生成

在step4时，或可能为 r， 再跟 r 异或操作后为0，所有肯定会得到或是.

表格

描述已自动生成

* **高效的不经意传输（OT）扩展（IKNP协议）**

该协议的目的是通过执行固定次数的不经意协议，实现任意数量的不经意传输。

具体内容如下：

文本

描述已自动生成

手机屏幕截图

描述已自动生成

当时，为,

当时，为.

文本, 信件

描述已自动生成

隐私集合求交(Private Set Intersection, PSI)

**FHE在计算有限次乘法后需要较复杂的去除噪声的操作，经典的通用MPC协议通信开销较大，而TEE的安全性高度依赖于硬件厂商，无法提供密码学上严谨的安全性。**可信执行环境（Trusted Execution Environment，TEE）

PSI实现方案：

# PSI定义

PSI，即Private Set Intersection，是允许持有集合的两方，通过比较加密的集合来计算交集的一种两方安全计算密码技术。

文本

描述已自动生成

一般的PSI协议结果如图所示，Alice作为客户端，Bob作为服务端。两者通过运行协议，Alice除了知道最后的集合相交结果，并不知道Bob的集合有哪些元素。Bob则既不知道交集的结果，也不知道Alice的集合有哪些元素。当然，也存在一些协议的变种，比如C和S都能得到交集的结果，或者C只得到交集的个数，或者C得到了交集的和。

# 基于DDH的PSI方案

图示

低可信度描述已自动生成

方案基于DDH问题，选取一个群G，群的阶是2048 bit长度的素数，生成元为g。方案分Setup和Excute阶段。

Setup阶段: Alice和Bob分别选取私钥⍺，β。

Excute阶段:

1a. Alice把自己的集合x经过Hash后，再用私钥加密，用集合X表示，发送给Bob；

1b. Bob把自己的集合y经过Hash后，再用私钥加密，用集合Y表示，发送给Alice；

2a. Alice接收到Bob的集合Y，用私钥加密，生成了集合Y^ ⍺；

2b. Bob接收到Alice发过来的集合X，用私钥加密，生成了集合 X^β；

3. Alice比较集合Y^ ⍺与 X^β，选取其中相同的元素。

正确性说明：

如果x=y，则（H(x)^⍺)^β  = （H(x)^β)^⍺  = （H(y)^β)^⍺

# Privpy中实现

Privpy架构中数据和计算分离，分为DS和ES。PSI在privpy中的执行流程如下：

1. 在接收到任务后，通过任务配置中的参数确定DS的身份，client或者server；

      2. client把数据发送给S1，server把数据发送给S2；

      3. S1与S2执行PSI协议隐匿求交，S1获取到最终结果。

手机屏幕的截图

描述已自动生成

* **零知识证明（zero-knowledge proof）**

零知识证明实质上是一种涉及两方或更多方的协议，即两方或更多方完成一项任务所需采取的一系列步骤。证明者向验证者证明并使其相信自己知道或拥有某一消息，但证明过程不能向验证者泄漏任何关于被证明消息的信息。**简而言之，零知识证明就是既能充分证明自己是某种权益的合法拥有者，又不把有关的信息泄露出去——即给外界的“知识”为“零”。**

满足的规则：

1. A无法欺骗B，若A取出的东西不是来自房间里的，则B可以分辨的出来，即A成功欺骗B的概率很低，我们将这一规则称之为——正确性；
2. B无法欺骗A，若A取出的东西是来源于房间里，B无法欺骗A，A有绝对的优势可以确保B相信他能证明。我们将这一规则称之为——完备性；
3. 正确性和完备性，就是在最小泄露协议中零知识证明所必须具备的性质。二者不可或缺，否则会导致整个逻辑无法闭环，零知识证明也就不存在了。而在未泄露任何信息的情况下，完成零知识证明，我们称之为——零知识性。

图形用户界面, 应用程序

描述已自动生成

**零知识证明在隐私计算上的应用：**

零知识证明可以满足消息的隐私性，可以解决常见的区块链网络中因透明性所带来的地址和资产额度等消息泄露。例如在区块链交易中，如果你需要证明你拥有某种资产，但同时又不想暴露资产的任何信息，那么就需要用到零知识证明。

隐私计算是零知识证明的一个重要的应用领域。若想保护隐私，则可以通过密码学的解决方案对链上数据进行加密，让链上的不同交易之间找不出关联性。零知识证明可以验证计算而不会暴露有关输入和计算本身的任何信息，保证链上数据隐私和安全。保护隐私数据、身份认证以及计算压缩与区块链扩容。

* **秘密分享的门限方案**

解决方案：

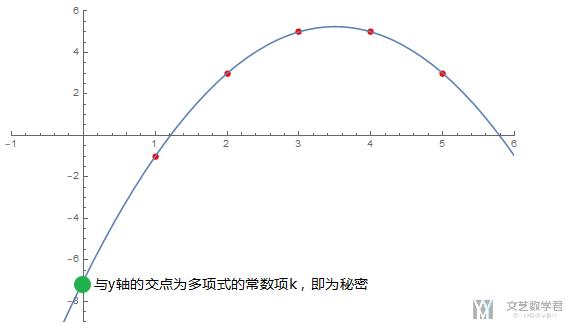
* 拉格朗日插值法（Shamir门限法）

文本, 信件

描述已自动生成

我们可以利用了下面这个事实：知道m-1次多项式函数上的任意m个点就能恢复出整个多项式，例如在（3，5）门限中，我么可以构造已下式子：**f(x)=a x^2+b x+k**，其中k为秘密，任意选出该式子上的五个点，分给5个人，这时任意三个点（任意三个人）就可以解除（a，b，k）其中k就是秘密。当只有两个或更少的人时，是无法解出的。

假设我们要保守的秘密是-7，则我们可以构造多项式f(x) = -7. + 7. x - 1. x^2，并简单计算出五个点{1, -1}, {2, 3}, {3, 5}, {4, 5}, {5, 3}分发个5个人。我们可以看下面的示意图，红色的点为要分配的五个点，绿色的点为多项式与y轴的交点，即我们的秘密。



只有当五个人中有三个来到，我们才能解出秘密。上面的方法易于扩展成（k，n）门限，即只需要对k-1次多项式取n个点分给n个人即可。

文本

描述已自动生成

还有一些Shamir的推广方案，例如Brickell 方案和中国剩余定理（CRT），详情链接：<https://zhuanlan.zhihu.com/p/95362628>。

* 利用三维空间中的点的确定（Blakley 方案）

把秘密文件编码为三维空间中的一个点，然后生成5个过该点的平面，每个人持有其中一个平面方程。显然，两个人在一起是无法获得原文件的，因为两个平面的公共点有无穷多个；但三个平面的交点是唯一的，因此任意三个人在一起都能解开原文件。

这种方法与上面的方法类似，上面的方程为构造形如**f(x)=a x^2+b x+k**的式子，而在这里是构造形如**z=Ax+By+k**的式子，并且，这种方法也较为容易的扩展为（k，n）门限，只需要在k维空间里取n个点即可。

图片包含 表格

描述已自动生成

* 传统方法

把这份文件的密钥拆成C(n,m-1)份，每个人持有C(n-1,m-1)份密钥。

例如在(3,5)门限方案中，我们需要C(5,2)=10个密钥，不妨分别用0到9编号；5个人各持有C(4,2)=6个密钥，密钥的分配如下：

* 人#1: 012345
* 人#2: 012 678
* 人#3: 0 34 67 9
* 人#4: 1 3 56 89
* 人#5: 2 45 789

上述分配表的构造其实很简单：为每一个人的5选3组合分配一个密钥，例如：

* 把密钥0分给特工1、2、3
* 把密钥1分给特工1、2、4
* 把密钥9分给特工3、4、5

这样的话，任意两个人在场都无法打开文件，因为他们始终缺少一把钥匙（这把钥匙分给了其余三个人）。而任意三个人在场都足以打开文件，**因为根据鸽笼原理，任何一个5选3组合中总有一个人落在这三个人当中。**这样，我们便利用组合数学巧妙地解决了这一问题。