How to Share a Secret[1]

如何分享一个密码

Adi Shamir

Massachusetts Institute of Technology

翻译：TIMETOA（2264515424@qq.com）[2]

V1.0

在本文中我们展示了一种数据切割与重构方法，用这种方法将数据 D 分成 n 块，D 可以由其中任何 k 块重建，但即使完全了解 k–1 块也绝对不会泄露任何关于数据 D 的信息。该技术可以构建密码系统的健壮密钥管理方案，即使当因为某些原因，破坏了一半的数据分块，并且安全漏洞暴露了除剩余分块中一个分块之外的所有分块时，该管理方案仍然可以安全可靠地运行。

关键词：密码学，密钥管理，插值

1.Introduction

在[4]中，Liu 考虑了以下问题：

11 位科学家正在进行一项秘密计划。他们希望将文件锁在一个柜子里，这样当且仅当六个或更多的科学家在场时，柜子才能打开。需要的最小锁数是多少？每个科学家必须携带的锁的最少钥匙数量是多少？

不难看出，最小的解决方案需要使用 462 把锁和每个科学家 252 把钥匙。这些数字显然是不切实际的，当科学家的数量增加时，它们会呈指数级增长。

[1] 原文引用：Shamir A . How to share a secret[J]. Communications of the ACM, 1979.

[2] 译文来自于经典文献翻译项目 https://gitee.com/uisu/InfSecClaT，欢迎大家加入经典翻译项目，使更多的人能够阅读这些经典文献，能够获得这些经典文献所传递的信息，并受到这些信息所表达的思想的启发。

本文将该问题推广到秘密是某个数据 D 的情形（例如，安全组合密码），并且允许非机械解决方案（其可操作这个数据）。我们的目标是以如下方式将 D 分成 n 个部分$D_1, \dots, D_n$：

（1）任何 k 个或更多个D_i分块的信息使得 D 易于计算;

（2）任何 k- 1 个或更少的D_i分块的信息使得 D 完全不确定（从某种意义上说，取所有可能的值都是同样可能的）。

这种方案被称为（k，n）门限方案。

有效的门限方案在密钥管理中具有重要的意义。为了保护数据，我们可以对其进行加密，但为了保护加密密钥，我们需要不同的方法（进一步加密会改变问题，而不是解决问题）。最安全的密钥管理方案将密钥保存在一个单独的、受到严密保护的位置（计算机、人脑或保险箱）。这种方案是非常不可靠的，因为一旦这些机器或者人因为某些原因无法正常工作（计算机故障，人突然死亡，或机器被破坏），将使信息无法访问。一个明显的解决方案是在不同的位置存储密钥的多个副本，但这增加了密钥被安全漏洞泄露的危险（计算机渗透、背叛或人为错误）。通过使用$n = 2k - 1$的（k, n）门限方案，我们得到非常鲁棒的密钥管理方案：即使 n 个片段中的$[n/2] = k - 1$个片段被破坏，我们也可以恢复原始密钥，但即使安全漏洞暴露了剩余 k 个片段中的$[n/2] = k - 1$个片段，我们的对手也无法重建密钥。

在其他应用中，不需要在保密性和可靠性之间权衡，需要权衡的是安全性和使用方便性。例如，考虑一家对所有支票进行数字签名的公司（参见 RSA [5]）。如果给每位高管一份公司的秘密签名密钥，这个系统虽然方便，但很容易被滥用。如果为了签署每一张支票，公司所有高管都要一起合作才能完成，那么该系统是安全但不方便的。标准的解决方案要求每次检查至少三个签名，并且很容易用（3，n）阈值方案实现。每个管理人员都有一张带有一个D_i的小磁卡，公司的签名生成设备接受其中的任何三个，以便生成（稍后销毁）实际签名密钥 D 的临时副本。该设备不包含任何秘密信息，因此不需要保护其免受检查。如果有一个管理人员想要伪造公司的签名，那么他必须至少有两个同谋才能完成这个计划。

门限方案非常适合于一组相互怀疑、利益冲突的个体必须合作的应用。理想的情况是，我们希望合作建立在相互同意的基础上，但这一机制赋予每个成员的

否决权可能使该团体的功能无法运行。通过适当地选择 k 和 n 参数，我们可以给予任何足够大的多数采取某种行动的权力，同时给予任何足够大的少数阻止它的权力。

2.一个简单的(k, n)门限方案

我们的方案基于多项式插值：给定 2 维平面中的 k 个点$(x_1, y_1) \dots \dots (x_k, y_k)$。对于不同的$x_i$，存在且仅存在一个 k - 1 次多项式$q(x)$，使得对于所有$i$，$q(x) = y_i$。在不失一般性的情况下，我们可以假设数据 D 是（或可以被制成）数字。为了将其分成多个部分D_i，我们选取随机 k-1 次多项式$q(x) = a_0 + a_1 x + \cdots a_{k-1} x^{k-1}$,其中$a_0 = D$，并且计算：

$$D_1 = q(1), \dots, D_i = q(i), \dots, D_n = q(n)$$

给定这些D_i值的 k 个元素的任何子集（连同它们的标识索引），我们可以通过插值来找到$q(x)$的系数，然后计算$D = q(0)$。另一方面，仅仅知道这些值中的 k-1 个是不足以计算 D 的。

为了使这一方案被更精确地描述，我们使用模算术代替真实的算术。以素数 p 为模的整数集合形成了一个域，在该域中可以进行多项式插值。给定一个整数值数据 D，我们选择一个比 D 和 n 都大的素数 p。从[0，p）中的整数上的均匀分布中随机选择$q(x)$系数$a_1, \dots \dots, a_{k-1}$，并且值$D_1, \dots \dots, D_n$以 p 为模计算。

现在让我们假设这 n 块中的 k-1 块被展示给敌手。对于[0，p）中的每个候选值D'，他可以构造一个且仅一个 k-1 次多项式$q'(x)$，使得对于 k-1 个给定自变量，$q'(0) = D'$并且$q'(i) = D_i$。通过这样的构造，这 p 个可能的多项式的可能性相等，因此敌手绝对不能推断出 D 的真实的值。

[1]和[3]中讨论了多项式求值和插值的有效$O(n \ \ log^2 \ \ n)$算法，但即使是简单的二次算法对于实际的密钥管理方案也足够快。如果数据 D 很长，建议将其分成较短的位块（单独处理），以避免多精度算术运算。分块不能任意短，因为 p 的最小可用值是 n + 1（在[0，p）中必须至少有 n + 1 个不同的参数来评估$q(x)$）。然而，这不是严格的限制，因为 16 位模数（其可以由便宜的 16 位算术单元处理）足以用于具有高达 64000 个D_i分块的应用。

该（k，n）门限方案的一些有用性质（当与机械锁和钥匙解决方案相比时）

是：

（1）每一块的大小不超过原始数据的大小。

（2）当 k 保持固定时，可以动态地添加或删除D_i分块（例如，当管理者加入或离开公司时），而不影响其他D_i分块。（只有当一位即将离职的管理者让它完全无法访问，甚至连他自己也无法访问时，才会删除这个数据块。）

（3）在不改变原始数据 D 的情况下，很容易改变D_i分块--我们所需要的只是一个具有相同自由项的新多项式$q(x)$。这种类型的频繁改变可以极大地增强安全性，因为由安全漏洞暴露的片段不能被累积，除非它们都是$q(x)$多项式的相同版本的值。

（4）通过使用多项式值的元组作为D_i块，我们可以得到一个分层方案，其中确定 D 所需的块的数量取决于它们的重要性。例如，如果我们给予公司总裁三个$q(x)$值，给每个副总裁两个 q（x）值，给每个高管一个$q(x)$值，那么一个（3，n）门限方案使得支票可以由任意三个高管签署，或者由任意两个高管（其中一个是副总裁）签署，或者由总裁一个人签署。

最近，G.R. Blakley [2] 提出了一种不同的（效率稍低的）阈值方案。.

1979 年 4 月收到此文章；1979 年 9 月修改；

参考文献

[1] Aho, A., Hopcroft, J., and Ullman, J. The Design and Analysis of Computer Algorithms. Addison-Wesley, Reading, Mass., 1974.

[2] Blakley, G.R. Safeguarding cryptographic keys. Proc. AFIPS 1979 NCC, Vol. 48, Arlington, Va., June 1979, pp. 313-317.

[3] Knuth, D. The Art of Computer Programming, Vol. 2: Seminumerical Algorithms. Addison-Wesley, Reading, Mass., 1969.

[4] Liu, C.L. Introduction to Combinatorial Mathematics. McGraw-Hill, New York, 1968.

[5] Rivest, R., Shamir, A., and Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. Comm. A CM 21, 2(Feb. 1978), 120-126.