第五届（2020）全国高校密码数学挑战赛

赛题二

一、赛题名称：向量布尔函数求逆问题

二、赛题描述

2.1 符号说明

F_2：布尔域$\{0,1\}$；

$X=(x_0,x_1,\cdots,x_{n-1})$：$n$维布尔向量，即$x_i \in \mathrm{F}_2$；

$Y=(y_0,y_1,\cdots,y_{m-1})$：$m$维布尔向量，即$y_i \in \mathrm{F}_2$；

l：为Y中前置 0 串的长度,即从y_0开始连续 0 的最大个数；

H：为$F_2^n \to F_2^m$的向量布尔函数，可以写成如下形式：

$$H(\mathrm{x}_0,x_1,\cdots,x_{n-1})=(y_0,y_1,\cdots,y_{m-1}).$$

2.2 基础知识

布尔函数定义在布尔域F_2上，域中元素只包含 0 和 1。布尔域中的加法运算是整数加法模 2，也称为异或运算，即$0\oplus 0=0,0\oplus 1=1,1\oplus 1=0$。布尔域中的乘法是整数乘法模 2，即$0\cdot 0=0,0\cdot 1=0,1\cdot 1=1$。定义$f:F_2^n \to F_2$，则$f$为布尔函数。包含布尔函数所有可能的输出输入值的表格称为真值表，是布尔函数的表示形式之一。此外，还可以用布尔多项式表示布尔函数，多项式中变量即为布尔函数的输入，多项式的系数取值范围也是$\{0,1\}$。

当函数的输出值取值范围为向量空间$\{0,1\}^m$时，即为向量布尔函数。

向量布尔函数求逆问题是一个经典问题，其数学表示为：给定向量布尔函数$H：F_2^n \to F_2^m$，以及m维布尔向量$Y=(y_0,y_1,\cdots,y_{m-1})$，计算$X=(x_0,x_1,\cdots,x_{n-1})$使得$H(X)=Y$成立。

密码学中，很多密码算法都是向量布尔函数，相关密码的攻击问题，如求秘钥或求原像消息值，都可以归结为向量布尔函数求逆问题，因此研究向量布尔函数求逆问题具有重要的意义。

2.3 问题描述

本竞赛的目标是求X值，使得输出Y向量中前置 0 串长度最大。具体来说，竞赛中考虑如下特殊的向量布尔函数H：

$$H(x_0,x_1,\cdots,x_{\mathrm{k}*1088-1},n)=(y_0,y_1,\cdots,y_{255}).$$

注意这里的n称为函数H的轮数，k可以为整数$1,2,\cdots$，下面的算法将会对n和k的内涵进行更清晰的解释。l为Y向量中前置 0 串的长度，本赛题的具体目标是计算$(x_0,x_1,\cdots,x_{k*1088-1})$使得$l$尽可能大，$l$越大则选手从相应问题得到的分数越多。

向量布尔函数H定义如下：

输入：消息值:$(x_0,x_1,\cdots,x_{k*1088-1})$; 轮数:$n=1,2,3,4,5.$

输出：摘要值:$(y_0,y_1,\cdots,y_{255})$

$s_i=0,\text{for i}=0,1,\cdots,1599$

for $\boldsymbol{m=0}$ to $\boldsymbol{k-1}$ do

$(s_0,s_1,\cdots,s_{1087})$

$=(s_0,s_1,\cdots,s_{1087})\oplus\left(x_{\mathrm{m}*1088},x_{\mathrm{m}*1088+1},\cdots,x_{(m+1)*1088-1}\right)$

---

for $r = \mathbf{1}$ to $\boldsymbol{n}$ do

$(\mathrm{s}_0, \mathrm{s}_1, \ldots, \mathrm{s}_{1599}) = \iota \circ \chi \circ \pi \circ \rho \circ \theta(\mathrm{s}_0, \mathrm{s}_1, \ldots, \mathrm{s}_{1599})$

end for

end for

$Y = (s_0, s_1, \cdots, s_{255})$

其中，$\theta, \rho, \pi, \chi, \iota$操作定义请见研究背景部分。

2.4 成绩评判标准

本次竞赛H函数的最高轮次为5轮。我们将对每轮向量布尔函数H中使得Y中前置0串长度l最大的消息X进行评分，并将5轮得分的总和作为选手最后的总得分。记n轮下选手计算出的最长前置 0 串长度为l_{n}，则其总得分计算公式为：

$$\mathrm{Score} = \sum_{n=1}^{5} Score_n, \mathrm{Score_n} = l_n \times \delta_n;$$

其中，δ_n为第n轮的轮难度系数：$\delta_1 = 1, \delta_2 = 2, \delta_3 = 3, \delta_4 = 10, \delta_5 = 20$。每轮前置0串长度都不会超过256，因此所有竞赛题的满分为9216分。

特别注意：为了鼓励选手对求解算法的研究，当$l_{\mathrm{n}} < 64$时,将l_{n}作为 0 计分，即只有计算出使Y前置0串长度不小于64的消息X才会计分。

下面举例说明选手得分的算法，假设某位选手得到以下结果：

1轮问题中，找到使Y前置0串长度为64, 96, 128的三个消息；

2轮问题中，找到使Y前置0串长度为120, 129, 256的三个消息；

3轮问题中，未找到使Y前置0串长度不小于64的消息；

4 轮问题中，找到使Y前置 0 串长度为 64，100 两个消息；

5 轮问题中，未找到使Y前置 0 串长度不小于 64 的消息；

在这种情况系，该选手各轮前置 0 串最大长度为$l_1 = 128, l_2 = 256, l_3 = 0, l_4 = 100, l_5 = 0$；各轮得分为：$\mathrm{Score}_1 = 128$，$\mathrm{Score}_2 = 256 \times 2 = 512$，$\mathrm{Score}_3 = 0, \mathrm{Score}_4 = 100 \times 10, \mathrm{Score}_5 = 0.$

总分为：$\mathrm{Score} = 128 + 512 + 0 + 1000 + 0 = 1640$；

比赛排名规则：

（1）最终总分（Score）最高者获胜，分数相同的队伍比较其在最高轮数中的得分，如分数仍相同，则比较次高轮数得分，以此类推。

（2）如分数无法分出胜负，则根据完成挑战方法进行综合评定。

提交结果说明：

（1）针对于已完成的挑战实例，需给出计算平台、计算所需空间时间、和计算结果，并简述求解原理和步骤；

（2）利用特殊方法求解或求解方法中有创新内容的，酌情加分；

（3）参赛者在报告摘要中明确列出轮数n、前置 0 串最大长度l_{n}及相应的填充后消息X、消息块数k、压缩值Y，以及自己获得的总分数 *Score*。在报告正文中详细描述每个挑战问题的计算方法及相关数据。引用前人的方法需在报告中明确指出，否则结果作废。

三、密码学背景及相关问题的研究进展

区块链中的挖矿过程就是速度的较量过程。挖矿的时候，矿工使

用计算机反复迅速地猜一个数学难题的答案，直到有矿工找到正确答案为止。更具体来说，矿工会对区块头元数据（包括时间戳、软件号和这个区块的交易哈希值）进行哈希函数运算（函数会得到一个由字母和数字组成、固定长度的字符串，即哈希值），其中会影响哈希值结果的随机值是唯一的变量。如果某矿工率先找到满足要求的哈希值，那么该矿工将获得虚拟货币作为奖励。

公共区块链平台以太坊中的杂凑函数算法使用了 Keccak-256 算法，竞赛题目就是模拟以太坊中的挖矿过程，本质上就是计算满足条件的 Keccak-256 的原像问题。目前 Keccak 的原像攻击主要还是通过求解代数系统来实现。低轮次的原像攻击主要通过 SAT 求解器完成。由于高轮 Keccak 算法对应的代数系统次数较高，在随后的几年里，原像攻击方面都没有实质性的进展。直到 2016 年新加坡南洋理工大学的郭建和中国科学院信息工程研究所的刘美成、宋凌提出了一种线性结构，才获得了更好分析结果[2]。2019 年，中国科学院信息工程研究所的孙瑶和李婷基于非线性代数方法给出了目前 3 轮和 4 轮 Keccak 原像攻击的最好结果[3]。

Keccak 算法是美国国家标准与技术研究院（NIST）确定的第三代哈希算法标准（SHA-3），采用新型的海绵结构，是目前最重要的哈希算法。Keccak-f 是 Keccak 算法的主要置换函数，包括m轮R置换，每个R置换包含以下 5 个操作复合而成：

$$R = \iota \circ \chi \circ \pi \circ \rho \circ \theta,$$

其中每个操作如下所示, 每个比特可以用 3 元组(x, y, z)来定义该比

特的坐标位置，以$A_{x,y,z}$表示：

θ: $A_{x,y,z} = A_{x,y,z} \oplus \oplus_{j=0}^{4}\left(A_{x-1,j,z} \oplus A_{x+1,j,z-1}\right)$，

ρ: $A_{x,y,z} = A_{x,y,z+r(x,y)}$，

π: $A_{y,2x+3y,z} = A_{x,y,z}$，

χ: $A_{x,y,z} = A_{x,y,z} \oplus \left(A_{x+1,y,z} \oplus 1\right) \cdot A_{x+2,y,z}$，

ι: $A_{0,0,z} = A_{0,0,z} \oplus RC_z$.

Keccak 实例中，用“10*1”的比特串对消息进行填充，并且填充后的消息长度是比特率（bit rate）的倍数。满足这一条件的“10*1”比特串有多个，选择其中长度最短的进行填充。

Keccak 算法的具体参数和实现细节可参考文献[1]。Keccak 团队提供了 2 个程序：XKCP 程序（https://github.com/XKCP/XKCP）给出了各类版本的 Keccak 算法的具体实现；KeccakTools（https://github.com/KeccakTeam/KeccakTools）可用于对 Keccak 算法进行基本分析。本次竞赛也会提供验证程序验证选手提供消息的正确性。

四、 参考文献

[1] G. Bertoni, J. Daemen, M. Peeters, G. Van Assche. The Keccak reference. http://keccak.noekeon.org,January (2011). Version 3.0

[2] Guo, J., Liu, M., Song, L. (2016, December). Linear structures: Applications to cryptanalysis of round-reduced KECCAK. In International Conference on the Theory and Application of Cryptology and Information Security (pp. 249-274). Springer, Berlin, Heidelberg.

[3] Li, T., Sun, Y.(2019). Preimage Attacks on Round-Reduced Keccak-224/256 via an Allocating Approach. In: Ishai Y., Rijmen V. (eds) Advances in Cryptology – EUROCRYPT 2019. EUROCRYPT 2019. Lecture Notes in Computer Science, vol 11478. Springer, Cham.