



对1K-AES结构的量子滑动攻击

申洲¹, 张凤荣^{1*}, 韦永壮², 王保仓³

1. 西安电子科技大学网络与信息安全学院, 西安 710071

2. 桂林电子科技大学计算机与信息安全学院, 桂林 541004

3. 西安电子科技大学空天地一体化综合业务网全国重点实验室, 西安 710071

*联系人, E-mail: zhangfengrong@xidian.edu.cn

收稿日期: 2024-11-28; 接受日期: 2025-01-02; 网络出版日期: 2025-02-19

国家自然科学基金(编号: 62372346)、高等学校111学科创新引智计划(编号: B16037)和陕西高校青年创新团队资助项目

摘要 随着量子技术的不断发展, 对称密码算法的量子安全性受到人们的广泛关注. 聚焦1K-AES的安全性, 分别利用量子滑动攻击和经典滑动攻击对1K-AES进行了安全性分析. 一方面, 对最后一轮包含列混淆的1K-AES, 在无需量子随机存储的条件下, 利用CNS算法能够在查询复杂度为 $O(2^{2n/5})$ 和经典存储规模为 $O(2^{n/5})$ 的条件下将成功率由63%提升到74%, 其中 n 表示分组大小. 若使用量子随机存储, 利用通用量子claw查找算法能够在查询复杂度为 $O(2^{n/3})$ 和量子存储规模为 $O(2^{n/3})$ 的条件下将成功率提升到100%. 同时, 在经典选择明文攻击条件下, 本文将经典滑动攻击的成功率也提升到了100%. 另一方面, 对最后一轮无列混淆的1K-AES, 利用通用量子claw查找算法实施量子滑动攻击, 其查询和存储复杂度均为 $O(2^{n/3})$. 与经典滑动攻击相比, 该方法实现了亚指数加速; 与Grover搜索相比, 查询复杂度从 $O(2^{n/2})$ 降低至 $O(2^{n/3})$.

关键词 1K-AES, 滑动攻击, 量子claw查找算法, CNS算法

PACS: 03.67.Dd, 03.67.Lx, 03.67.-a

1 引言

2010年, Kuwakado和Mori^[1]证明, 通过使用Simon算法^[2]可在多项式时间内区分3轮Feistel结构和随机置换. 这一结果颠覆了此前Grover算法^[3]作为对称密码唯一量子威胁的认识, 表明将加密算法的弱点转移到后量子时代确实能够改进通用的Grover量子穷搜. 这引发了密码学研究者对后量子时代对称密码量子安全性的深入分析. 而在对称密码的量子分析中, 一个重要

的方向是将经典计算模型下的密码分析技术量子化, 并提高分析效果^[4].

滑动攻击是Biryukov和Wagner^[5]提出的一种密码分析技术. 该攻击方法主要利用了密码的自相似性, 并且其破解复杂度与密码迭代的轮次无关. Kalplan等人^[6]使用Simon算法对文献^[5]中的滑动攻击进行了量子化, 可以在多项式时间里恢复出具有自相似Even-Mansour密码的密钥, 这是第一个经典对称密码分析技术在量子条件下实现了指数级加速的案例.

引用格式: 申洲, 张凤荣, 韦永壮, 等. 对1K-AES结构的量子滑动攻击. 中国科学: 物理学 力学 天文学, 2025, 55: 240314

Shen Z, Zhang F R, Wei Y Z, et al. Quantum slide attack on 1K-AES structure (in Chinese). Sci Sin-Phys Mech Astron, 2025, 55: 240314, doi: 10.1360/SSPMA-2024-0582

2020年, Dong等人^[7]针对Feistel结构的高级滑动攻击(Advanced Slide Attack)技术^[8]进行了量子化, 且使用量子化后的分析方法在多项式时间里对1K/2K/4K-DES密码进行了密钥恢复攻击.

尽管滑动攻击要求所有轮函数完全相同且使用周期性子密钥, 但是这一分析技术已成功应用于多个Feistel结构密码^[9,10]的分析. 替换-置换网络结构(Substitution-Permutation Network, SPN)是对称密码算法中广泛应用的另一种结构, 通过多次迭代的替换(Substitution)和置换(Permutation)操作来实现加解密. 与Feistel结构相比, SPN结构由于引入了白化密钥, 导致最后一轮和其他轮的结构不一致, 破坏了密码结构的自相似性, 使得对SPN结构实施滑动攻击更加困难. 2018年, Bar-On等人^[10]提出了对具有自相性SPN结构的有效滑动攻击. 2019年, Bonnetain等人^[11]对SPN结构的滑动攻击进行了量子化讨论. Dunkelman等人^[12]详细讨论了对AES-like结构的滑动攻击, 并指出轻量级分组密码可能成为此类攻击的目标, 值得进一步的研究. 随后的研究发现, ULC密码^[13]在相关密钥条件下无法抵御滑动攻击^[14]. 尽管FESH密码^[15]采用了复杂的密钥调度算法, 但在相关密钥条件下仍存在全轮的滑动特性^[16].

虽然当前对SPN结构的滑动攻击复杂度过高, 尚不具有实际可行性, 但是借助量子算法对滑动攻击进行量子化可以使得复杂度有进一步的降低^[16]. 如Cai等人^[17]利用BHT算法^[18]对文献^[10]中提出的最后一轮有列混淆组件(MixColumns, MC)的1K-AES滑动攻击进行了量子化, 使得在相同成功率下, 时间、查询和存储复杂度与经典滑动攻击相比均实现了亚指数加速. 因此, 研究SPN结构在量子条件下的滑动攻击是必要的. 由于目前针对AES-like结构的量子滑动攻击尚不充分, 所以本文主要聚焦于对AES-like结构滑动攻击的量子化及对原有结果的改进. 需要注意的是文献^[10]中的1K-AES没有将最后一轮的MC进行删除, Dunkelman等人^[12]将其称为1K-AESf.

随着量子计算的发展, 在资源受限的条件下改进量子算法效率和提升分析精度受到了各个领域的广泛关注. 例如, Hou等人^[19]设计了一种简洁的无重复量子神经网络, 解决了传统构建通用量子神经网络方法中可能导致量子寄存器过大的问题; Song等人^[20]提出了一种专为经典客户端设计的量子联邦学习框架, 以解

决客户端的量子计算资源有限的问题. 这些改进在机器学习和神经网络领域具有重要意义. 因此, 在量子资源受限的条件下, 提升量子对称密码的分析能力和效率具有重要的理论价值和实际意义. 目前Cai等人^[17]对1K-AESf的量子化方法需要使用大规模量子存储(quantum Random Access Memory, qRAM)^[21], 而制造大规模qRAM的难度是巨大的^[4]. 所以本文将在不使用qRAM这一量子存储模型的条件对1K-AESf进行量子滑动分析.

Chailloux等人^[22]在2017年提出了无需大规模qRAM且优于生日攻击的量子碰撞算法, 称为CNS算法. 由于对1K-AESf的滑动攻击本质上是寻找两个随机函数的一个claw, 即两个函数的“碰撞”. 所以本文基于CNS算法^[22]提出了不需要qRAM存储模型即可寻找两个随机函数claw的量子算法, 并使用此算法对经典1K-AESf的滑动攻击进行了量子化. 由于目前对1K-AESf的滑动攻击没有将成功率提高到100%, 所以本文在选择明文攻击的条件下对其进行了经典和量子化分析使成功率均提高至100%.

对于SPN结构密码, 它的最后一轮结构总是和其他轮数不同^[12]. 例如AES^[23]的最后一轮除了进行白化密钥异或的操作外, 还省略了MC组件, 这导致密码的非对称性加强破坏了密码结构的自相似性, 使得滑动攻击更加困难. 但是文献^[12]利用其提出的引导性明文结构(Suggestive Plaintext Structures)对最后一轮无MC的1K-AES (1K-AES_t)进行了有效的滑动攻击, 而目前还没有文章对其进行量子化分析, 所以本文对1K-AES_t的滑动攻击分析技术进行了量子化.

2 预备知识

本文沿用文献^[12]对AES-like密码结构的命名方式: 1K-AESf表示所有轮次均使用相同的密钥, 每轮的结构一样, 但最后一轮以白化密钥结束. 1K-AES_t表示所有轮次均使用相同的密钥且结构一样, 但最后一轮没有MC组件且以白化密钥结束. 本节将介绍一些基础的量子算法.

2.1 Grover算法

Grover算法^[3]是一种基于振幅放大的量子搜索算法, 用于在一个无序的数据库中寻找特定的元素. 相比

于经典算法, Grover算法在无序数据库的搜索问题中可以取得二次加速. 例如, 在一个包含 $N = 2^n$ 个数据元素的无序数据库中寻找满足特定条件的唯一元素 x_0 . 经典算法需要进行 $\mathcal{O}(N)$ 次搜索, 而Grover算法仅需进行 $\mathcal{O}(\sqrt{N})$ 次的搜索. Grover算法的具体步骤如下.

(1) 初始化: 初始化 n 比特的量子寄存器为 $|0\rangle^{\otimes n}$, 然后对第一个量子寄存器进行Hadamard变换使其处于均匀叠加态中:

$$H^{\otimes n}|0\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle = |\psi\rangle. \quad (1)$$

(2) 振幅放大: 重复进行Grover迭代 $R = \pi\sqrt{2^n}/4$ 次, 每步迭代分为调用量子预言机 O_f 和执行扩散变换 $2|\psi\rangle\langle\psi| - I$ 两步:

$$([2|\psi\rangle\langle\psi| - I]O_f)^R |\psi\rangle \approx |x_0\rangle, \quad (2)$$

其中 $O_f|x\rangle = (-1)^{f(x)}|x\rangle$. $f(x)$ 的取值情况如下:

$$f(x) = \begin{cases} 1, & |x\rangle = |x_0\rangle, \\ 0, & |x\rangle \neq |x_0\rangle. \end{cases} \quad (3)$$

(3) 测量寄存器得到 x_0 .

由Brassard等人^[24]提出的量子振幅放大算法(Quantum Amplitude Amplification, QAA)可以看作Grover算法的推广. QAA算法的结果可以由Brassard等人^[24]提出的定理1进行准确的描述.

定理1 设 A 为不进行量子测量的 q 量子比特的量子算法. $B: \mathbb{F}_2^q \rightarrow \{0,1\}$ 为一个分类器, 将 A 的输出分为“good”和“bad”. 设 $p > 0$ 为测量 $A|0\rangle$ 为“good”态的初始成功概率, $k = \lfloor \pi/4\theta \rfloor$, 其中 θ 定义为 $\sin^2(\theta) = p$. 定义算符 $Q = -AS_0A^{-1}S_B$, 其中 S_B 可以改变“good”态的符号:

$$|x\rangle = \begin{cases} -|x\rangle, & B(x)=1, \\ |x\rangle, & B(x)=0. \end{cases} \quad (4)$$

而 S_0 仅改变 $|0\rangle$ 态的符号. 对 $A|0\rangle$ 经过 Q^k 次迭代后进行测量, 至少以 $\max\{1-p, p\}$ 的概率得到“good”态.

由定理1可知QAA算法的时间消耗主要来自算符 A 和 B 的 k 次迭代, 所以QAA算法的时间复杂度为

$$\mathcal{O}(k(|A|_{RT} + |B|_{RT})) \approx \mathcal{O}\left(\sqrt{\frac{1}{p}}(|A|_{RT} + |B|_{RT})\right). \quad (5)$$

2.2 BHT算法

BHT算法^[18]是一种结合了生日攻击和Grover算法的量子算法, 通过在时间和空间复杂度之间取得折中, 以高效解决碰撞问题. 设函数 $F: X \rightarrow Z$ 为一个2对1函数, 其中 $|X| = |Z| = N$. BHT算法寻找函数 F 碰撞的思路为: 首先选择一个子集 $K \subseteq X$, 然后使用Grover算法寻找碰撞 $\{x_0 \in K, x_1 \in X/K\}$. 而算法的时间、查询和存储复杂度由子集 K 的大小决定.

BHT算法的具体步骤如下.

(1) 任意选择大小为 k 的集合 $K \subseteq X$, 然后构造一个大小为 k 的表 L . 表 L 中的每一条记录为 $(x, F(x))$, 其中 $x \in K$.

(2) 根据表 L 中每条记录的第二项对表 L 进行排序, 若 L 中存在 $(x_0, F(x_0)), (x_1, F(x_1)) \in L$, 使 $F(x_0) = F(x_1)$, 转至步骤(4). 其中表 L 中的数据使用qRAM进行存储.

(3) 计算 $x_1 = \text{Grover}(H, 1)$, 其中 $H: X \rightarrow \{0,1\}$ 表示以下条件定义的函数:

$$H(x) = \begin{cases} 1, & (x_0, F(x)) \in L, x \neq x_0, \\ 0, & \text{其他}. \end{cases} \quad (6)$$

(4) 找到 $(x_0, F(x_1)) \in L$, 输出碰撞 $\{x_0, x_1\}$.

当取 $k = \sqrt[3]{N}$ 时, BHT算法寻找函数 F 碰撞的时间、查询和存储复杂度为 $\mathcal{O}(\sqrt[3]{N})$.

而与碰撞相关的一个概念为claw, 一个claw是指函数 F 和 G 之间存在一组 (x, y) 使 $F(x) = G(y)$. 设两个随机函数为 $F: X \rightarrow Z, G: Y \rightarrow Z$, 其中 $|X| = |Y| = |Z| = N = 2^n$, 那么可以定义如下的函数 J :

$$\begin{aligned} J: \{0,1\}^{n+1} &\rightarrow \{0,1\}^n \\ J(b \parallel x) &= \begin{cases} F(x), & b = 0, \\ G(x), & b = 1. \end{cases} \end{aligned} \quad (7)$$

通过式(7)的定义构造了一个2对1函数 J . 这也是BHT算法可以用于寻找两个随机函数claw的原因.

2.3 CNS算法

由于制造大规模的qRAM面临着巨大的挑战, 而BHT算法在寻找函数碰撞时需要 $\mathcal{O}(\sqrt[3]{N})$ 规模的qRAM存储, 因此, Chailloux等人^[22]在2017年使用嵌套的QAA算法提出了无需qRAM且时间复杂度优于 $\mathcal{O}(\sqrt{N})$

的量子碰撞查找算法.

文献[22]中给出了一个成员预言机, 用于检测一个元素 x 是否存在于一个大型的非结构化集合中, 定义1描述了该成员预言机.

定义1^[22] 设集合 T 为包含 2^t 个两两不同的 n 比特串的集合, 定义经典的成员检测函数 f_T 为

$$f_T(x) = \begin{cases} 1, & x \in T, \\ 0, & x \notin T, \end{cases} \quad (8)$$

f_T 对应的量子算符为 O_{f_T} :

$$O_{f_T}(|x\rangle|b\rangle) = |x\rangle|b \oplus f_T(x)\rangle, \quad (9)$$

当集合 T 被存储在经典内存中时, O_{f_T} 可以在 $n2^t$ 的时间内使用 $2n+1$ 位量子比特实现.

设随机函数 $F: \{0,1\}^n \rightarrow \{0,1\}^n$. 定义集合 $S_r^F = \{(x, F(x))\}$, 其中 S_r^F 中每一项的第二个条目满足前 r 位为0比特串. CNS算法寻找函数 F 的碰撞分为两步.

(1) 使用Grover算法构建一个大小为 2^{t-r} 的表 L , L 为 S_r^F 的子集. 根据式(8)和(9)可以分别构建成员检测函数 f_L^F 和对应的量子算符 $O_{f_L^F}$. 由表 L 的大小可知需要重复执行Grover算法 2^{t-r} 次, 所以构建 L 所需的时间为 $O(2^{t-r/2})$.

(2) 使用QAA($\mathcal{A}, \mathcal{B}$), 其中 $\mathcal{A}$ 为利用Grover算法来生成集合 S_r^F 的叠加态, 时间复杂度为 $O(2^{r/2})$, $\mathcal{B} = O_{f_L^F}$ 用于检测 $(x, F(x))$ 是否属于表 L , 时间复杂度为 $O(2^{t-r})$. 由于初始成功概率 $p = 2^{t-r+1}/2^{n-r}$, 所以QAA($\mathcal{A}, \mathcal{B}$)的时间复杂度为

$$O\left(2^{\frac{n-t-1}{2}}(2^{r/2} + 2^{t-r})\right). \quad (10)$$

综上, CNS寻找函数 F 碰撞所需的时间为

$$O\left(2^{\frac{n-t-1}{2}}(2^{r/2} + 2^{t-r}) + 2^{t-r/2}\right), \quad (11)$$

当取 $r = 2t/3 = 2n/5$ 时, $|\mathcal{A}|_{RT}$, $|\mathcal{B}|_{RT}$ 和构建 L 的时间复杂度之间取得了折中, 此时式(11)取得最小值. 而在存储复杂度方面, 需要 $O(n)$ 规模的量子比特和 $O(2^{n/5})$ 规模的经典存储.

3 1K-AESf的滑动攻击

3.1 无需qRAM的量子滑动攻击

文献[10]对1K-AESf的滑动攻击如图1所示. 其中对于一个明文 P , 1K-AESf的加密函数可以表示为 $E_k(P) = ARK \circ f_k \circ f_k \circ \dots \circ f_k(P)$, ARK 表示与密钥 k 进行异或运算. $f_k = MC \circ SR \circ SB \circ ARK(\cdot)$ 表示一轮的加密函数, 其中 SB^{-1} , SR^{-1} , MC^{-1} 分别表示组件 SB , SR , MC 的逆变换.

Bar-On等人^[10]对1K-AESf滑动攻击的主要思路为: 首先假设明文对 (P, P') 为1K-AESf的一个滑动对, 即 $f_k(P) = P'$:

$$P = k \oplus (SB^{-1} \circ SR^{-1} \circ MC^{-1}(P')), \quad (12)$$

$$\bar{P}' = SB^{-1} \circ SR^{-1} \circ MC^{-1}(P'). \quad (13)$$

然后根据式(12)和(13)可以得到

$$P \oplus \bar{P}' = k. \quad (14)$$

由于组件框(Component Box, CB)中分别进行了两个相同轮次的 f_k 迭代, 且根据假设可得CB的输入是相同的, 所以由 $C' = k \oplus (MC \circ SR \circ SB(C))$ 和 $\bar{C} = MC \circ SR \circ SB(C)$ 可得明文对 (P, P') 对应的密文对 (C, C') 满足式(15):

$$C' \oplus \bar{C} = k. \quad (15)$$

根据式(14)和(15)可以得到

$$P \oplus \bar{C} = \bar{P}' \oplus C'. \quad (16)$$

当找到了满足式(16)的明文对 (P, P') 后, 可以根据式(14)得到候选密钥. 关于1K-AESf滑动攻击的详细流程, 请参考文献[10]. 在已知明文条件下, 文献[10]可以

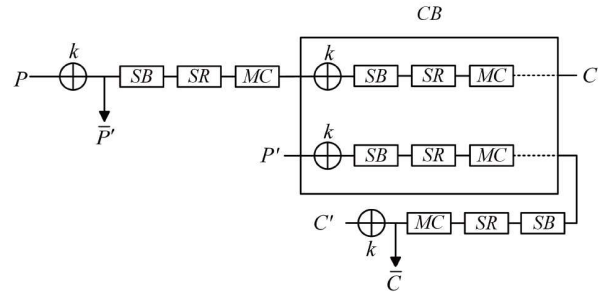


图 1 对1K-AESf的滑动攻击
Figure 1 Slide attack on 1K-AESf.

在数据复杂度、存储复杂度和时间复杂度均为 $\mathcal{O}(2^{n/2})$ 的条件下, 以63%的概率恢复出密钥 k .

而寻找满足式(16)中明文对 (P, P') 的问题可以转换为寻找两个函数的一个claw. 为了便于表述, 定义明文集合为 T_P , 式(16)的左侧为函数 $f_1(x)$:

$$\begin{aligned} f_1: \{0, 1\}^n &\rightarrow \{0, 1\}^n, \\ x &\rightarrow x \oplus MC \circ SR \circ SB(E_k(x)) \quad x \in T_P. \end{aligned} \quad (17)$$

式(16)的右侧为函数 $f_2(y)$:

$$\begin{aligned} f_2: \{0, 1\}^n &\rightarrow \{0, 1\}^n, \\ y &\rightarrow SB^{-1} \circ SR^{-1} \circ MC^{-1}(y) \oplus E_k(y) \quad y \in T_P. \end{aligned} \quad (18)$$

通过定义函数 f_1 和 f_2 , 将寻找明文对 (P, P') 的过程转化为寻找 x_0 和 y_0 使 $f_1(x_0) = f_2(y_0)$ 成立的问题. 对此问题, 文献[17]中使用BHT算法对其进行了量子化, 其结果表明通过使用 $\mathcal{O}(2^{n/3})$ 规模的qRAM和 $\mathcal{O}(n)$ 规模的量子比特, 可以在查询复杂度和时间复杂度均为 $\mathcal{O}(2^{n/3})$ 的条件下, 以63%的概率恢复出密钥 k .

针对寻找上述函数 f_1 和 f_2 的claw问题, 本文基于CNS提出了算法1. 该算法在无需qRAM的条件下能够找到这两个函数的一个claw.

对于1K-AES的量子滑动攻击, 使用算法1计算得到 (x_0, y_0) 后即可推导出候选密钥 $k = x_0 \oplus SB^{-1} \circ SR^{-1} \circ MC^{-1}(y_0)$. 之后, 对候选密钥进行加密验证即可恢复出正确的密钥 k .

算法1的详细流程如下:

步骤(1): 构造表 L

通过使用Grover($H, 1$)构造表 L . 对于 n 比特值 $z \in \mathbb{F}_2^n$, 函数 H 的定义如下:

$$H(z) = \begin{cases} 1, & z = (0_r \parallel c), \\ 0, & \text{其他}, \end{cases} \quad (19)$$

其中, $0_r = (00 \dots 0) \in \mathbb{F}_2^r$, $c \in \mathbb{F}_2^{n-r}$, 符号 $\parallel$ 表示两个比特串的级联. 设式(19)对应的量子算符为 O_H . 由于 $|S_r^{f_2}| = 2^{n-r}$, 所以执行一次Grover($H, 1$)得到集合 $S_r^{f_2}$ 中的元素时间复杂度为 $\mathcal{O}(2^{r/2})$, 由于表 L 的大小为 2^{t-r} , 所以此步骤的时间复杂度为 $\mathcal{O}(2^{t-r/2})$.

步骤(2): 执行QAA($\mathcal{A}, \mathcal{B}$)

(i) 构建初态 $|\phi_r\rangle$, 即使用 $\mathcal{A} = \text{Grover}(H, 1)$ 进行构建, 流程如下.

(i.1) 初始化寄存器为: $|0^{\otimes n}\rangle_1 |0^{\otimes n}\rangle_2 |0\rangle_3 |-\rangle_4$.

(i.2) 对寄存器1(R_1)使用 n 个Hadamard门:

$$\frac{1}{2^{n/2}} \sum_{x \in \{0, 1\}^n} |x\rangle_1 |0^{\otimes n}\rangle_2 |0\rangle_3 |-\rangle_4. \quad (20)$$

(i.3) 查询量子预言机 O_{f_1} , 得到当前状态:

$$\frac{1}{2^{n/2}} \sum_{x \in \{0, 1\}^n} |x\rangle_1 |f_1(x)\rangle_2 |0\rangle_3 |-\rangle_4. \quad (21)$$

(i.4) 重复下列步骤 $\mathcal{O}(2^{r/2})$ 次:

(a) 首先对 R_2 和 R_3 进行 O_H 的查询:

算法1

输入: 两个任意函数 $f_1, f_2: \{0, 1\}^n \rightarrow \{0, 1\}^n$, 其中对于 f_1, f_2 有对应的量子预言机 O_{f_1} 和 O_{f_2} .

输出: 找到函数 f_1 和 f_2 的一个claw, 即 (x_0, y_0) .

设置参数 r 和 t . 对于函数 f_2 , 定义集合 $S_r^{f_2} = \{(y, f_2(y))\}$, 其中 $S_r^{f_2}$ 中每条记录的第二项满足前 r 位为0比特串. 以同样的方式定义集合 $S_r^{f_1} = \{(x, f_1(x))\}$.

(1) 根据 $S_r^{f_2}$ 中的元素构造表 L , 表的大小为 $|L| = 2^{t-r}$. 定义成员检测函数为 f_L , 若存在 $(y, f_1(x)) \in L$, 则 $f_L(f_1(x)) = 1$, 否则 $f_L(f_1(x)) = 0$, O_{f_L} 算符为 f_L 的量子实现.

(2) 计算 $x = \text{QAA}(\mathcal{A}, \mathcal{B})$.

$$\mathcal{A} \text{ 算符用于构建初态 } |\phi_r\rangle = \frac{1}{\sqrt{|S_r^{f_1}|}} \sum_{x \in S_r^{f_1}} |x, f_1(x)\rangle = \frac{1}{\sqrt{|S_r^{f_1}|}} \sum_{x \in S_r^{f_1}} |x\rangle_1 |f_1(x)\rangle_2.$$

$$\mathcal{B} = O_{f_L}, \text{ 用于检测 } (y, f_1(x)) \text{ 是否属于表 } L: O_{f_L}(|x, f_1(x)\rangle |b\rangle) = |x, f_1(x)\rangle |b \oplus f_L(f_1(x))\rangle.$$

(3) 对步骤(2)输出的 x , 计算 $f_1(x)$, 判断 $(y, f_1(x))$ 是否属于表 L . 若属于, 则得到 $x_0 = x$, $y_0 = y$, 输出 (x_0, y_0) 作为函数 f_1 和 f_2 的一个claw.

$$\frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x\rangle_1 |f_1(x)\rangle_2 |H(f_1(x))\rangle_3 |-\rangle_4, \quad (22)$$

(b) 然后对 R_3 和 R_4 执行 CNOT 门操作, 其中 R_3 作为控制位, 用于改变 R_1 中“good”态的符号.

(c) 再次对 R_2 和 R_3 执行 O_H 的查询, 以恢复 R_3 为 $|0\rangle$ 态.

$$(d) \text{ 对 } R_1 \text{ 和 } R_2 \text{ 执行 } (2|\psi\rangle\langle\psi| - I), \text{ 其中 } |\psi\rangle = \frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x\rangle |f_1(x)\rangle.$$

通过步骤 (i), 可以得到态 $|\phi\rangle = |\text{good}\rangle + |\text{bad}\rangle = |\phi_r\rangle + |\text{bad}\rangle = \frac{1}{\sqrt{|S_r^{f_1}|}} \sum_{x \in S_r^{f_1}} |x\rangle_1 |f_1(x)\rangle_2 + |\text{bad}\rangle$. 由于接下来将对 $|\phi_r\rangle$ 这个希尔伯特空间进行操作, 依据文献[24]中的引理1, 可以得知后续的操作不会改变 $|\text{bad}\rangle$ 中振幅的绝对值, 因此可以将其视为得到了态 $|\phi_r\rangle$. 操作 (i) 的时间复杂度为 $|A|_{RT} = 2^{r/2}$.

(ii) 进行 QAA 的迭代, 由于集合 $S_r^{f_1}$ 中共有 2^{n-r} 个元素, 解的个数为 $|L| = 2^{t-r}$, 因此迭代的次数为 $\sqrt{2^{n-r}/2^{t-r}} = \sqrt{2^{n-t}} = 2^{(n-t)/2}$.

$$(ii.1) \text{ 对 } |\phi_r\rangle = \frac{1}{\sqrt{|S_r^{f_1}|}} \sum_{x \in S_r^{f_1}} |x\rangle_1 |f_1(x)\rangle_2 |0\rangle_3 |-\rangle_4$$

使用成员检测算符 O_{f_L} 作用于 R_2 和 R_3 上, 得到

$$\frac{1}{\sqrt{|S_r^{f_1}|}} \sum_{x \in S_r^{f_1}} |x\rangle_1 |f_1(x)\rangle_2 |f_L(f_1(x))\rangle_3 |-\rangle_4. \quad (23)$$

由定义1可知当忽略多项式级别的因子时, O_{f_L} 算符的时间复杂度为 $\mathcal{O}(2^{t-r})$.

(ii.2) 对 R_3 和 R_4 使用 CNOT 门, R_3 为控制比特, 用于修改 R_1 中“good”态振幅的符号.

(ii.3) 再次执行步骤 (ii.1) 的操作以恢复 R_3 为 $|0\rangle$ 态.

$$(ii.4) \text{ 对 } R_1 \text{ 和 } R_2 \text{ 执行 } (2|\psi\rangle\langle\psi| - I), \text{ 其中 } |\psi\rangle = \frac{1}{\sqrt{|S_r^{f_1}|}} \sum_{x \in S_r^{f_1}} |x\rangle |f_1(x)\rangle.$$

此步骤的时间复杂度由构建态 $|\psi\rangle$ 所需的时间决定, 即由算符 A 的时间复杂度决定, 所以此步骤的时间复杂度为 $\mathcal{O}(2^{r/2})$.

综上所述可以得到算法1的时间复杂度为

$$2^{(n-t)/2} (2^{r/2} + 2^{t-r}) + 2^{t-r/2}, \quad (24)$$

当取 $t = 3n/5$, $r = 2n/5$ 时, 式(24)取得最小值. 此时, 时间复杂度为 $\mathcal{O}(2^{2n/5})$. 对于任意的两个元素 $(x, f_1(x)) \in S_r^{f_1}$ 和 $(y, f_2(y)) \in S_r^{f_2}$, $f_1(x) = f_2(y)$ 的概率为 $2^{-(n-r)}$, 由于数据集包含 $2^{t-r} \cdot 2^{n-r} = 2^{n+t-2r}$ 对 $(x, f_1(x))$ 和 $(y, f_2(y))$, 因此算法1的成功率为

$$p = 1 - (1 - 2^{-(n-r)})^{2^{n+t-2r}} \approx 1 - (e^{-1})^{\frac{n+t-2r}{n-r}}, \quad (25)$$

当取 $t = 3r/2 = 3n/5$ 时, 可以得到 $p \approx 73.64\%$. 所以算法1通过使用 $\mathcal{O}(2^{n/5})$ 规模的经典存储和 $\mathcal{O}(n)$ 规模的量子比特, 可以在 $\mathcal{O}(2^{2n/5})$ 的时间和查询复杂度下以 73.64% 的概率恢复出密钥 k .

3.2 选择明文下的经典滑动攻击

无论是文献[10]中对 1K-AESf 的经典滑动攻击, 还是文献[17]及第3.1节中对 1K-AESf 的量子滑动攻击, 都无法以 100% 的概率成功. 其主要原因在于无法保证随机选择的明文集合 $\mathcal{T}_P$ 中一定存在一个滑动对, 即集合 $\mathcal{T}_P$ 中可能没有明文对满足式(14).

根据文献[12]中使用的引导性明文结构, 本节采用选择明文集合 $\mathcal{T}_P = \{P_i\}_{i=1,2,\dots,2^{n/2}}$, $\mathcal{T}_Q = \{Q_j\}_{j=1,2,\dots,2^{n/2}}$ 对 1K-AESf 进行经典滑动攻击, 如图2所示.

集合 $\mathcal{T}_P$ 中所有元素的后 $n/2$ 比特均为 0 比特串, 集合 $\mathcal{T}_Q$ 中的所有元素与集合 $\mathcal{T}_Q^- = \{\bar{Q}_j\}_{j=1,2,\dots,2^{n/2}}$ 中的元素满足关系:

$$Q_j = MC \circ SR \circ SB(\bar{Q}_j). \quad (26)$$

集合 $\mathcal{T}_Q^-$ 中所有元素的前 $n/2$ 比特均为 0 比特串. 通过定义明文集合 $\mathcal{T}_P$ 和 $\mathcal{T}_Q$ 可以保证 $P_i \oplus \bar{Q}_j$ 一定能遍历到正确的密钥, 而这一思路在文献[25]对 PRINCE 密码[26]的相关密钥攻击中也有体现. 本文的算法2描述了对 1K-AESf 成功率为 100% 的经典滑动攻击.

在选择明文条件下, 算法2可以在数据复杂度、存储复杂度和时间复杂度均为 $\mathcal{O}(2^{n/2})$ 的条件下, 以 100% 的概率恢复出密钥 k .

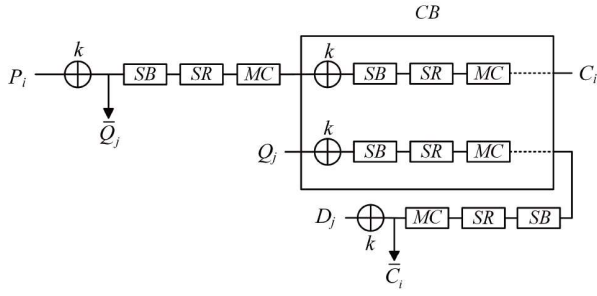


图 2 对 1K-AESf 成功率为 100% 的滑动攻击

Figure 2 Slide attack with 100% success rate on 1K-AESf.

算法 2

- (1) 对集合 $\mathcal{T}_P$ 中的每个元素 P_i 进行如下的处理:
 - 访问 E_k 得到对应的密文 C_i ;
 - 计算 $P_i \oplus C_i$, $\bar{C}_i = MC \circ SR \circ SB(C_i)$;
 - 根据 $P_i \oplus \bar{C}_i$ 将记录 $(P_i \oplus \bar{C}_i, P_i)$ 存储在表 L 中.
- (2) 对集合 $\mathcal{T}_Q$ 中的每个元素 Q_j 进行如下的处理:
 - 访问 E_k 得到对应的密文 D_j ;
 - 计算 $\bar{Q}_j \oplus D_j$, $\bar{Q}_j = SB^{-1} \circ SR^{-1} \circ MC^{-1}(Q_j)$;
 - 若表 L 中存在索引 i 使得 $\bar{Q}_j \oplus D_j = P_i \oplus \bar{C}_i$, 则对候选密钥 $k = P_i \oplus \bar{Q}_j$ 进行加密验证.

3.3 需要 qRAM 的量子滑动攻击

引导性明文结构同样可以用于将 1K-AESf 的量子滑动攻击成功率提升至 100%. 设明文集合 $\mathcal{T}_P = \{P_i\}_{i=1,2,\dots,2^{n-r}}$ 中所有元素的后 r 比特均为 0 比特串, 集合 $\mathcal{T}_{\bar{Q}} = \{\bar{Q}_j\}_{j=1,2,\dots,2^r}$ 中所有元素的前 $n-r$ 比特均为 0 比特串, 集合 $\mathcal{T}_Q$ 和 $\mathcal{T}_{\bar{Q}}$ 中的元素满足式 (26). 采用与第 3.1 节同样的分析方法, 在使用引导性明文结构的条件下, 寻找集合 $\mathcal{T}_P$ 和 $\mathcal{T}_Q$ 中的滑动对问题可以转化为寻找函数 f_1 和 f_2 的一个 claw, 这两个函数的定义如下:

$$\begin{cases} f_1 : \{0, 1\}^n \rightarrow \{0, 1\}^n, \\ x \rightarrow x \oplus MC \circ SR \circ SB(E_k(x)), x \in \mathcal{T}_P, \\ f_2 : \{0, 1\}^n \rightarrow \{0, 1\}^n, \\ y \rightarrow SB^{-1} \circ SR^{-1} \circ MC^{-1}(y) \oplus E_k(y), y \in \mathcal{T}_Q. \end{cases} \quad (27)$$

需要注意的是, 式 (27) 中定义的两个函数定义域的大小不一致. 函数 f_1 的定义域大小为 $|\mathcal{T}_P| = 2^{n-r}$, 而函数 f_2 中定义域大小为 $|\mathcal{T}_Q| = 2^r$, 这里假设 $|\mathcal{T}_Q| \leq |\mathcal{T}_P|$. 而文献 [17] 中使用 BHT 算法寻找两个函数的 claw 前提为两个函数的定义域大小相等. 因此, 本文基于 Buhrman 等人 [27] 的通用 claw 查找算法 (Algorithm Generic claw-finder), 提出了算法 3 用于寻找式 (27) 中定义的两个函数的 claw.

根据算法 3 的输出 (P_i, Q_j) , 可以计算出候选密钥 $k = P_i \oplus SB^{-1} \circ SR^{-1} \circ MC^{-1}(Q_j)$ 并对其进行加密验证.

分析算法 3 的复杂度: 步骤 (2) 中, 表 L 使用 qRAM 进行存储, 对表 L 进行排序的时间复杂度为 $\mathcal{O}(\ell \log \ell)$, 加上对 $\mathcal{T}_{QS}$ 中元素进行加密的时间, 可得步骤 (2) 中的时间复杂度为 $\mathcal{O}(\ell + \ell \log \ell)$. 根据对函数 f_1 和 f_2 的定义可知两个函数只有一个 claw, 所以步骤 (3) 的迭代次数为 $\mathcal{O}(\sqrt{\ell^2})$, 时间复杂度为 $\mathcal{O}(\ell(1 + \log \ell))$, 对函数 H 预言机的构造可以使用文献 [17] 中的实现方案. 由于步骤 (1)–(3) 输出一个 claw 的概率为 $\ell^3 / 2^{n+1}$, 所以步骤 (4) 的迭代次数为 $\mathcal{O}(\sqrt{2^{n+1} / \ell} / \ell)$, 总的时间复杂度为

$$\mathcal{O}\left(\left(\sqrt{2^{n+1} / \ell} / \ell\right) \times [\ell + \ell \log \ell + \ell(1 + \log \ell)]\right), \quad (29)$$

算法 3

- (1) 从集合 $\mathcal{T}_Q$ 和 $\mathcal{T}_P$ 中随机选择元素构成集合 $\mathcal{T}_{QS}$ 和 $\mathcal{T}_{PS}$, 两个集合满足 $\mathcal{T}_{QS} \subseteq \mathcal{T}_Q$, $|\mathcal{T}_{QS}| = \ell$, $\mathcal{T}_{PS} \subseteq \mathcal{T}_P$, $|\mathcal{T}_{PS}| = \ell^2$.
- (2) 构建一个大小为 ℓ 的表 L , 对集合 $\mathcal{T}_{QS}$ 中的每个元素计算 $f_2(Q_j)$, 并存储记录 $(Q_j, f_2(Q_j))$. 然后, 根据每条记录的第二项进行排序, 其中表 L 中的数据使用 qRAM 进行存储.
- (3) 计算 $P_i = QAA(H, 1)$, 其中定义 $H: \mathcal{T}_{PS} \rightarrow \{0, 1\}$ 函数为

$$H(P_i) = \begin{cases} 1, & (Q_j, f_1(P_i)) \in L, \\ 0, & (Q_j, f_1(P_i)) \notin L. \end{cases} \quad (28)$$
- (4) 对步骤 (1)–(3) 应用振幅放大.
- (5) 步骤 (4) 输出 P_i , 根据 $(Q_j, f_1(P_i))$ 在表 L 中找到对应的记录输出 (P_i, Q_j) 为函数 f_1 和 f_2 的一个 claw.

式(29)忽略对数因子可得 $\mathcal{O}(\sqrt{2^{n+1}/\ell})$, ℓ 取最大值时算法3的时间复杂度最小. 根据文献[27]中的结论, 为使查询复杂度最小, 取 $\ell = \min\{|T_Q|, \sqrt{|T_P|}\} = \min\{2^r, 2^{(n-r)/2}\}$. 所以当取 $|T_Q| = \sqrt{|T_P|}$, 即 $r = n/3$ 时 ℓ 取得最大值 $2^{n/3}$. 此时步骤(1)–(3)输出一个claw的概率至少为50%, 由文献[27]可知由于步骤(1)–(3)产生了恒定的成功概率, 所以算法3中步骤(4)无需执行.

因此, 算法3在使用引导性明文结构的条件下, 通过使用 $\mathcal{O}(2^{n/3})$ 规模的qRAM和 $\mathcal{O}(n)$ 规模的量子比特, 可以在 $\mathcal{O}(2^{n/3})$ 的时间和查询复杂度下以100%的概率恢复出密钥 k .

4 1K-AES_t的滑动攻击

文献[12]对1K-AES_t的滑动攻击如图3所示, 其中对于一个明文 P , 1K-AES_t的加密函数表示为 $E_k(P) = \text{ARK} \circ \text{SR} \circ \text{SB} \circ \text{ARK} \circ f_k \circ f_k \circ \dots \circ f_k(P)$, 文献[12]对1K-AES_t的滑动攻击思路如下.

首先定义集合 $T_P = \{P_i\}$, $T_Q = \{Q_j\}$, $T_{\bar{Q}} = \{\bar{Q}_j\}$, $T_R = \{R_i\}$ 和 $T_S = \{S_j\}$, 所有集合的大小都为 $2^{n/2}$, 常量 $a = (0_{96} \parallel 1_{32})$, 0_{96} 表示长为96比特的0比特串. 集合 T_P 中所有元素的后 $n/2$ 比特均为0比特串, $T_{\bar{Q}}$ 中所有元素的前 $n/2$ 比特均为0比特串, $T_{\bar{Q}}$ 与 T_Q 中的所有元素满足式(26), 集合 T_R 中的元素 $R_i = P_i \oplus a$, 集合 T_S 中的元素 $S_j = \text{MC} \circ \text{SR} \circ \text{SB}(\bar{Q}_j \oplus a)$. P_i , R_i , Q_j 和 S_j 对应的密文为 C_i , C_i' , D_j 和 D_j' .

根据上述集合的定义, Dunkelman等人[12]得出了

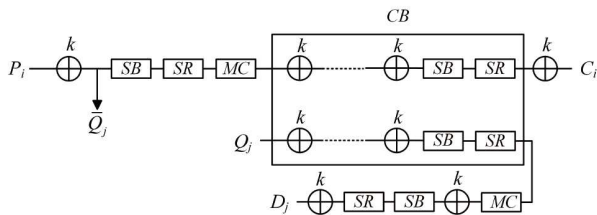


图3 对1K-AES_t的滑动攻击
Figure 3 Slide attack on 1K-AES_t.

两个结论.

结论1^[12] 其他滑动对的获取

(P_i, Q_j) 为一个滑动对 $\Leftrightarrow (R_i, S_j)$ 为一个滑动对.

结论2^[12] 密钥信息的获取

设 P_i 为滑动对的第一项, 则可以得到候选密钥信息的前 $n/2$ 比特位.

根据结论1可以得到下列等式:

$$D_j = \text{ARK} \circ \text{SR} \circ \text{SB} \circ \text{ARK} \circ \text{MC} \circ \text{ARK}(C_i), \quad (30)$$

$$D_j' = \text{ARK} \circ \text{SR} \circ \text{SB} \circ \text{ARK} \circ \text{MC} \circ \text{ARK}(C_i'), \quad (31)$$

对式(30)和(31)进行处理得到

$$\begin{aligned} f_1(P_i) &= \text{SR} \circ \text{SB} \circ \text{ARK} \circ \text{MC} \circ \text{ARK}(E_k(P_i)) \\ &\quad \oplus \text{SR} \circ \text{SB} \circ \text{ARK} \circ \text{MC} \circ \text{ARK}(E_k(R_i)), \quad (32) \\ R_i &= P_i \oplus a, \end{aligned}$$

$$\begin{aligned} f_2(Q_j) &= E_k(Q_j) \oplus E_k(S_j), \\ S_j &= \text{MC} \circ \text{SR} \circ \text{SB}(a \oplus \text{SB}^{-1} \circ \text{SR}^{-1} \circ \text{MC}^{-1}(Q_j)). \quad (33) \end{aligned}$$

根据式(32)和(33)定义的两个函数, 将 (P_i, Q_j) 滑动对的寻找问题转化为了求解函数 f_1 和 f_2 的一个claw. 但由于函数 f_1 需要密钥 k 才能进行计算. 而由结论2可得候选 $k_{\text{Col}(0,1)} = (P_i)_{\text{Col}(0,1)}$, $\text{Col}(0,1)$ 表示字符串的前8字节. 所以 $\text{SR} \circ \text{SB} \circ \text{ARK} \circ \text{MC} \circ \text{ARK}(\cdot)_{\text{SR}(\text{Col}(0,1))}$ 可以计算得到, $\text{SR}(\text{Col}(0,1))$ 表示前8字节行移位后字节所在位置的数值. 由于 Q_j 和 $\bar{Q}_j$ 一一对应, 为了方便后续的分析, 本节的后续部分定义滑动对为 $(P_i, \bar{Q}_j)$. 函数 f_1 和 f_2 可以重新进行如下的定义:

$$\begin{cases} f_1 : \{0,1\}^n \rightarrow \{0,1\}^{n/2}, \\ x \rightarrow \left(\text{SR} \circ \text{SB} \circ \text{ARK} \circ \text{MC} \circ \text{ARK}(E_k(x)) \oplus \text{SR} \right. \\ \quad \left. \circ \text{SB} \circ \text{ARK} \circ \text{MC} \circ \text{ARK}(E_k(x \oplus a)) \right)_{\text{SR}(\text{Col}(0,1))}, \\ x \in T_P, \\ f_2 : \{0,1\}^n \rightarrow \{0,1\}^{n/2}, \\ y \rightarrow \left(E_k(\text{MC} \circ \text{SR} \circ \text{SB}(y)) \right. \\ \quad \left. \oplus E_k(\text{MC} \circ \text{SR} \circ \text{SB}(y \oplus a)) \right)_{\text{SR}(\text{Col}(0,1))}, \\ y \in T_{\bar{Q}}. \end{cases} \quad (34)$$

根据式(34)的定义可知, 函数 f_1 和 f_2 的claw约有 $2^{n/2}$

算法4

- (1) 从集合 $\mathcal{T}_{\bar{Q}}$ 和 $\mathcal{T}_P$ 中随机选择元素构成集合 $\mathcal{T}_{\bar{Q}S}$ 和 $\mathcal{T}_{PS}$, 两个集合满足 $\mathcal{T}_{\bar{Q}S} \subseteq \mathcal{T}_{\bar{Q}}$, $|\mathcal{T}_{\bar{Q}S}| = \ell$, $\mathcal{T}_{PS} \subseteq \mathcal{T}_P$, $|\mathcal{T}_{PS}| = \ell^2$.
- (2) 构建一个大小为 ℓ 的表 L , 对集合 $\mathcal{T}_{\bar{Q}S}$ 中的每个元素计算 $f_2(\bar{Q}_j)$, 并存储记录 $(\bar{Q}_j, f_2(\bar{Q}_j))$. 然后, 根据每条记录的第二项进行排序, 其中表 L 中的数据使用qRAM进行存储.
- (3) 计算 $P_i = \text{QAA}(H, 1)$, 其中定义 $H: \mathcal{T}_{PS} \rightarrow \{0, 1\}$, m 为一任意的明文, $c = E_k(m)$. 函数 H 会进行如下的两次判断:
 - 若 $(\bar{Q}_j, f_1(P_i)) \notin L$, $H(P_i) = 0$. 否则计算 $k' = P_i \oplus \bar{Q}_j$;
 - 对于密钥 k' 计算 $c = E_{k'}(m)$ 是否成立, 若成立, 输出 $H(P_i) = 1$, 否则 $H(P_i) = 0$.
- (4) 对步骤(1)–(3)应用振幅放大.
- (5) 步骤(4)输出 P_i , 根据 $(\bar{Q}_j, f_1(P_i))$ 在表 L 中找到对应的记录输出 $(P_i, \bar{Q}_j)$ 为函数 f_1 和 f_2 的一个claw.

个, 即每一个 P_i 都有一个 $\bar{Q}_j$ 与之对应. 所以为了得到正确的密钥, 对于得到的每个候选密钥 $k = P_i \oplus \bar{Q}_j$ 需进行加密验证. 对1K-AESf使用引导性明文结构的滑动攻击的详细流程参考文献[12]中的第5节, 可以以 $\mathcal{O}(2^{n/2})$ 的数据复杂度、存储复杂度和时间复杂度, 以100%的概率恢复出密钥 k .

与算法3类似, 本文提出了算法4用于对1K-AESf的量子滑动攻击. 算法4中集合 $\mathcal{T}_P = \{P_i\}_{i=1,2,\dots,2^{n-r}}$ 满足其元素的后 r 比特均为0比特串, $\mathcal{T}_{\bar{Q}} = \{\bar{Q}_j\}_{j=1,2,\dots,2^r}$ 满足其元素的前 $n-r$ 比特均为0比特串, 其中为了符合结论2, 要求 $r < n/2$.

算法4的复杂度分析与算法3类似. 当 $r = n/3$ 时时间和查询复杂度最小, 通过使用 $\mathcal{O}(2^{n/3})$ 规模的qRAM和 $\mathcal{O}(n)$ 规模的量子比特, 可以在 $\mathcal{O}(2^{n/3})$ 的时间和查询复杂度下以100%的概率恢复出密钥 $k = P_i \oplus \bar{Q}_j$.

5 结论

本文通过使用滑动攻击密码分析技术主要对两种1K-AES结构的密码进行了量子密码分析. 针对这两种1K-AES的量子滑动攻击, 本文首先将问题转化为求解两个函数的claw问题, 然后利用CNS或通用claw查找算法寻找这两个函数的claw.

在1K-AESf的量子滑动攻击中, 文献[17]利用BHT算法, 在使用qRAM的情况下对1K-AESf进行了量子滑动攻击. 而本文的算法1可以在无需qRAM的情况下对1K-AESf进行量子滑动攻击, 虽然查询复杂度有所增

加, 但是使用的量子存储模型更加可行且将成功率提高了约11%. 此外, 通过结合引导性明文结构和通用claw查找算法, 本文所提出的算法3将量子条件下的成功率提升至100%, 且相比于文献[17]的攻击条件和复杂度均保持一致(见表1).

此外, 由于文献[10]对1K-AESf的经典滑动攻击成功率不能达到100%. 所以本文在此基础上增强了攻击的前提, 在选择明文攻击条件下, 利用引导性明文结构

表 1 1K-AESf的量子滑动攻击分析

Table 1 Quantum slide attack analysis of 1K-AESf

来源	量子存储模型	查询复杂度	qRAM复杂度	经典存储复杂度	成功率(%)
[17]	qRAM	$\mathcal{O}(2^{n/3})$	$\mathcal{O}(2^{n/3})$	—	63
本文(算法1)	经典存储	$\mathcal{O}(2^{2n/5})$	—	$\mathcal{O}(2^{n/5})$	74
本文(算法3)	qRAM	$\mathcal{O}(2^{n/3})$	$\mathcal{O}(2^{n/3})$	—	100

表 2 1K-AESf的经典滑动攻击分析

Table 2 Classical slide attack analysis of 1K-AESf

来源	攻击类型	时间复杂度	数据复杂度	存储复杂度	成功率(%)
[10]	已知明文攻击	$\mathcal{O}(2^{n/2})$	$\mathcal{O}(2^{n/2})$	$\mathcal{O}(2^{n/2})$	63
本文(算法2)	选择明文攻击	$\mathcal{O}(2^{n/2})$	$\mathcal{O}(2^{n/2})$	$\mathcal{O}(2^{n/2})$	100

表 3 1K-AESf的滑动攻击分析

Table 3 Slide attack analysis of 1K-AESf

文献	攻击类型	查询复杂度	qRAM复杂度	经典存储复杂度	成功率(%)
[12]	经典	$\mathcal{O}(2^{n/2})$	—	$\mathcal{O}(2^{n/2})$	100
[3]	量子	$\mathcal{O}(2^{n/2})$	—	—	100
本文(算法4)	量子	$\mathcal{O}(2^{n/3})$	$\mathcal{O}(2^{n/3})$	—	100

提出了算法2, 将1K-AESf的经典滑动攻击成功率提高到100% (见表2).

最后, 本文提出了算法4对1K-AESf的滑动攻击进行了量子化分析(见表3). 与经典攻击方法相比, 量子

化后的滑动攻击在查询复杂度和存储复杂度上取得了亚指数加速. 与使用Grover算法对1K-AESf进行密钥恢复攻击相比, 算法4的量子化方法可将查询复杂度降低到Grover算法的 $1/2^{n/6}$.

参考文献

- 1 Kuwakado H, Morii M. Quantum distinguisher between the 3-round Feistel cipher and the random permutation. In: Proceedings of the IEEE International Symposium on Information Theory. Austin: IEEE, 2010. 2682–2685
- 2 Simon D R. On the power of quantum computation. *SIAM J Comput*, 1997, 26: 1474–1483
- 3 Grover L K. A fast quantum mechanical algorithm for database search. In: Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing. New York: ACM, 1996. 212–219
- 4 Dong X Y. A survey of quantum cryptanalysis of symmetric cryptography (in Chinese). *J Cryptol Res*, 2024, 11: 159–173 [董晓阳. 对称密码的量子分析法综述. 密码学报, 2024, 11: 159–173]
- 5 Biryukov A, Wagner D. Slide attacks. In: International Workshop on Fast Software Encryption. Berlin, Heidelberg: Springer, 1999. 245–259
- 6 Kaplan M, Leurent G, Leverrier A, et al. Breaking symmetric cryptosystems using quantum period finding. In: Advances in Cryptology-CRYPTO 2016. Berlin, Heidelberg: Springer, 2016. 207–237
- 7 Dong X, Dong B, Wang X. Quantum attacks on some Feistel block ciphers. *Des Codes Cryptogr*, 2020, 88: 1179–1203
- 8 Biryukov A, Wagner D. Advanced slide attacks. In: International Conference on the Theory and Applications of Cryptographic Techniques. Berlin, Heidelberg: Springer, 2000. 589–606
- 9 Indestege S, Keller N, Dunkelman O, et al. A practical attack on KeeLoq. In: Advances in Cryptology-EUROCRYPT 2008. Berlin, Heidelberg: Springer, 2008. 1–18
- 10 Bar-On A, Biham E, Dunkelman O, et al. Efficient slide attacks. *J Cryptol*, 2018, 31: 641–670
- 11 Bonnetain X, Naya-Plasencia M, Schrottenloher A. On quantum slide attacks. In: International Conference on Selected Areas in Cryptography. Cham: Springer, 2019. 492–519
- 12 Dunkelman O, Keller N, Lasry N, et al. New slide attacks on almost self-similar ciphers. In: Advances in Cryptology-EUROCRYPT 2020. Cham: Springer, 2020. 250–279
- 13 Sliman L, Omrani T, Tari Z, et al. Towards an ultra lightweight block ciphers for internet of things. *J Inf Security Appl*, 2021, 61: 102897
- 14 Zhang K, Lai X, Wang L, et al. Slide attack on full-round ULC lightweight block cipher designed for IoT. *Secur Commun Netws*, 2022, 2022: 4291000
- 15 Jia K T, Dong X Y, Wei C M, et al. On the design of block cipher FESH (in Chinese). *J Cryptol Res*, 2019, 6: 713–726 [贾珂婷, 董晓阳, 魏淙滔, 等. 分组密码算法FESH. 密码学报, 2019, 6: 713–726]
- 16 Zheng Y, Wu W. Similarity property and slide attack of block cipher FESH. *IET Inf Secur*, 2023, 2023: 1–8
- 17 Cai B B, Wu Y, Dong J, et al. Quantum attacks on 1K-AES and PRINCE. *Comput J*, 2023, 66: 1102–1110
- 18 Brassard G, Høyer P, Tapp A. Quantum cryptanalysis of hash and claw-free functions. In: LATIN'98: Theoretical Informatics. Berlin, Heidelberg: Springer, 1998. 163–169
- 19 Hou X, Zhou G, Li Q, et al. A duplication-free quantum neural network for universal approximation. *Sci China-Phys Mech Astron*, 2023, 66: 270362
- 20 Song Y, Wu Y, Wu S, et al. A quantum federated learning framework for classical clients. *Sci China-Phys Mech Astron*, 2024, 67: 250311
- 21 Giovannetti V, Lloyd S, Maccone L. Quantum random access memory. *Phys Rev Lett*, 2008, 100: 160501
- 22 Chailloux A, Naya-Plasencia M, Schrottenloher A. An efficient quantum collision search algorithm and implications on symmetric cryptography. In: Advances in Cryptology-ASIACRYPT 2017. Cham: Springer, 2017. 211–240
- 23 National Institute of Standards and Technology. Specification for the Advanced Encryption Standard (AES). Technical Report. FIPS PUB 197, NIST. 2001

- 24 Brassard G, Hoyer P, Mosca M, et al. Quantum amplitude amplification and estimation. *Contemp Math*, 2002, 305: 53–74
- 25 Jean J, Nikolić I, Peyrin T, et al. Security analysis of PRINCE. In: *International Workshop on Fast Software Encryption*. Berlin, Heidelberg: Springer, 2013. 92–111
- 26 Borghoff J, Canteaut A, Güneysu T, et al. PRINCE—A low-latency block cipher for pervasive computing applications. In: *Advances in Cryptology-ASIACRYPT 2012*. Berlin, Heidelberg: Springer, 2012. 208–225
- 27 Buhrman H, Dürr C, Heiligman M, et al. Quantum algorithms for element distinctness. *SIAM J Comput*, 2005, 34: 1324–1330

Quantum slide attack on 1K-AES structure

SHEN Zhou¹, ZHANG FengRong^{1*}, WEI YongZhuang² & WANG BaoCang³

¹ School of Cyber Engineering, Xidian University, Xi'an 710071, China

² School of Computer Science and Information Security, Guilin University of Electronic Technology, Guilin 541004, China

³ State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China

*Corresponding author (email: zhangfengrong@xidian.edu.cn)

With the continuous development of quantum technology, the quantum security of symmetric ciphers has attracted widespread attention. This study focuses on the security of 1K-AES, employing both quantum and classical slide attacks to analyze its security. On the one hand, for the 1K-AES with MixColumns in the last round, using the CNS algorithm achieves a success rate of 74%, an improvement from 63%, under a query complexity of $\mathcal{O}(2^{2n/5})$ and classical memory complexity of $\mathcal{O}(2^{n/5})$ without the need for quantum random access memory, where n denotes the block size. When using a quantum random access memory, the universal quantum claw-finding algorithm improves the success rate to 100% with a query complexity of $\mathcal{O}(2^{n/3})$ and quantum random memory complexity of $\mathcal{O}(2^{n/3})$. Additionally, under the chosen plaintext attack, the success rate of the classical slide attack is also increased to 100%. On the other hand, for the 1K-AES with no MixColumns in the last round, the universal quantum claw-finding algorithm is used to perform a quantum slide attack, with both time and memory complexity of $\mathcal{O}(2^{n/3})$. Compared with the classical slide attack, this method achieves sub-exponential acceleration; compared with Grover's search, the query complexity reduces from $\mathcal{O}(2^{n/2})$ to $\mathcal{O}(2^{n/3})$.

1K-AES, slide attack, quantum claw-finding algorithm, CNS algorithm

PACS: 03.67.Dd, 03.67.Lx, 03.67.-a

doi: [10.1360/SSPMA-2024-0582](https://doi.org/10.1360/SSPMA-2024-0582)