软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2015,26(10):2684−2695 [doi: 10.13328/j.cnki.jos.004805]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

分组密码 TWINE 的中间相遇攻击*

汪艳凤[1,2], 吴文玲[1]

[1](中国科学院 软件研究所 可信计算与信息保障实验室,北京 100190)

[2](中国科学院 研究生院,北京 100049)

通讯作者: 汪艳凤, E-mail: wangyanfeng@tca.iscas.ac.cn

摘 要: 将 Biclique 初始结构与标准的三子集中间相遇攻击相结合,给出了一种普遍的中间相遇攻击模式.与 Biclique 分析相比,该模式下的攻击作为算法抗中间相遇攻击的结果更为合理.进一步地,评估了算法 TWINE 抗中间相遇攻击的能力,通过合理选择中立比特位置以及部分匹配位置,给出了 18 轮 TWINE-80 以及 22 轮 TWINE-128 算法的中间相遇攻击结果.到目前为止,这是 TWINE 算法分析中数据复杂度最小的攻击结果.

关键词: 分组密码;TWINE;中间相遇攻击;Biclique;数据复杂度

中图法分类号: TP309

中文引用格式: 汪艳凤,吴文玲.分组密码 TWINE 的中间相遇攻击.软件学报,2015,26(10):2684−2695. http://www.jos.org.cn/1000-9825/4805.htm

英文引用格式: Wang YF, Wu WL. Meet-in-the-Middle attack on TWINE block cipher. Ruan Jian Xue Bao/Journal of Software, 2015,26(10):2684−2695 (in Chinese). http://www.jos.org.cn/1000-9825/4805.htm

Meet-in-the-Middle Attack on TWINE Block Cipher

WANG Yan-Feng[1,2], WU Wen-Ling[1]

[1](Trusted Computing and Information Assurance Laboratory, Institute of Software, The Chinese Academy of Sciences, Beijing 100190, China)

[2](Graduate University, The Chinese Academy of Sciences, Beijing 100049, China)

Abstract: This paper presents a general structure of meet-in-the-middle attack by combing the advantages of Biclique and three sub-set meet-in- the-middle. Compared with the Biclique cryptanalysis proposed in Asiacrypt 2011, this attack model is more reasonable to be regarded as the security of one block cipher against meet-in-the-middle attack. Moreover, the study evaluates the security of TWINE against meet-in-the- middle attack and gives attacks on 18-round TWINE-80 and 22-round TWINE-128. Meanwhile, the data complexities of these attacks are the least among the precious attacks on TWINE.

Key words: block cipher; TWINE; meet-in-the-middle; Biclique; data complexity

中间相遇攻击方法最早由 Diffie 和 Hellman 提出[1],之后被广泛应用于 Hash 函数的安全性分析.Aoki 及 Sasaki 使用该方法给出了基于 AES 算法的 Hash 函数的原像攻击,并应用到 Whirlpool 算法中[2].除此之外,该分析方法首次给出了 MD5 及 Tiger 算法的全轮的原像攻击[3,4]、SHA-1 及 SHA-2 算法的最好的原像攻击[5,6]以及 SHA-2 的伪碰撞攻击[7].该分析方法在分析 Hash 函数的过程中,分析人员提出了各种分析技巧,比如剪接技术、初始结构技巧以及部分匹配技术[6,8].近年来,很多密码学分析者将该方法应用到分组密码的安全性分析中[9,10],并提出了三子集中间相遇攻击[11].

三子集中间相遇攻击基于的事实是:密码算法的部分中间状态既可以从明文出发向后计算得到,也可以从密文出发向前计算得到.同时,两部分的计算过程中都不覆盖全部的主密钥信息.如此,攻击者可以在优于穷搜

* 基金项目: 国家重点基础研究发展计划(973)(2013CB338002); 国家自然科学基金(61272476, 61232009, 61202420)
收稿时间: 2013-09-02; 修改时间: 2014-10-23; 定稿时间: 2014-12-18

攻击的时间内恢复主密钥信息.该攻击方法首次给出了 KATAN 族算法全轮的攻击方案.此外,Hash 函数分析中的初始结构技巧以及部分匹配技巧同样可以应用于分组密码算法的分析中.一种广泛应用的初始结构被称为 Biclique.备受关注的 Biclique 分析将 Biclique 初始结构与中间相遇相结合,给出了全轮 AES 算法的攻击结果[12].但是,该攻击方法在中间相遇的计算过程中遍历了全密钥空间,借助时间存储折中技巧,通过预计算并存储部分计算过程,其他密钥下的计算过程只重复计算与存储过程不同的非线性部分,以此达到微优于穷举攻击的单密钥攻击方案.正如文献[13]中所言,Biclique 分析方法可以看作是一种加速后的穷举攻击,可以给出任意算法的全轮攻击并使时间复杂度略优于穷举攻击,如 Present[14],LED,mCrypton[15],LBlock[16].因此,将 Biclique 分析的结果作为算法抗中间相遇攻击的安全性评估结果有点牵强.

为了避开中间相遇过程的穷搜密钥情形,本文将 Biclique 初始结构直接与标准的三子集中间相遇过程结合.当然,与加速后的穷举方法相比,该分析方法下攻击轮数较少,但是该方法下的评估结果作为算法抗中间相遇攻击的能力更有说服力.此外,Biclique 结构与三子集部分匹配结合的技巧曾经给出了 14 轮的 Piccolo-80 的攻击过程[17];但是该攻击过程中的 Biclique 位于中间位置,因此导致整个攻击过程需要全密码本的数据量,攻击成功的意义明显下降.为了避免此类情况的发生,在本文提出的中间相遇方法中,将 Biclique 初始结构建立在明密文附近,以控制攻击的数据复杂度.

TWINE 算法[18]是一种轻量级分组密码算法,其分组长度为 64bit,密钥长度可以是 80bit 或者是 128bit.设计文档中,针对算法最好的攻击结果是 23 轮 TWINE-80 的不可能差分分析以及 24 轮的 TWINE-128 的不可能差分分析,两种攻击需要的数据量均在 2^{50} 以上.如上所述,Biclique 分析可以给出全轮 TWINE 算法时间复杂度略优于穷举攻击的结果[19],数据复杂度为 2^{60} 的选择明密文.由于在很多具体的协议环境中只有少量的明密文可以使用[20,21],因此,低数据量下的攻击备受关注.本文将 Biclique 初始结构与三子集中间相遇相结合,给出了 18 轮的 TWINE-80 以及 22 轮的 TWINE-128 的中间相遇攻击,攻击成功需要的选择明密文量分别为 2^{16} 和 2^{12}.这是现存的针对 TWINE 算法攻击中数据量最少的攻击结果.

1 TWINE 算法简介

轻量级分组密码 TWINE 算法发表在 2012 年 SAC 会议上,其分组长度为 64bit,密钥长度可以是 80bit 或 128bit.根据密钥长度的不同,算法分别记为 TWINE-80 以及 TWINE-128.下面简单介绍算法的加密过程以及密钥编排算法.

1.1 加密过程

TWINE 算法采用 16 分支的广义 Feistel 结构,两种密钥长度下,算法迭代轮数均为 36 轮.其轮函数包括 3 种操作:轮密钥加、4bit 的 S 盒变换以及拉线置换 P,具体过程如图 1 所示,其中,每个方块代表长度为 4bit 的单元.因为具体的 S 盒置换信息与本文攻击过程无关,所以这里不再详述 S 盒变换.

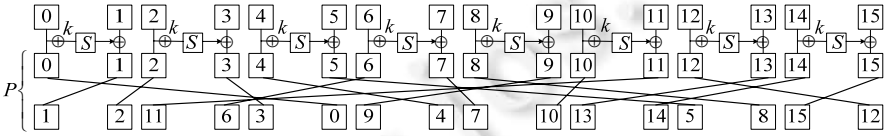

Fig.1 Round function of TWINE

图 1 TWINE 算法的轮函数

1.2 密钥编排算法

由图 1 可知,每轮加密过程需要 32bit 的轮子密钥.下面根据密钥长度的不同,分别介绍轮子密钥的生成过程.

1.2.1 TWINE-80 的密钥编排算法

将 80bit 的主密钥 K 以 4bit 为单位分为 20 块,迭代更新后选择 8 个块(共 32bit)作为轮子密钥 RK 嵌入加密过程,具体生成算法为:

$K_0 \| K_1 \| ... \| K_{18} \| K_{19} \leftarrow K$

$RK_0^1 \leftarrow K_1, RK_1^1 \leftarrow K_3, RK_2^1 \leftarrow K_4, RK_3^1 \leftarrow K_6, RK_4^1 \leftarrow K_{13}, RK_5^1 \leftarrow K_{14}, RK_6^1 \leftarrow K_{15}, RK_7^1 \leftarrow K_{16}$

$RK_{(32)}^1 \leftarrow RK_0^1 \| RK_1^1 \| ... \| RK_6^1 \| RK_7^1$

for $i \leftarrow 2$ to 36

$K_1 \leftarrow K_1 \oplus S(K_0)$

$K_4 \leftarrow K_4 \oplus S(K_{16})$

$K_7 \leftarrow K_7 \oplus 0 \| CON_H^{i-1}$

$K_{19} \leftarrow K_{19} \oplus 0 \| CON_L^{i-1}$

$temp_0 \leftarrow K_0, temp_1 \leftarrow K_1, temp_2 \leftarrow K_2, temp_3 \leftarrow K_3$

for $j \leftarrow 0$ to 3

$K_{j*4} \leftarrow K_{j*4+4}, K_{j*4+1} \leftarrow K_{j*4+5}$

$K_{j*4+2} \leftarrow K_{j*4+6}, K_{j*4+3} \leftarrow K_{j*4+7}$

$K_{16} \leftarrow temp_1, K_{17} \leftarrow temp_2, K_{18} \leftarrow temp_3, K_{19} \leftarrow temp_0$

$RK_0^i \leftarrow K_1, RK_1^i \leftarrow K_3, RK_2^i \leftarrow K_4, RK_3^i \leftarrow K_6, RK_4^i \leftarrow K_{13}, RK_5^i \leftarrow K_{14}, RK_6^i \leftarrow K_{15}, RK_7^i \leftarrow K_{16}$

$RK_{(32)}^i \leftarrow RK_0^i \| RK_1^i \| ... \| RK_6^i \| RK_7^i$

$RK_{(32\times 36)} \leftarrow RK_{(32)}^1 \| RK_{(32)}^2 \| ... \| RK_{(32)}^{35} \| RK_{(32)}^{36}$

1.2.2 TWINE-128 的密钥编排算法

将 128bit 的主密钥 K 分为 32 块,迭代更新后每轮选择 32bit 作为相应轮子密钥,具体生成算法如下:

$K_0 \| K_1 \| ... \| K_{30} \| K_{31} \leftarrow K$

$RK_0^1 \leftarrow K_2, RK_1^1 \leftarrow K_3, RK_2^1 \leftarrow K_{12}, RK_3^1 \leftarrow K_{15}, RK_4^1 \leftarrow K_{17}, RK_5^1 \leftarrow K_{18}, RK_6^1 \leftarrow K_{28}, RK_7^1 \leftarrow K_{31}$

for $i \leftarrow 2$ to 36

$K_1 \leftarrow K_1 \oplus S(K_0)$

$K_4 \leftarrow K_4 \oplus S(K_{16})$

$K_{23} \leftarrow K_{23} \oplus S(K_{30})$

$K_7 \leftarrow K_7 \oplus 0 \| CON_H^{i-1}$

$K_{19} \leftarrow K_{19} \oplus 0 \| CON_L^{i-1}$

$temp_0 \leftarrow K_0, temp_1 \leftarrow K_1, temp_2 \leftarrow K_2, temp_3 \leftarrow K_3$

for $j \leftarrow 0$ to 6

$K_{j*4} \leftarrow K_{j*4+4}, K_{j*4+1} \leftarrow K_{j*4+5}$

$K_{j*4+2} \leftarrow K_{j*4+6}, K_{j*4+3} \leftarrow K_{j*4+7}$

$K_{28} \leftarrow temp_1, K_{29} \leftarrow temp_2, K_{30} \leftarrow temp_3, K_{31} \leftarrow temp_0$

$RK_0^i \leftarrow K_2, RK_1^i \leftarrow K_3, RK_2^i \leftarrow K_{12}, RK_3^i \leftarrow K_{15}, RK_4^i \leftarrow K_{17}, RK_5^i \leftarrow K_{18}, RK_{6(4)}^i \leftarrow K_{28}, RK_7^i \leftarrow K_{31}$

$RK_{(32)}^i \leftarrow RK_0^i \| RK_1^i \| ... \| RK_6^i \| RK_7^i$

$RK_{(32\times 36)} \leftarrow RK_{(32)}^1 \| RK_{(32)}^2 \| ... \| RK_{(32)}^{35} \| RK_{(32)}^{36}$

2 中间相遇攻击框架

本文中使用的中间相遇攻击方案将 Biclique 初始结构与三子集中间相遇相结合,前后向独立计算得到部分匹配位置的值,进而通过是否匹配来达到删减密钥的目的,从而形成单密钥下优于穷举的攻击方案.下文中提到

的前向(后向)过程的中立比特是指前向(后向)的计算过程与此比特无关.

文中使用到的中间相遇攻击的具体攻击模式如图 2 所示,过程描述如下:

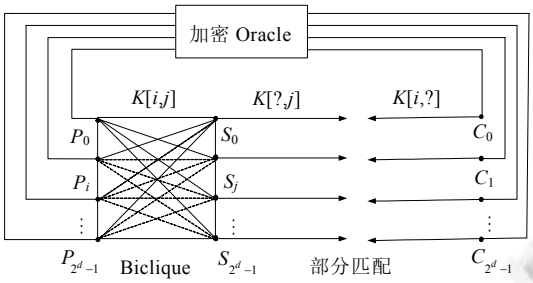

Fig.2 General structure of meet-in-the-middle attack

图 2 中间相遇攻击框架

- 第 1 阶段:中间相遇阶段.

1. 将 k 比特的主密钥分为 3 块,分别为 A_0,A_1 和 A_2,共用比特位置记为 A_0,前向过程和后向过程的中立比特分别为 A_1,A_2.一般情形下,A_1,A_2 的规模是一样的,我们记$|A_1|=|A_2|=d$,用 $i,j\in\{0,1\}^d$ 标识集合 A_1,A_2 中的各个密钥;
2. 遍历 A_0 的所有可能的值,此时主密钥中只有 A_1,A_2 未知,用主密钥 $K[i,j]$来表示 $A_1=i$,$A_2=j$:
   2.1 建立 Biclique 过程:
      I. 任选明文 P_0 在密钥 $K[0,0]$下加密得到 S_0 的值;
      II. 在密钥 $K[0,j]$下加密 P_0 得到 S_j 的值;与步骤 I 相比,这是密钥 A_2 活跃的一条差分传播路径;
      III. 在密钥 $K[i,0]$下解密 S_0 得到 P_i 的值;与步骤 I 相比,这是密钥 A_1 活跃的一条差分传播路径;
      IV. 如果两条差分路径没有相交的非线性组件,则成功建立 Biclique 结构.即:任意的 $i,j\in\{0,1\}^d$,均有 $P_i \xrightarrow{K[i,j]} S_j$;
   2.2 部分匹配过程:
      I. 遍历 S_j,在未知 A_1 的情形下,向前计算得到匹配位置 u 的值并将(S_j,u)保存到表 1 中;
      II. 询问加密 Oracle,得到 P_i 对应的密文 C_i.对每一个 C_i,在未知 A_2 的情形下,向后计算得到 v 的值,在表 1 中寻找是否有 $u=v$ 匹配:如果有,则将对应的(i,j)作为候选密钥.

Table 1 Key schedule of TWINE-80 influenced by neutral bits

表 1 中立比特对 TWINE-80 密钥编排算法的影响

轮数	第 8 轮中 K_3 活跃								第 8 轮中 K_1 活跃							
	RK_0	RK_1	RK_2	RK_3	RK_4	RK_5	RK_6	RK_7	RK_0	RK_1	RK_2	RK_3	RK_4	RK_5	RK_6	RK_7
0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
14	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
15	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
16	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1
17	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1

- 第 2 阶段:密钥穷搜阶段.

设密码算法总的密钥长度为 k,部分匹配位置的长度为 b,则上述过程结束之后,存活密钥个数为 2^{k-b}.借助已知的明密文对验证剩余候选密钥是否为正确的密钥值.

- 第 3 阶段:复杂度评估.

1. 数据复杂度.

攻击过程需要的数据复杂度取决于 Biclique 建立过程中需要的选择明文数目.更具体地说,在 A_1 活跃下,反向解密的差分传播路径在明文处的活跃比特数决定了整个攻击过程的数据复杂度.设此路径之后明文处活跃比特数为 m,则攻击的数据复杂度是 2^m 个选择明密文.

2. 时间复杂度.

整个攻击的时间复杂度主要分为两个部分:中间相遇阶段的时间代价以及密钥穷搜阶段的时间复杂度.

中间相遇阶段分为两步:建立 Biclique 以及部分匹配.假设 Biclique 建立轮数为 R_b 轮,前向部分匹配轮数为 R_{for} 轮,后向部分匹配轮数为 R_{back} 轮.针对每一个密钥集合,Biclique 的建立过程中使用独立的相关密钥差分路径,两部分的计算均不含有相交的非线性部分,因此两部分的计算代价的加和不超过 $2^d\times R_b$ 个轮计算;前向部分匹配的计算过程中,对 2^d 个状态 S 向前计算了 R_{for} 轮到达部分匹配位置.类似地,后向部分匹配过程中,2^d 个明文 C 向后计算了 R_{back} 轮得到部分匹配的值.因此,两部分的时间复杂度为 $2^d(R_{for}+R_{back})$个轮计算.以全轮的加密计算量为单位,一个密钥集合中间相遇过程的时间复杂度为

$$2^d\times(R_b+R_{for}+R_{back})/(R_b+R_{for}+R_{back})=2^d.$$

这一过程总共涉及到 $2^{|A_0|}$ 个密钥集合,因此,中间相遇阶段的时间复杂度不会超过 $2^{|A_0|+d}$.

密钥穷搜阶段将使用已有的明密文对候选密钥进行逐个验证,判断其是否为正确密钥.这一阶段的时间复杂度即为中间相遇阶段剩余的候选密钥量 2^{k-b}.

因此,整个攻击过程的时间复杂度为 $C=2^{|A_0|+d}+2^{k-b}$.因为全部密钥比特与中立比特长度之间有关系式 $k=|A_0|+2d$,因此,当 $d>1$ 时,此攻击方法比穷搜攻击有效.

3. 存储复杂度.

在攻击过程中,需要存储的部分为表 1 所示部分,规模为 $2^{\min\{|A_1|,|A_2|\}}=2^d$ 个 b 比特.

3 TWINE 算法的中间相遇攻击过程

3.1 TWINE-80算法的18轮攻击过程

若将初始主密钥作为恢复目标,中间相遇攻击只能攻击到 17 轮.为了达到更长的攻击轮数,我们将主密钥迭代更新至提取第 8 轮子密钥时的密钥状态作为恢复目标,共 20 个分块 80bit 的密钥信息.在此基础上,给出 18 轮的 TWINE-80 算法的中间相遇攻击结果.按照图 2 中的标识,我们选择后向匹配过程的中立比特 A_2 的活跃位置为第 8 轮等价主密钥下的第 3 个(计数从 0 开始)密钥分块.相应地,前向过程中的中立比特 A_1 为第 1 个密钥分块.

3.1.1 中立比特对密钥编排算法的影响

根据 TWINE-80 算法的密钥编排方案,在给定活跃主密钥比特的情形下,前向后向分别计算得知,从第 0 轮~第 17 轮(共 18 轮)的轮子密钥的活跃模式见表 1,表中 0 表示相应主密钥变化对其没有影响,1 表示会随对应密钥的变化而发生变化.

3.1.2 5 轮 Biclique 的建立

为了降低攻击的数据复杂度,我们固定 P_0 为全 0 比特明文,具体建立过程如图 3 所示.加密 P_0 得到 S_j 的值,是密钥 A_2 活跃的一条差分传播路径,如图 3 上所示;类似地,解密 S_0 得到 P_i 的值,是密钥 A_1 活跃的一条差分传播路径,如图 3 下所示.经验证可知,图 3 中上下两条差分路径没有相交的 S 盒部分,这样我们就成功建立了 5 轮的 Biclique 结构,即:任意的 $i,j\in\{0,1\}^d$,均有 $P_i\xrightarrow{K[i,j]}S_j$.此过程中选择明文的数量决定了整个攻击过程中的数据复杂度,通过图 3 下可知:明文状态初始为全 0,扩散结果只有 0,1,3,11 共 4 个子块活跃,则攻击过程中需要的明

文在其余 12 个字节位置均与 P_0 的值相同,均为 0;更进一步地,选择明密文个数至多为 2^{16}.

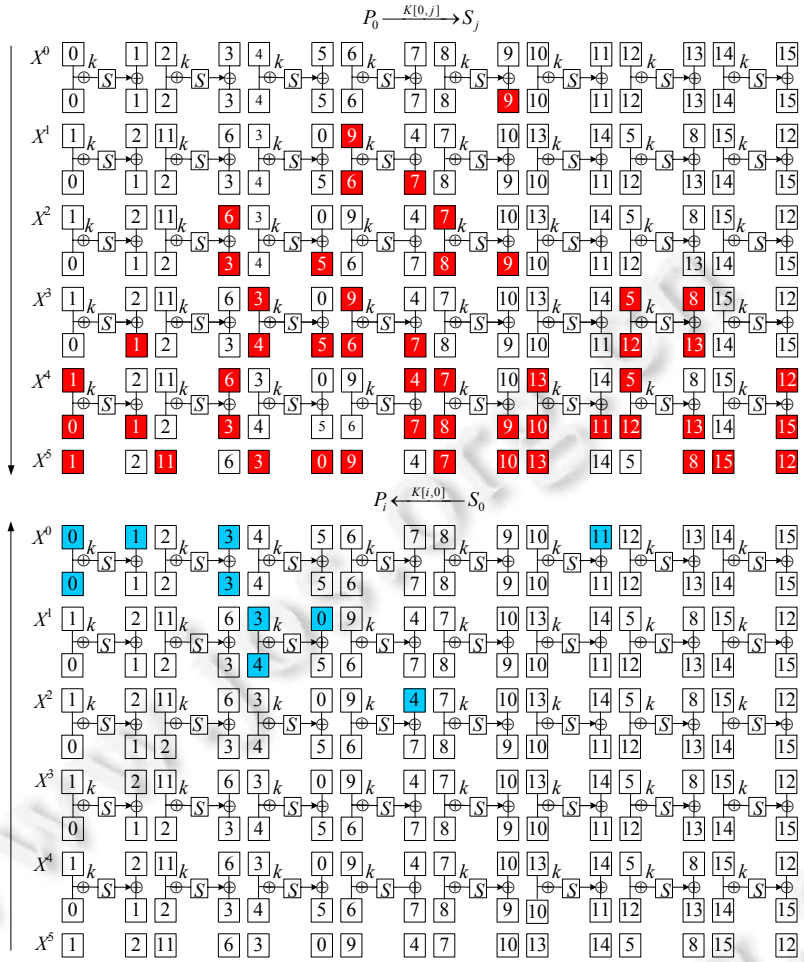

Fig.3 Construction of 5-round Biclique in TWINE-80

图 3 TWINE-80 中 5 轮 Biclique 的建立

3.1.3 13 轮的中间相遇过程

在 Biclique 建立成功之后,我们再进行部分匹配的过程.匹配位置选择在第 10 轮输入的第 12 个字节,从前后向分别计算部分匹配位置的值,通过是否存在匹配来删减候选密钥.

- 前向过程

具体计算过程如图 4 上所示.图中蓝色部分表示 i 未知的情形下导致的未知中间状态位置,由此可知,匹配位置 u(第 10 轮输入的第 12 个字节位置)的计算不受 i 部分密钥的影响.此外,为了降低攻击的时间复杂度,只计算图中黄色部分即可得到部分匹配位置的值,共需计算 12 个 S 盒.

- 后向过程

具体计算过程如图 4 下所示.图中红色部分表示 j 未知导致的未知中间状态位置,同样可知,匹配位置 v(第 10 轮输入的第 12 个字节位置)的计算不受 j 部分密钥的影响.类似地,只计算图中黄色部分即可得到 v 的值,共覆盖 27 个 S 盒.

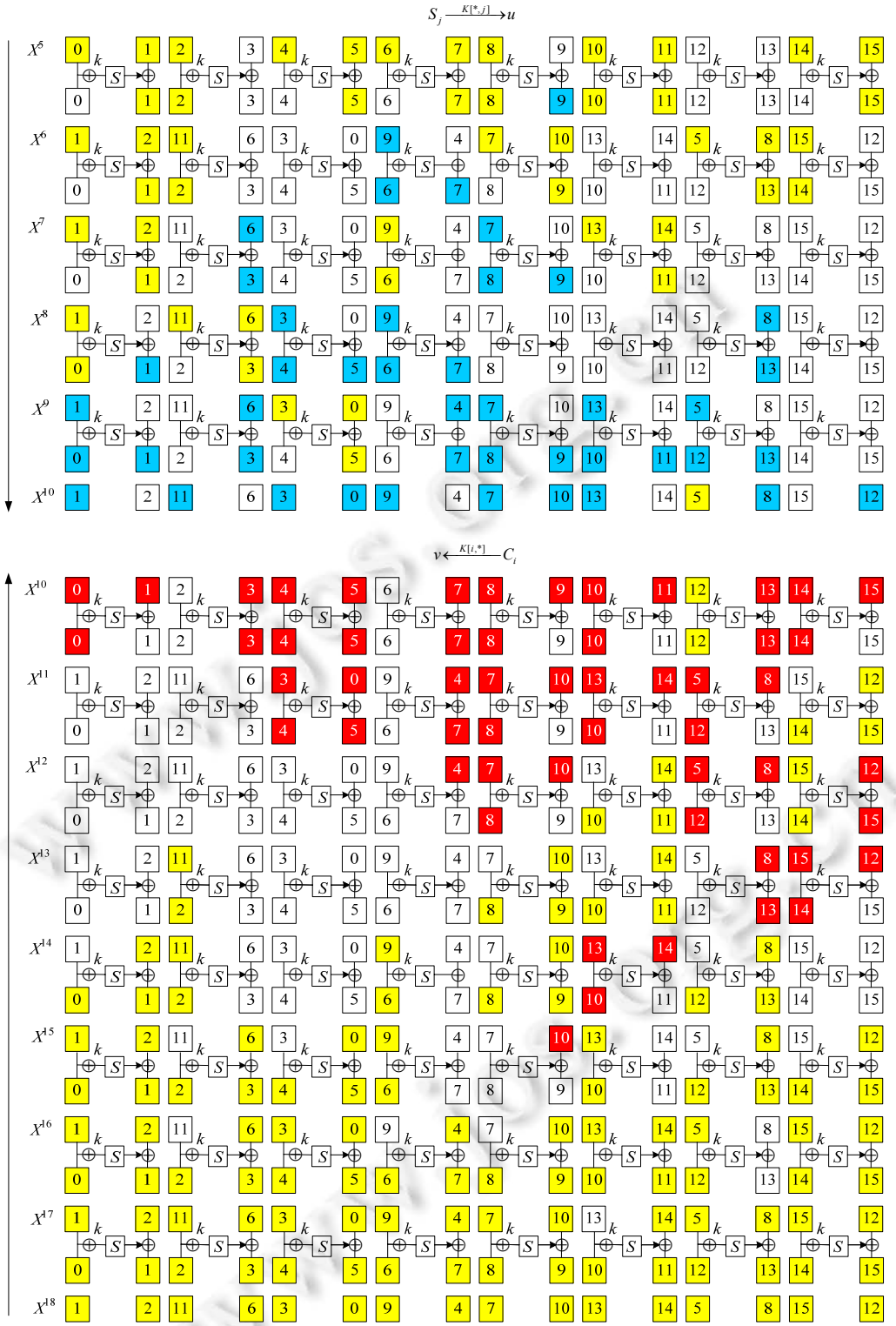

Fig.4 Partial matching computation of TWINE-80

图 4 TWINE-80 的部分匹配过程

3.1.4 复杂度分析

1. 数据:建立 Biclique 的过程决定了整个攻击的数据复杂度,需要 2^{16} 选择明密文对;
2. 时间:近年来,对于时间复杂度的计算越来越细致,我们将非线性组件作为最小计算单元,上述攻击的时间复杂度为 $2^{72}\left(\frac{2^4(12+4)+40+2^4(12+27)}{18\times 8}\right)+2^{76}\approx 2^{76}$ 的 18 轮 TWINE-80 加密过程;
3. 存储:只要存储单向过程的结果即可,可以用 Hash 值索引找到,因此存储 2^4 个 4bit 信息.

3.2 TWINE-128算法的22轮攻击过程

类似地,我们将提取第 8 轮子密钥时的 128bit 作为恢复目标,给出 22 轮的 TWINE-128 算法的中间相遇攻击结果.后向匹配过程的中立比特 A_2 的活跃位置为第 8 轮等价主密钥下的第 22 个密钥分块.相应地,前向过程中的中立比特 A_1 为第 24 个密钥分块.

3.2.1 中立比特对密钥编排算法的影响

根据 TWINE-128 算法的密钥编排算法,前向后向计算得知:从第 0 轮~第 21 轮(共 22 轮)的轮子密钥的活跃模式见表 2,其中,0 表示相应主密钥变化对其没有影响,1 表示会随对应密钥的变化而发生变化.

Table 2 Key schedule of TWINE-128 influenced by neutral bits

表 2 中立比特对 TWINE-128 密钥编排算法的影响

轮数	第 8 轮 K_{22} 活跃								第 8 轮 K_{24} 活跃							
	RK_0	RK_1	RK_2	RK_3	RK_4	RK_5	RK_6	RK_7	RK_0	RK_1	RK_2	RK_3	RK_4	RK_5	RK_6	RK_7
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
5	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
13	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
16	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
17	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
21	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

3.2.2 7 轮 Biclique 的建立

建立 Biclique 的过程与 TWINE-80 算法类似.通过验证图 5 中上下两条差分路径没有相交的 S 盒部分,则可成功建立 7 轮的 Biclique 结构.

通过图 5 下可知:明文状态只有 13,14,15 共 3 个子块位置活跃,亦即攻击需要的明文在其余 13 个字节与 P_0 的值相同,均为 0.因此,整个攻击过程中需要的选择明密文个数为 2^{12}.

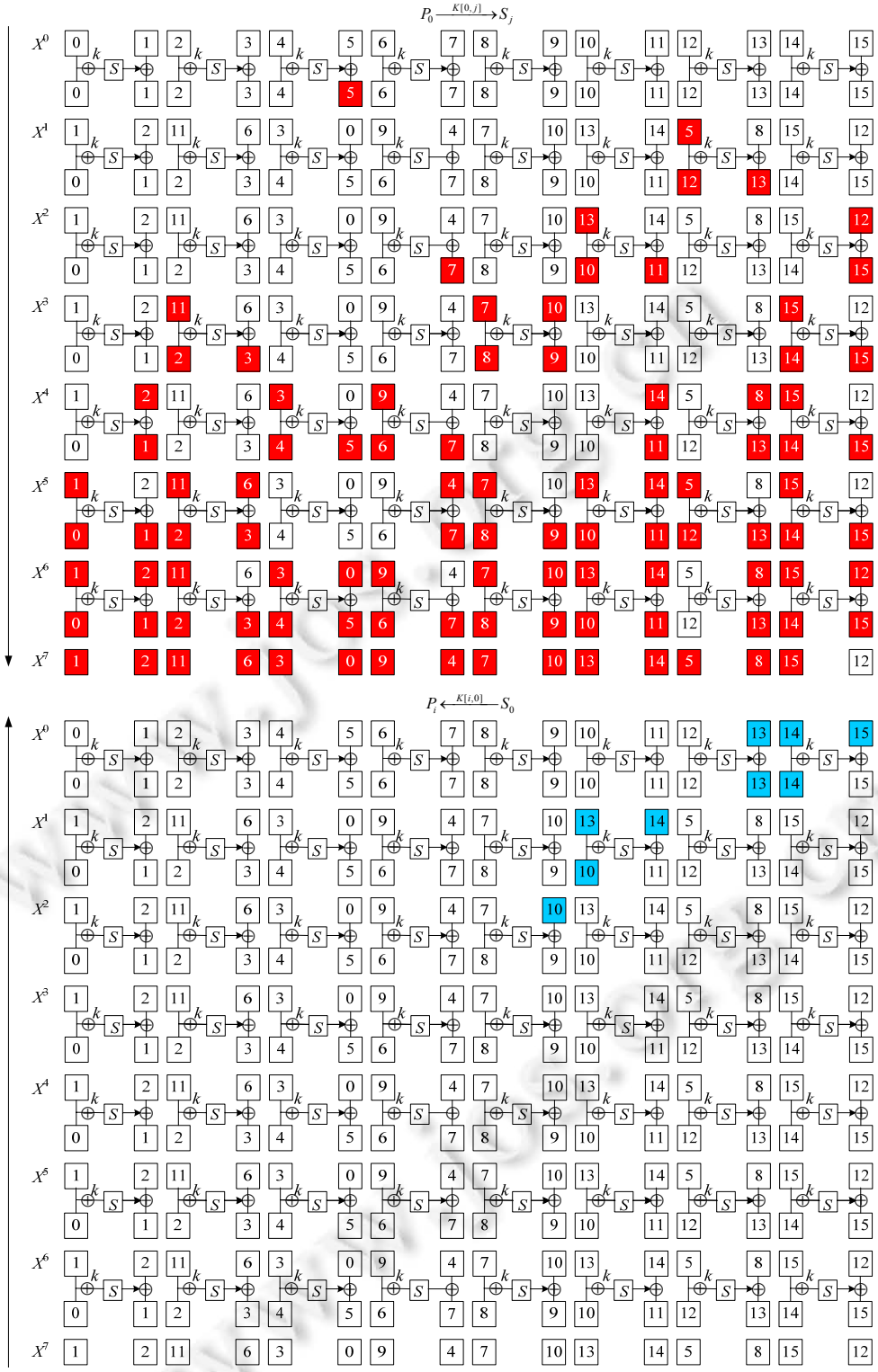

Fig.5 Construction of 7-round Biclique in TWINE-128

图 5 TWINE-128 中 7 轮 Biclique 的建立

3.2.3 15 轮的部分匹配过程

在 Biclique 建立成功之后,我们将部分匹配位置选择在第 13 轮输入的第 4 个字节位置.从前后向分别计算得到部分匹配位置的值,通过是否匹配来删减密钥.由于受篇幅所限,这里不再详述部分匹配的过程,只简单说明这一过程的计算复杂度,前向过程覆盖 19 个 S 盒,后向过程覆盖 35 个 S 盒.

3.2.4 复杂度分析

1. 数据:2^{12} 选择明密文对;
2. 时间:详细计算为 $2^{120}\left(\frac{2^4(23+3)+56+2^4(19+35)}{22\times 8}\right)+2^{124}\approx 2^{124}$ 的 22 轮 TWINE-128 加密过程;
3. 存储:2^4 个 4bit 信息.

4 与现存攻击结果的比较

现将之前存在的 TWINE 算法的攻击结果与本文结果相对比,总结为表 3.观察表 3 可知:本文中的攻击结果与之前 TWINE 算法的结果相比,攻击轮数没有优势,但数据复杂度有很大优势.虽然 TWINE 算法的 Biclique 分析均可以使得全轮攻击下的时间复杂度优于穷举攻击,但是 Biclique 分析可以给出任意算法全轮下的攻击结果,如 Present,LED,mCrypton 以及 LBlock 等.因此,Biclique 分析结果作为 TWINE 算法抗中间相遇攻击的结果有些牵强.本文可以看作是首次给出 TWINE 算法抗传统中间相遇攻击的结果,具体时间、数据以及存储复杂度可见表 3.

Table 3 Comparisons of cryptanalysis results on TWINE

表 3 对 TWINE 算法分析结果的比较

分组密码	攻击方法	轮数	数据	时间	存储	参考文献
TWINE-80	零相关线性	23	$2^{62.1}$	$2^{72.15}$	2^{60}	文献[22]
	饱和攻击	22	2^{62}	$2^{68.43}$	2^{67}	文献[18]
	不可能差分分析	23	$2^{61.39}$	$2^{76.88}$	2^{74}	文献[18]
	Biclique 分析	36	2^{60}	$2^{79.10}$	2^4	文献[19]
	中间相遇攻击	18	2^{16}	2^{76}	2^4	本文
TWINE-128	零相关线性	25	$2^{62.1}$	$2^{122.12}$	2^{60}	文献[22]
	饱和攻击	23	$2^{62.81}$	$2^{106.14}$	2^{103}	文献[18]
	Biclique 分析	36	2^{60}	$2^{126.82}$	2^4	文献[19]
	不可能差分分析	24	$2^{52.21}$	$2^{115.10}$	2^{118}	文献[18]
	中间相遇攻击	22	2^{12}	2^{124}	2^4	本文

5 结束语

本文提出了一种通用的中间相遇攻击模式,并给出了 18 轮 TWINE-80 以及 22 轮 TWINE-128 算法的攻击结果,攻击时间复杂度分别为 2^{76} 和 2^{124},数据复杂度分别为 2^{16} 和 2^{12},这是迄今为止针对 TWINE 算法攻击中数据复杂度最小的攻击结果.此外,因为 Biclique 分析可以给出任意算法全轮下优于穷举攻击的结果,所以此中间相遇攻击模式作为设计者评估算法抗中间相遇分析的结果更为合适.类似本文的攻击模式,在解密 Oracle 情形下也可以建立类似的攻击模式,对应的数据复杂度类型为选择密文.进一步要做的工作是评估其他算法抗此类攻击的结果以及此分析方法的自动化实现.

References:

[1] Diffie W, Hellman ME. Special feature exhaustive cryptanalysis of the NBS data encryption standard. Computer, 1977,10(6):74–84. [doi: 10.1109/c-m.1977.217750]

[2] Sasaki Y. Meet-in-the-Middle preimage attacks on AES Hashing modes and an application to whirlpool. In: Joux A, ed. Proc. of the Fast Software Encryption. Berlin, Heidelberg: Springer-Verlag, 2011. 378–396. [doi: 10.1007/978-3-642-21702-9_22]

[3] Sasaki Y, Aoki K. Finding preimages in full MD5 faster than exhaustive search. In: Joux A, ed. Proc. of the Advances in Cryptology—EUROCRYPT 2009. Berlin, Heidelberg: Springer-Verlag, 2009. 134−152. [doi: 10.1007/978-3-642-01001-9_8]

[4] Guo J, Ling S, Rechberger C, Wang H. Advanced meet-in-the-middle preimage attacks: First results on full tiger, and improved results on MD4 and SHA-2. In: Abe M, ed. Proc. of the Advances in Cryptology—ASIACRYPT 2010. Berlin, Heidelberg: Springer-Verlag, 2010. 56−75. [doi: 10.1007/978-3-642-17373-8_4]

[5] Knellwolf S, Khovratovich D. New preimage attacks against reduced SHA-1. In: Safavi-Naini R, Canetti R, eds. Proc. of the Advances in Cryptology—CRYPTO 2012. Berlin, Heidelberg: Springer-Verlag, 2012. 367−383. [doi: 10.1007/978-3-642-32009-5_22]

[6] Khovratovich D, Rechberger C, Savelieva A. Bicliques for preimages: Attacks on Skein-512 and the SHA-2 family. In: Canteaut A, ed. Proc. of the Fast Software Encryption. Berlin, Heidelberg: Springer-Verlag, 2012. 244−263. [doi: 10.1007/978-3-642-34047-5_15]

[7] Li J, Isobe T, Shibutani K. Converting meet-in-the-middle preimage attack into pseudo collision attack: Application to SHA-2. In: Canteaut A, ed. Proc. of the Fast Software Encryption. Berlin, Heidelberg: Springer-Verlag, 2012. 264−286. [doi: 10.1007/978-3-642-34047-5_16]

[8] Aoki K, Sasaki Y. Preimage attacks on one-block MD4, 63-step MD5 and more. In: Avanzi R, Keliher L, Sica F, eds. Proc. of the Selected Areas in Cryptography. Berlin, Heidelberg: Springer-Verlag, 2009. 103−119. [doi: 10.1007/978-3-642-04159-4_7]

[9] Wei YZ, Su CM, Ma CB. Meet-in-the-Middle attack on Rijndael-256 algorithm. Computer Engineering, 2012,38(7):107−109 (in Chinese with English abstract). [doi: 10.3969/j.issn.1000-3428.2012.07.035]

[10] Wang Z, Zhang WY. Meet-in-Middle attack on 5-round square. Computer Technology and Development, 2011,21(6):132−135, 139 (in Chinese with English abstract). [doi: 10.3969/j.issn.1673-629X.2011.06.036]

[11] Bogdanov A, Rechberger C. A 3-subset meet-in-the-middle attack: Cryptanalysis of the lightweight block cipher KTANTAN. In: Biryukov A, Gong G, Stinson D, eds. Proc. of the Selected Areas in Cryptography. Berlin, Heidelberg: Springer-Verlag, 2011. 229−240. [doi: 10.1007/978-3-642-19574-7_16]

[12] Bogdanov A, Khovratovich D, Rechberger C. Biclique cryptanalysis of the full AES. In: Lee D, Wang X, eds. Proc. of the Advances in Cryptology—ASIACRYPT 2011. Berlin, Heidelberg: Springer-Verlag, 2011. 344−371. [doi: 10.1007/978-3-642-25385-0_19]

[13] Canteaut A, Naya-Plasencia M, Vayssière B. Sieve-in-the-Middle: Improved MITM attacks. In: Canetti R, Garay J, eds. Proc. of the Advances in Cryptology—CRYPTO 2013. Berlin, Heidelberg: Springer-Verlag, 2013. 222−240. [doi: 10.1007/978-3-642-40041-4_13]

[14] Lee C. Biclique cryptanalysis of PRESENT-80 and PRESENT-128. The Journal of Supercomputing, 2014,70(1):95−103. [doi: 10.1007/s11227-014-1103-3]

[15] Jeong K, Kang H, Lee C, Sung J, Hong S, Lim J. Weakness of lightweight block ciphers mCrypton and LED against Biclique cryptanalysis. In: Proc. of the Peer-to-Peer Networking and Applications. 2013. 1−17. [doi: 10.1007/s12083-013-0208-4]

[16] Wang Y, Wu W, Yu X, Zhang L. Security on LBlock against Biclique cryptanalysis. In: Lee D, Yung M, eds. Proc. of the Information Security Applications. Berlin, Heidelberg: Springer-Verlag, 2012. 1−14. [doi: 10.1007/978-3-642-35416-8_1]

[17] Isobe T, Shibutani K. Security analysis of the lightweight block ciphers XTEA, LED and Piccolo. In: Susilo W, Mu Y, Seberry J, eds. Proc. of the Information Security and Privacy. Berlin, Heidelberg: Springer-Verlag, 2012. 71−86. [doi: 10.1007/978-3-642-31448-3_6]

[18] Suzaki T, Minematsu K, Morioka S, Kobayashi E. TWINE: A lightweight block cipher for multiple platforms. In: Knudsen L, Wu H, eds. Proc. of the Selected Areas in Cryptography. Berlin, Heidelberg: Springer-Verlag, 2013. 339−354. [doi: 10.1007/978-3-642-35999-6_22]

[19] Çoban M, Karakoç F, Boztaş Ö. Biclique cryptanalysis of TWINE. In: Pieprzyk J, Sadeghi AR, Manulis M, eds. Proc. of the Cryptology and Network Security. Berlin, Heidelberg: Springer-Verlag, 2012. 43−55. [doi: 10.1007/978-3-642-35404-5_5]

[20] Bouillaguet C, Derbez P, Fouque PA. Automatic search of attacks on round-reduced AES and applications. In: Rogaway P, ed. Proc. of the Advances in Cryptology—CRYPTO 2011. Berlin, Heidelberg: Springer-Verlag, 2011. 169−187. [doi: 10.1007/978-3-642-22792-9_10]

[21] Bouillaguet C, Derbez P, Dunkelman O, Fouque P, Keller N, Rijmen V. Low-Data complexity attacks on AES. IEEE Trans. on Information Theory, 2012,58(11):7002−7017. [doi: 10.1109/TIT.2012.2207880]

[22] Wang Y, Wu W. Improved multidimensional zero-correlation linear cryptanalysis and applications to LBlock and TWINE. In: Susilo W, Mu Y, eds. Proc. of the Information Security and Privacy. Springer-Verlag, 2014. 1−16. [doi: 10.1007/978-3-319-08344-5_1]

附中文参考文献:

[9] 韦永壮,苏崇茂,马春波.Rijndael-256 算法的中间相遇攻击.计算机工程,2012,38(7):107−109. [doi: 10.3969/j.issn.1000-3428.2012.07.035]

[10] 王哲,张文英.对 5 轮 Square 的中间相遇攻击.计算机技术与发展,2011,21(6):132−135,139. [doi: 10.3969/j.issn.1673-629X.2011.06.036]

汪艳凤(1989－),女,博士生,山东潍坊人,主要研究领域为分组密码分析与设计.

吴文玲(1966－),女,博士,研究员,博士生导师,主要研究领域为对称密码的设计理论与分析方法.