doi:10.11887/j.cn.201506024 http://journal.nudt.edu.cn

低轮 PUFFIN 算法的积分攻击*

赵光耀[1],成 磊[2],李瑞林[3],李 超[1,2],孙 兵[2]
(1.国防科技大学 计算机学院,湖南 长沙 410073;2.国防科技大学 理学院,湖南 长沙 410073;
3.国防科技大学 电子科学与工程学院,湖南 长沙 410073)

摘 要:PUFFIN是一个分组长度为64bit的轻量级分组密码算法,其密钥长度为128bit。对PUFFIN抵抗积分攻击的能力进行研究,构造并证明PUFFIN算法存在5轮和6轮积分区分器。利用6轮积分区分器对8轮PUFFIN进行积分攻击,可恢复2轮共100bit轮密钥,攻击的数据复杂度为2^{20}个选择明文,时间复杂度约为2^{33}次8轮加密,存储复杂度为2^{20},这是目前为止对PUFFIN最好的积分分析结果。

关键词:PUFFIN;轻量级分组密码;积分攻击
中图分类号:TN918 **文献标志码**:A **文章编号**:1001-2486(2015)06-129-06

Integral cryptanalysis on reduced-round PUFFIN

ZHAO Guangyao[1], *CHENG Lei*[2], *LI Ruilin*[3], *LI Chao*[1,2], *SUN Bing*[2]
(1. College of Computer, National University of Defense Technology, Changsha 410073, China;
2. College of Science, National University of Defense Technology, Changsha 410073, China;
3. College of Electronic Science and Engineering, National University of Defense Technology, Changsha 410073, China)

Abstract: PUFFIN is a lightweight block cipher, in which the block length is 64 bit while the key size is 128 bit. The integral cryptanalysis resistance ability of PUFFIN was analyzed. The existence of 5 and 6 round integral distinguisher in PUFFIN was constructed and proved. An integral attack on 8 round PUFFIN was mounted by 6 round integral distinguisher to recover 2 round 100 bit round cipher. The data complexity of the attack is 2^{20} chosen plaintexts, the time complexity is about 2^{33} 8 round encryptions, and the space complexity is 2^{20}. This has been the best integral attack on PUFFIN up to now.

Key words: PUFFIN; lightweight block cipher; integral attack

随着物联网等应用的兴起,适用于资源受限环境的轻量级密码算法得到了飞速发展,密码学者根据不同的应用需求设计了许多的轻量级算法,例如HIGHT[1], LBlock[2], LED[3], PRESENT[4]等。PUFFIN[5]也是一种轻量级分组密码算法,采用混淆扩散网络结构(Substitution Permutation Networks,SPN)型结构,其分组长度为64bit,密钥长度为128bit。非线性层由16个相同的4×4的S盒并置组成,线性层则为64bit置换,其S盒及P置换均为对合结构,使得其加解密结构一致,硬件实现时占用芯片面积非常小。文献[5]中,设计者对PUFFIN抵抗差分分析[6]、线性分析[7]、相关密钥攻击[8]的能力进行了分析,并分析了算法的弱密钥[9],目前对PUFFIN的第三方安全性分析结果主要有线性分析[10]和积分攻击[11]。文献[10]主要分析了PUFFIN的线性特征,并对其进行了线性攻击;文献[11]则首次对PUFFIN抵抗积分攻击的安全性进行了研究。

积分攻击[12]的原理由Knudsen和Wagner提出。自提出以后,积分攻击在许多基于字节设计的算法上得到了很好的应用,例如Camellia[13], CLEFIA[14]等。Z′aba等针对Noekeon, Serpent, Present等基于比特设计的算法对积分攻击进行了扩展,提出了基于比特的积分攻击[15]。

1 基础理论

1.1 PUFFIN简介

PUFFIN为SPN型结构分组密码算法,其分组长度和密钥长度分别为64bit和128bit,迭代32

* **收稿日期**:2015-01-12
基金项目:国家自然科学基金资助项目(61402515);信息保障技术国家重点实验室开放基金资助项目(KJ-14-003)
作者简介:赵光耀(1982—),男,湖南湘潭人,博士研究生,E-mail:securityzgy@163.com;
孙兵(通信作者),男,讲师,博士,E-mail:happy_come@163.com

轮。64bit 明文(中间状态,轮密钥及密文)排列成一个 4 行 16 列的二维数组形式,即 $p_0, p_1, \cdots, p_{63}$ 可表示成 $\boldsymbol{V}_0, \boldsymbol{V}_1, \cdots, \boldsymbol{V}_{15}$ 共 16 个向量,其中 $\boldsymbol{V}_i = (p_{4i}, p_{4i+1}, p_{4i+2}, p_{4i+3})^{\mathrm{T}}, 0 \leqslant i \leqslant 15$,如图 1 所示。

$\boldsymbol{V}_0$	$\boldsymbol{V}_1$	$\boldsymbol{V}_2$	$\boldsymbol{V}_3$	$\boldsymbol{V}_4$	$\boldsymbol{V}_5$	$\boldsymbol{V}_6$	$\boldsymbol{V}_7$	$\boldsymbol{V}_8$	$\boldsymbol{V}_9$	$\boldsymbol{V}_{10}$	$\boldsymbol{V}_{11}$	$\boldsymbol{V}_{12}$	$\boldsymbol{V}_{13}$	$\boldsymbol{V}_{14}$	$\boldsymbol{V}_{15}$
p_0	p_4	p_8	p_{12}	p_{16}	p_{20}	p_{24}	p_{28}	p_{32}	p_{36}	p_{40}	p_{44}	p_{48}	p_{52}	p_{56}	p_{60}
p_1	p_5	p_9	p_{13}	p_{17}	p_{21}	p_{25}	p_{29}	p_{33}	p_{37}	p_{41}	p_{45}	p_{49}	p_{53}	p_{57}	p_{61}
p_2	p_6	p_{10}	p_{14}	p_{18}	p_{22}	p_{26}	p_{30}	p_{34}	p_{38}	p_{42}	p_{46}	p_{50}	p_{54}	p_{58}	p_{62}
p_3	p_7	p_{11}	p_{15}	p_{19}	p_{23}	p_{27}	p_{31}	p_{35}	p_{39}	p_{43}	p_{47}	p_{51}	p_{55}	p_{59}	p_{63}

图 1 PUFFIN 分组比特顺序

Fig. 1 Block state of PUFFIN

轮函数包含以下 3 个变换:

1)非线性层 γ 由 16 个相同的 4 × 4 的 S 盒并置组成,每列(V_i)通过 1 个 S 盒。S 盒映射见表 1。

表 1 S 盒映射(16 进制表示)

Tab. 1 S box map (in hexadecimal)

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	D	7	3	2	9	A	C	1	F	4	5	E	6	0	B	8

性质(S 盒表达式) 设 $x = (x_3, x_2, x_1, x_0)$ 和 $y = (y_3, y_2, y_1, y_0)$ 分别表示 S 盒的输入和输出(如图 2 所示),则 x 和 y 满足:

$$y_0 = x_0x_1x_2 + x_0x_1x_3 + x_0x_1 + x_0x_2 + x_0x_3 + x_1x_2 + x_2x_3 + 1,$$

$$y_1 = x_0x_1x_3 + x_0x_1 + x_0 + x_1x_2 + x_1 + x_3,$$

$$y_2 = x_0x_1x_2 + x_0x_2x_3 + x_1x_2x_3 + x_1x_3 + x_1 + x_2x_3 + x_2 + 1,$$

$$y_3 = x_0x_1x_3 + x_0x_1 + x_0x_2 + x_0 + x_1x_2x_3 + x_1x_2 + x_1 + x_2x_3 + 1。$$

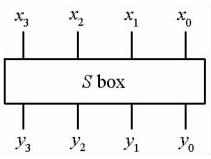

图 2 S 盒的输入输出

Fig. 2 Input and output of S box

2)密钥加 σ:64bit 的轮密钥与 64bit 的状态进行异或。

3)线性层进行 $P64$:64bit 的一个置换,其映射见表 2。变换输入为(行标 × 8 + 列标 + 1),如表 2 中(0,0)位置对应的元素为 13,即 $P64(0 \times 8 + 0 + 1) = P64(1) = 13$。

加密之初,首先进行密钥白化以及一个 $P64$ 转换,然后进行 32 轮轮函数的迭代,所以 PUFFIN 加密过程可表示为

$$\prod_{r=1}^{32}(P64 \circ \sigma_{k_r} \circ \gamma) \circ P64 \circ \sigma_{k_0}。$$

值得注意的是,PUFFIN 算法中的组件 S 盒及 $P64$ 均为对合的,即 $S(S(x)) = x, P64(P64(y)) = y$,其中 $x \in F_2^4, y \in F_2^{64}$。

表 2 $P64$ 映射

Tab. 2 $P64$ map

	0	1	2	3	4	5	6	7
0	13	2	60	50	51	27	10	36
1	25	7	32	61	1	49	47	19
2	34	53	16	22	57	20	48	41
3	9	52	6	31	62	30	28	11
4	37	17	58	8	33	44	46	59
5	24	55	63	38	56	39	15	23
6	14	4	5	26	18	54	42	45
7	21	35	40	3	12	29	43	64

1.2 基于比特的积分攻击

Z′aba 等提出的基于比特的积分,实际上是基于计数方法来进行的,通过统计每个比特上不同元素出现的奇偶次数来判断其平衡性。实际上,通过布尔函数的最高次项系数取值也可以判定其平衡性。

定理 1[16] 设多项式 $f(x) = \sum_{i=0}^{q-1} a_i x^i \in F_q[x]$,其中 q 是某个素数的方幂,则

$$\sum_{x \in F_q} f(x) = -a_{q-1}。$$

定理 2[16] 若多项式 $f(x) = \sum_{i=0}^{q-1} a_i x^i \in F_q[x]$ 是置换多项式,则 $a_{q-1} = 0$。

定理 1 说明,要确定密文某个位置是否平衡,可通过研究该位置密文与明文之间多项式函数的最高项系数来判断。定理 2 则给出了一个判断最高项系数的方法。

设算法分组长度为 n,其中的 m 个比特遍历 $\{0,1\}^m$,记为 $x_0, x_1, \cdots, x_{m-1}$;其余比特为常数,记作 $c_0, c_1, \cdots, c_{n-m-1}$。密文每个比特的值都是关于 $x_0, x_1, \cdots, x_{m-1}, c_0, c_1, \cdots, c_{n-m-1}$ 的函数,不妨记作 $f(x_0, x_1, \cdots, x_{m-1}, c_0, c_1, \cdots, c_{n-m-1})$,易知 f 是从 F_2^m 到 F_2 的一个映射。

令 $deg(f)$ 表示 f 的最高次数,若密文某个比特位置的表达式 f 对任意的 $c_0, c_1, \cdots, c_{n-m-1}$ 都

满足

$$deg(f) \leqslant m-1,$$

则对此位置上出现的所有 2^m 个元素(g_0, g_1,…, g_{2^m-1})求和有

$$\sum g_i = 0, 0 \leqslant i \leqslant 2^m - 1,$$

即对应的多项式函数最高次项系数为 0,此时该比特是平衡的。

2　PUFFIN 的积分攻击

2.1　PUFFIN 的 5 轮积分区分器

定理 3　设明文为 $P=(p_0, p_1, \cdots, p_{63})$,则当($p_6$, p_{24}, p_{31}, p_{60})遍历$\{0,1\}^4$ 时, 5 轮加密后密文有 29 个比特是平衡的。

证明:当输入明文的活跃位置为 p_6, p_{24}, p_{31}, p_{60}时,状态可表示为:

0	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60
1	5	9	13	17	21	25	29	33	37	41	45	49	53	57	61
2	6	10	14	18	22	26	30	34	38	42	46	50	54	58	62
3	7	11	15	19	23	27	31	35	39	43	47	51	55	59	63

方格内数字表示比特顺序,每一轮开始时重新编号。

经过 *P*64 后,活跃位置将位于同一列,即为:

12	50	24	0	33	56	8	61	36	32	23	55	13	17	20	11
1	26	6	48	52	19	51	29	16	43	54	38	3	53	34	28
59	9	31	46	15	47	5	27	57	45	62	14	4	41	39	42
49	35	60	18	21	40	30	10	7	58	37	22	25	44	2	63

再经过第 1 轮的 *S* 盒后,状态为:

0	4	y_0	12	16	20	24	28	32	36	40	44	48	52	56	60
1	5	y_1	13	17	21	25	29	33	37	41	45	49	53	57	61
2	6	y_2	14	18	22	26	30	34	38	42	46	50	54	58	62
3	7	y_3	15	19	23	27	31	35	39	43	47	51	55	59	63

记这 4 个位置的变量分别为 y_0, y_1, y_2, y_3,则经过第 1 轮的 *P*64 后,状态变为:

12	50	24	0	33	56	y_0	61	36	32	23	55	13	17	20	y_3
1	26	6	48	52	19	51	29	16	43	54	38	3	53	34	28
59	y_1	31	46	15	47	5	27	57	45	62	14	4	41	39	42
49	35	60	18	21	40	30	y_2	7	58	37	22	25	44	2	63

灰色底纹标记的位置标注有当前比特表达式的最高次项,无底纹的位置为常数,即其取值不受 y_i影响。

经过第 2 轮的 *S* 盒后,状态变为:

0	y_1	8	12	16	20	y_0	y_2	32	36	40	44	48	52	56	y_3
1	y_1	9	13	17	21	y_0	y_2	33	37	41	45	49	53	57	y_3
2	y_1	10	14	18	22	y_0	y_2	34	38	42	46	50	54	58	y_3
3	y_1	11	15	19	23	y_0	y_2	35	39	43	47	51	55	59	y_3

再经过 *P*64 状态为:

12	50	y_0	0	33	56	8	61	36	32	23	55	13	17	20	y_3
1	y_0	y_1	48	52	19	51	29	16	43	54	38	3	53	34	28
59	9	y_2	46	15	47	y_1	27	57	45	62	14	4	41	39	42
49	35	y_3	18	21	40	y_2	y_2	7	58	37	22	25	44	2	63

经过第 3 轮的 *S* 盒后,依据性质 1,状态变为:

0	y_0	y_{012} y_{013}	12	16	20	y_{12}	y_{023}	y_1	36	y_3	44	y_{01}	52	56	y_{23}
1	y_0	y_{013}	13	17	21	y_1 y_2	y_{23} y_{02}	y_1	37	y_3	45	y_0 y_1	53	57	y_{23}
2	y_0	y_{012} y_{023} y_{123}	14	18	22	y_{12}	y_{023}	y_1	38	y_3	46	y_{01}	54	58	y_{23}
3	y_0	y_{013} y_{123}	15	19	23	y_{12}	y_{23} y_{03} y_{02}	y_1	39	y_3	47	y_{01}	55	59	y_{23}

其中 y_{ijk}表示此比特表达式的最高次项为 y_iy_j y_k,$0 \leqslant i, j, k \leqslant 3$。第 3 轮的输出状态为:

12	y_{01}	y_{12}	0	y_1	56	y_{012} y_{013}	y_{23}	36	y_1	23	55	13	17	20	y_{013} y_{123}
1	y_{12}	y_0	y_{01}	52	19	y_{01}	y_{23} y_{02}	16	y_3	54	38	3	53	y_1	y_{023}
59	y_{013}	y_{23} y_{03} y_{02}	46	15	47	y_0	y_{12}	57	45	y_{23}	14	y_0	y_3	39	y_3
y_0 y_1	y_1	y_{23}	18	21	y_3	y_{023}	y_{12} y_{023} y_{123}	y_0	58	37	22	y_1 y_2	44	2	y_{23}

依此类推,可得第 4 轮的输出状态为:

y_{01}	y_{01} y_{02}	y_{0123}	y_0 y_1	y_0	y_1	y_{0123}	y_{0123}	y_{13}	y_0	y_3	y_3	y_{01}	y_1	y_3	y_{0123}
y_0 y_1	y_{0123}	y_{0123}	y_{01} y_{02}	y_3	y_1	y_{01} y_{02}	y_{0123}	y_1	y_{23}	y_3	y_{13}	y_0 y_1	y_3	y_0	y_{0123}
y_1	y_{0123}	y_{0123}	46	y_{01}	47	y_{0123}	y_{0123}	y_1	45	y_{0123}	y_{01}	y_{0123}	y_{23}	y_{13}	y_{23}
y_0 y_1 y_2	y_0	y_{0123}	y_1	y_3	y_{23}	y_{0123}	y_{0123}	y_{0123}	y_1	y_{13}	y_3	y_{0123}	44	y_0 y_1	y_{0123}

第 5 轮的输出状态为:

y_{012}	y_{0123}	y_{0123}	y_{012}	y_{0123}	y_{013}	y_{0123}	y_{0123}	y_{0123}	y_{0123}	y_{123}	y_{123}	y_{012}	y_{013}	y_{123}	y_{0123}
y_{012}	y_{0123}	y_{0123}	y_{0123}	y_{123}	y_{013}	y_{0123}	y_{0123}	y_{013}	y_{0123}	y_{123}	y_{023} y_{01} y_{123}	y_{012}	y_{13} y_{23}	y_{0123}	y_{0123}
y_{013}	y_{0123}	y_{0123}	y_{013}	y_{012}	y_{013}	y_{0123}	y_{0123}	y_{013}	y_{013}	y_{0123}	y_{012}	y_{0123}	y_{0123}	y_{0123}	y_{0123}
y_{0123}	y_{0123}	y_{0123}	y_{013}	y_{123}	y_{0123}	y_{0123}	y_{0123}	y_{0123}	y_{013}	y_{0123}	y_{123}	y_{0123}	y_{013}	y_{012}	y_{0123}

综上讨论,当(p_6, p_{24}, p_{31}, p_{60})遍历$\{0,1\}^4$ 时,(y_0, y_1, y_2, y_3)也取遍 2^4 个值,则 5 轮加密后的密文中灰色底纹标记的比特位置关于 y_0, y_1, y_2, y_3 的布尔函数 $f(y_0, y_1, y_2, y_3)$,满足 $deg(f) \leqslant 3$,因此这些比特位置平衡。 □

2.2　PUFFIN 的 6 轮积分区分器

在 5 轮积分区分器前加 1 轮,可将其扩展至 6 轮的积分区分器。

定理 4　当明文的 16 个比特$\{p_5, p_8, p_9, p_{10}, p_{11}, p_{26}, p_{27}, p_{28}, p_{29}, p_{30}, p_{35}, p_{42}, p_{50}, p_{51},$

p_{61}, p_{63}}遍历{0,1}^{16}时,6 轮 PUFFIN 算法加密后密文的平衡比特与 2.1 节所述的 5 轮积分区分器输出平衡位置相同。

证明:如图 3 所示,当{p_5, p_8, p_9, p_{10}, p_{11}, p_{26}, p_{27}, p_{28}, p_{29}, p_{30}, p_{35}, p_{42}, p_{50}, p_{51}, p_{61}, p_{63}}遍历{0,1}^{16}时,经过白化和 $P64$ 变换后,输出状态的 V_1, V_6, V_7, V_{15}的级联遍历{0,1}^{16},经过第 1 轮非线性层后,第 6、第 24、第 31 和第 60 比特遍历{0,1}^{4},从而满足 5 轮积分区分器的输入状态,故原 5 轮积分区分器的平衡位置在该 6 轮积分区分器中仍然平衡。 □

上述证明说明 5 轮积分区分器的 29 个平衡位置在扩展的 6 轮高阶积分区分器中仍然平衡,通过实验测试验证,扩展的 6 轮高阶积分区分器有且仅有这 29 个平衡位置。

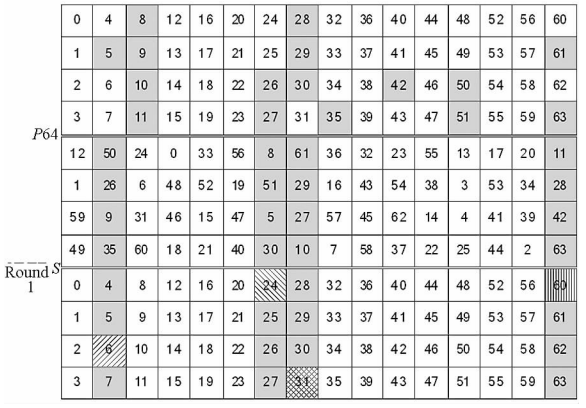

图 3 将 5 轮积分区分器扩展至 6 轮积分区分器

Fig. 3 Extend the 5-round integral distinguisher to the 6-round one

2.3 对 8 轮 PUFFIN 的积分攻击

利用 6 轮的高阶积分区分器,可以对 8 轮的 PUFFIN 进行积分攻击,从而可获取部分轮密钥信息。如图 4 所示。

攻击的主要思想是通过猜测第 8 轮的轮密钥 $RK^{(8)}$ 及第 7 轮的轮密钥 $RK^{(7)}$ 的部分比特,对密文进行部分解密后,观测第 6 轮输出的对应位置是否平衡来筛选密钥。当选择不同的平衡位置进行密钥筛选时,能够筛选的 $RK^{(8)}$ 和 $RK^{(7)}$ 的密钥字(4bit)是不同的,平衡位置与可筛选的密钥字间的对应关系见表 3。

以 44, 45, 46, 47 这 4 个平衡位置为例,其攻击步骤为:

1)选择一组明文(其中{p_5, p_8, p_9, p_{10}, p_{11}, p_{26}, p_{27}, p_{28}, p_{29}, p_{30}, p_{35}, p_{42}, p_{50}, p_{51}, p_{61}, p_{63}}取遍{0,1}^{16},其余位置为常数,故一组明文包含 2^{16} 个明文)进行 8 轮加密,密文记为 $C_0, C_1, \cdots, C_{2^{16}-1}$。

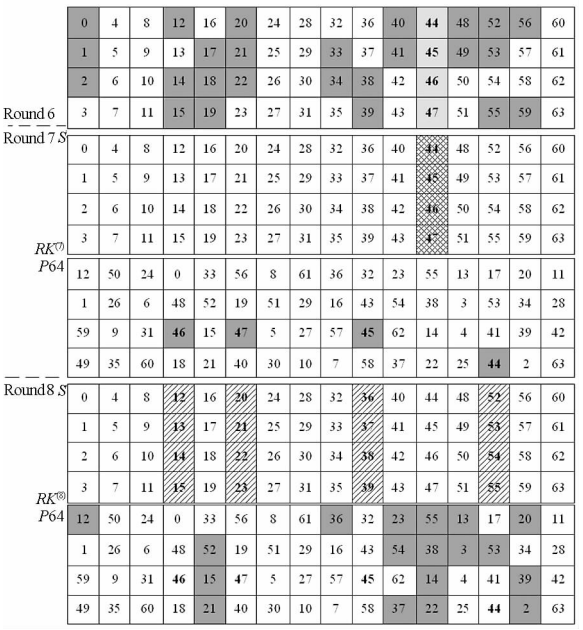

图 4 8 轮 PUFFIN 算法的积分攻击

Fig. 4 Integral attack on 8-round PUFFIN

表 3 选择平衡位置与可筛选密钥字的对应关系

Tab. 3 Relationship between balanced positions and nibbles of the recovered round keys

序号	区分器平衡位置	$RK^{(7)}$(密钥字)	$RK^{(8)}$(密钥字)	使用 16 组明文时剩余错误密钥期望值 $\bar{N}$
1	**44**,**45**,**46**,**47**	**11**	3,5,9,13	$(2^{20}-1)\times(2^{-4})^{16}\approx 2^{-44}$
2	0,1,2	0	0,**3**,12,14	$(2^{16}-1)\times(2^{-3})^{16}\approx 2^{-32}$
3	12,14,15	3	**0**,4,11,**12**	$(2^{12}-1)\times(2^{-3})^{16}\approx 2^{-36}$
4	17,18,19	4	**3**,**5**,8,**13**	$(2^{8}-1)\times(2^{-3})^{16}\approx 2^{-40}$
5	20,21,22	5	**4**,10,**11**,**14**	$(2^{8}-1)\times(2^{-3})^{16}\approx 2^{-40}$
6	52,53,55	13	**4**,**10**,**11**,**13**	$(2^{4}-1)\times(2^{-3})^{16}\approx 2^{-44}$
7	33,34	8	1,**4**,**9**,**14**	$(2^{8}-1)\times(2^{-2})^{16}\approx 2^{-24}$
8	38,39	9	**8**,**10**,**11**,**14**	$(2^{4}-1)\times(2^{-2})^{16}\approx 2^{-28}$
9	40,41	10	**5**,**9**,**13**,15	$(2^{8}-1)\times(2^{-2})^{16}\approx 2^{-24}$
10	48,49	12	**0**,**1**,**3**,6	$(2^{8}-1)\times(2^{-2})^{16}\approx 2^{-24}$
11	**56**,**59**	**14**	**0**,**5**,**8**,**9**	$(2^{4}-1)\times(2^{-2})^{16}\approx 2^{-28}$

2)猜测 $RK^{(8)}$ 的 4 个密钥字(共 16bit)

$RK_3^{(8)}$, $RK_5^{(8)}$, $RK_9^{(8)}$, $RK_{13}^{(8)}$,计算 $Q_j^{(i)} = \gamma^{-1}(P64^{-1}(C_i) \oplus RK_j^{(8)})$, $j \in \{3,5,9,13\}$。

3)计算 $T^i = P64^{-1}(Q^{(i)})$,猜测 $RK^{(7)}$ 的一个密钥字 $RK_{11}^{(7)}$,计算 $t = S^{-1}(T_{11}{}^i \oplus RK_{11}^{(7)})$。

4)判断 t 是否为 0,若 $t=0$,则猜测的 $RK_3^{(8)}$, $RK_5^{(8)}$, $RK_9^{(8)}$, $RK_{13}^{(8)}$, $RK_{11}^{(7)}$ 正确,否则,淘汰。

5)重新选取一组明文,重复上述步骤,直到唯一确定 $RK_3^{(8)}$,$RK_5^{(8)}$,$RK_9^{(8)}$,$RK_{13}^{(8)}$ 和 $RK_{11}^{(7)}$。

复杂度分析:实际攻击时可以根据表 3 从上到下(利用的平衡位置由多到少)对密钥字进行筛选,即攻击共需进行 11 次筛选。表 3 中 $RK^{(8)}$ 对应列粗体标注的表示此密钥字已经唯一确定;第 4 列给出了当使用 16 组明文进行攻击时,各次筛选中错误密钥剩余个数的期望值 $\overline{N}$,$\overline{N} \leqslant 2^{-24} < 1$,可认为当使用 16 组明文进行分析时,可将表 4 中涉及 $RK^{(7)}$ 的 11 个密钥字和 $RK^{(8)}$ 的 14 个密钥字共 100bit 信息唯一确定。因此攻击的数据复杂度为 $2^{16} \times 16 \approx 2^{20}$。11 次筛选过程需猜测密钥字个数不同(其中需猜测 5 个、4 个和 3 个密钥字的各 1 次,有 5 次筛选需猜测 2 个密钥字,3 次需猜测 1 个密钥字),所以攻击的时间复杂度为 $2^{20} \times (2^{4\times5} + 2^{4\times4} + 2^{4\times3} + 5 \times 2^{4\times2} + 3 \times 2^{4\times1}) \approx 2^{40}$ 次查表,这相当于 $2^{40}/(8 \times 16) = 2^{33}$ 次 8 轮加密。另外,为存储密钥,攻击需要对猜测的密钥字进行存储,存储复杂度为 $(2^{4\times5} + 2^{4\times4} + 2^{4\times3} + 5 \times 2^{4\times2} + 3 \times 2^{4\times1}) \approx 2^{20}$。

实验及结果:在 PC 机上利用 C++(Visual C++6.0)编程模拟了密钥筛选过程。实验中每组明文的常数值随机生成,首先考虑当 11 次筛选均对 20bit 轮密钥进行筛选时(重复筛),共做了 500 次模拟实验,统计唯一确定 20bit 密钥平均所需的明文组数。实验结果如图 4 所示。

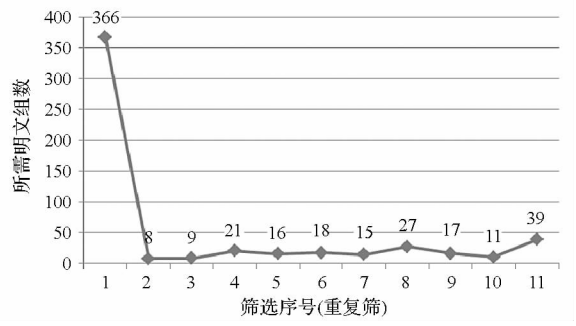

图 4　唯一确定密钥所需明文组数(重复筛选时)

Fig. 4　Group number of plaintexts to find the right key (when each filtration works on 20bit keys)

若在筛选过程中,已经确定的密钥字不再重新筛选(不重复筛),统计唯一确定猜测密钥字所需的明文组数。图 5 所示为 500 次模拟实验的平均结果。

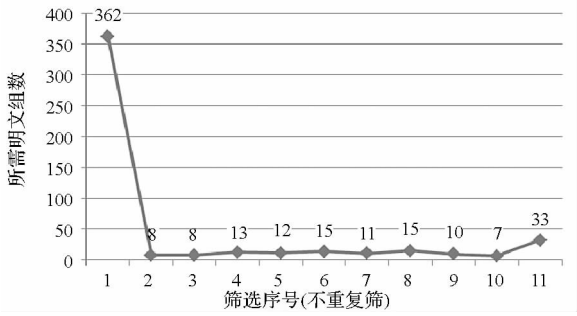

图 5　唯一确定密钥所需明文组数(不重复筛选时)

Fig. 5　Group number of plaintexts to find the right key (when filtrating step by step)

实验结果显示,大部分筛选使用约 16 组明文即可唯一确定正确密钥(攻击时可按照所需明文组数由少到多的顺序进行筛选,保证前面的筛选能够将密钥字唯一确定)。当使用平衡位置{56,59}以及{44,45,46,47}进行筛选时,需要的明文组较多,这主要是由于这些平衡位置比较特殊,对于猜测的 $RK^{(7)}$ 密钥字的所有可能取值,这些位置以较大概率保持平衡,所以在确定正确密钥过程中需要的明文组数很多。

3　结论

利用基于比特的积分思想对 PUFFIN 算法进行了积分分析,构造并证明了算法存在 5 轮和 6 轮积分区分器,对 8 轮 PUFFIN 算法进行了积分攻击。构造的积分区分器输出的平衡位置较多,因此对 8 轮 PUFFIN 算法进行积分攻击时效率较高,恢复 100bit 轮密钥所需的数据复杂度为 2^{20} 个选择明文,时间复杂度为 2^{33} 次 8 轮加密,存储复杂度为 2^{20}。

参考文献(References)

[1] Hong D, Sung J, Hong S, et al. HIGHT: a new block cipher suitable for low-resource device [C]//Proceedings of Cryptographic Hardware and Embedded Systems, 2006, 4249: 46 - 59.

[2] Wu W L, Zhang L. LBlock: a lightweight block cipher[C]// Proceedings of Applied Cryptography and Network Security, 2011, 6715: 327 - 344.

[3] Guo J, Peyrin T, Poschmann A, et al. The LED block cipher[C]// Proceedings of Cryptographic Hardware and Embedded Systems, 2011, 6917: 326 - 341.

[4] Bogdanov A, Knudsen L, Leander G, et al. PRESENT: an ultra-lightweight block cipher [C]//Proceedings of Cryptographic Hardware and Embedded Systems, 2007, 4727: 450 - 466.

[5] Cheng H, Heys H, Wang C. PUFFIN: a novel compact block

cipher targeted to embedded digital systems [C] // Proceedings of 11th EUROMICRO Conference on Digital System Design: Architectures, Methods and Tools, 2008: 383 – 390.

[6] Biham E, Shamir A. Differential cryptanalysis of DES-like cryptosystems[C] //Proceedings of Advances in Cryptology: CRYPTO'90, 1990, 537: 2 – 21.

[7] Matsui M. Linear cryptanalysis method for DES cipher[C]// Proceedings of Advances in Cryptology: EUROCRYPT '93, 1993, 765: 386 – 397.

[8] Biham E. New type of cryptanalytic attacks using related keys[J]. Journal of Cryptology, 1994, 7(4): 229 – 246.

[9] Moore J H, Simmons G J. Cycle structure of the DES for keys having palindromic (or antipalindromic) sequences of round keys[J]. IEEE Transactions on Software Engineering, 1987, 13(2): 262 – 273.

[10] Leander G. On linear hulls, statistical saturation attacks, PRESENT and a cryptanalysis of PUFFIN[C] //Proceedings of Advances in Cryptology-EUROCRYPT, 2011, 6632: 303 – 322.

[11] 魏悦川,孙兵,李超. 一种 PUFFIN 类 SPN 型分组密码的积分攻击[J]. 国防科技大学学报, 2010, 32(3): 139 – 143.
WEI Yuechuan, SUN Bing, LI Chao. An integral attack on PUFFIN and PUFFIN-like SPN cipher [J]. Journal of National University of Defense Technology, 2010, 32(3): 139 – 143. (in Chinese)

[12] Knudsen L, Wagner D. Integral cryptanalysis [C]// Proceedings of Fast Software Encryption, 2002, 2365: 112 – 127.

[13] Lei D, Li C, Feng K Q. New observation on camellia[C]// Proceedings of Selected Areas in Cryptography, 2005, 3897: 51 – 64.

[14] 王薇,王小云. 对 CLEFIA 算法的饱和度分析[J]. 通信学报, 2008, 29(10): 88 – 92.
WANG Wei, WANG Xiaoyun. Saturation cryptanalysis of CLEFIA[J]. Journal on Communications, 2008, 29(10): 88 – 92. (in Chinese)

[15] Z'aba M R, Raddum H, Henricksen M, et al. Bit-pattern based integral attack [C] //Proceedings of Fast Software Encryption, 2008, 5086: 363 – 381.

[16] 李超, 孙兵, 李瑞林. 分组密码的攻击方法与实例分析[M]. 北京:科学出版社, 2010.
LI Chao, SUN Bing, LI Ruilin. Cryptanalytic methods and instance analysis of block ciphers [M]. Beijing: Science Press, 2010. (in Chinese)

(上接第 120 页)

[6] Li H D. A practical algorithm for L_∞ triangulation with outliers[C]//Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), New York, IEEE, 2007: 1 – 8.

[7] Seo Y, Lee H, Lee S W. Outlier removal by convex optimization for L_∞ approaches[C] //Proceedings of Pacific Rim Symposium on Image and Video Technology (PSIVT), Berlin, Springer, 2009: 203 – 214.

[8] Ke Q F, Kanade T. Quasiconvex optimization for robust geometric reconstruction [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007, 29(10): 1834 – 1847.

[9] Csurka G, Zeller C, Zhang Z Y, et al. Characterizing the uncertainty of the fundamental matrix[J]. Computer Vision and Image Understanding, 1997, 68 (1): 18 – 36.

[10] Lavine M. Introduction to statistical thought[M]. Tallahassee, USA: Orange Grove Texts Plus, 2009.

[11] Silverman B W. Density estimation: for statistics and data analysis[M]. London, UK: Chapman and Hall, 1986.

[12] Wang H Z, Suter D. Robust adaptive-scale parametric model estimation for computer vision [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004, 26(11): 1459 – 1474.

[13] Lowe D G. Distinctive image features from scale-invariant keypoints [J]. International Journal of Computer Vision, 2004, 60(2): 91 – 110.

[14] Triggs B, Mclauchlan P F, Hartley R I, et al. Bundle adjustment—a modern synthesis [C]//Proceedings of ICCV'99: Proceedings of the International Workshop on Vision Algorithms: Theory and Practice, London, Springer Verlag, 2000: 298 – 375.