分组密码分析

清华大学计算机系
于红波
2023年3月29日

分组密码分析—攻击类型

- 唯密文攻击: 攻击者 除了截获的密文，没有其他可用信息
- 已知明文攻击：攻击者仅知道当前密钥下一些明密文对
- 选择明文攻击：攻击者能够获得当前密钥下的一些特定的明文对应的密文 (CPA)
  - 适应性选择明文
  - 非适应性选择明文
- 选择密文攻击：攻击者能够获得当前密钥下的一些特定的密文对应的明文(CCA)

分组密码分析—攻击复杂度

- 时间复杂度：实施攻击所需要的计算步骤，通常由加密解密次数表示
- 空间复杂度：攻击算法所需要的存储量
- 数据复杂度：实施攻击所需要输入的数据量，已知的明密文对的数量

用数据复杂度和时间复杂度的主要部分来刻画攻击的复杂度， 例如穷尽密钥搜索攻击、差分分析

通用密码分析方法

强力攻击：对任何分组密码都适用，攻击复杂度只依赖于分组长度和密钥长度

- 穷举密钥搜索攻击：唯密文攻击、已知明文攻击。k为密钥长度，平均复杂度为2^{k-1}加密
- 字典攻击：攻击者搜集明密文对，并把它们编排成一个“字典”。如果n是分组长度，需要2^n个明密文
- 查表攻击：选择明文的攻击，对一个给定的明文x, 用所有的2^k个密钥预计算密文$y_k=E_k(x)$, 构造一张有序表$\{(y_k,K)\}$

通用密码分析方法（续）

- 时间存储折中攻击（Time-Memory trade-off)
  - 1980年Martin Hellmen
  - 一种选择明文攻击方法，由穷尽密钥搜索攻击和查表攻击两种方法混合而成
- 对固定的明文P 和某个密文C_0，已知加密算法S，目标是恢复出加密过程中使用的密钥K_0：

$$C_0 = S_{K_0}(P)$$

通用密码分析方法（续）

$K_0=SP$

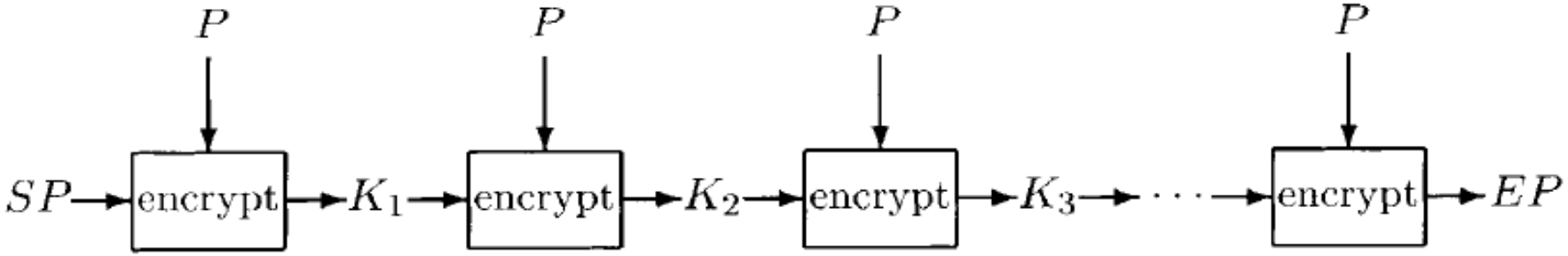
$K_1=E(P, SP)$

$K_2=E(P, K_1)$

$K_3=E(P, K_2)$

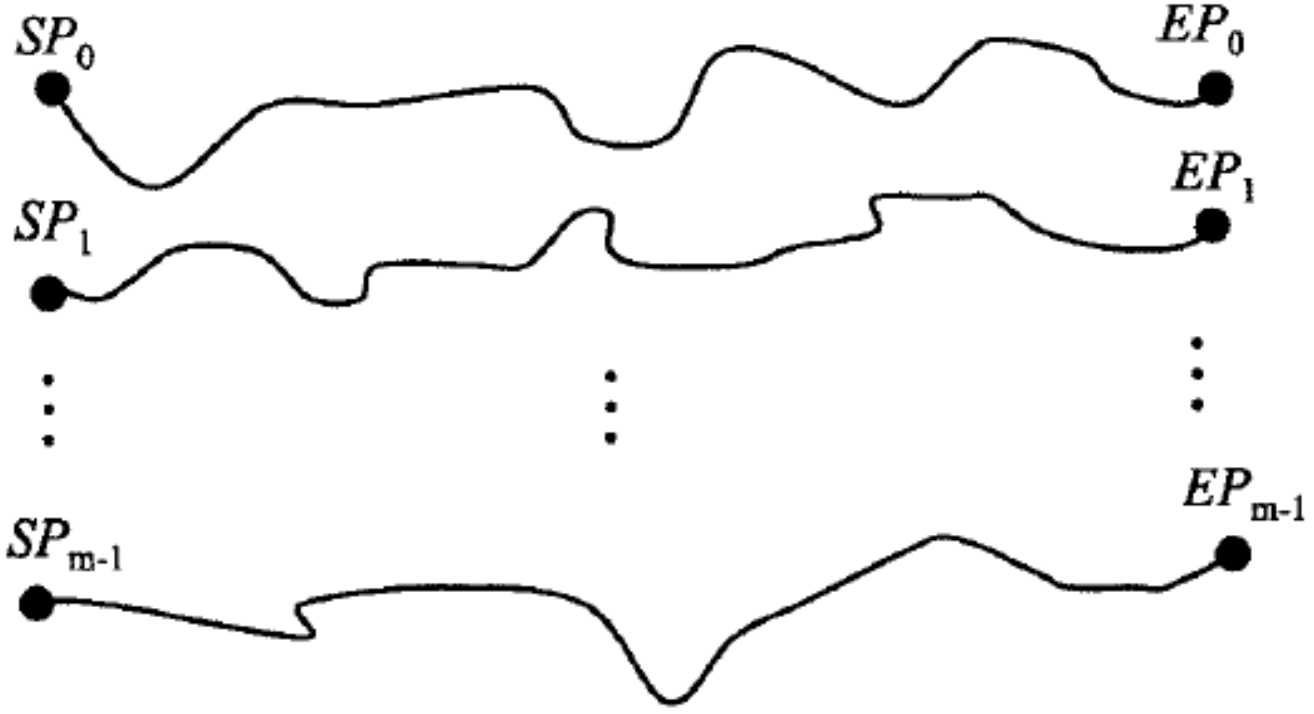
……

$EP=K_{t-1}=E(P, K_{t-2})$



通用密码分析方法（续）

- 计算m个链，每个链的长度为t



- 存储$(SP_0,EP_0),(SP_1,EP_1),\ldots,(SP_{m-1},EP_{m-1})$

通用密码分析方法（续）

- 攻击过程：选择相同的明文P,取得其对应的密文C，计算

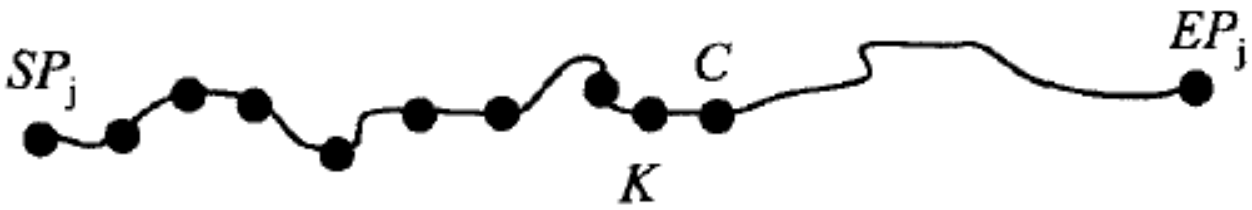
$$X_0=C$$
$$X_1=E(P,X_0)$$
$$X_2=E(P,X_1)$$
$$\cdots\cdots$$

把X_i与存储的末节点$EP_0,EP_1,\ldots,EP_{m-1}$相比较，假设$X_i=EP_j$

通用密码分析方法（续）

- 则从SP_j开始，构造

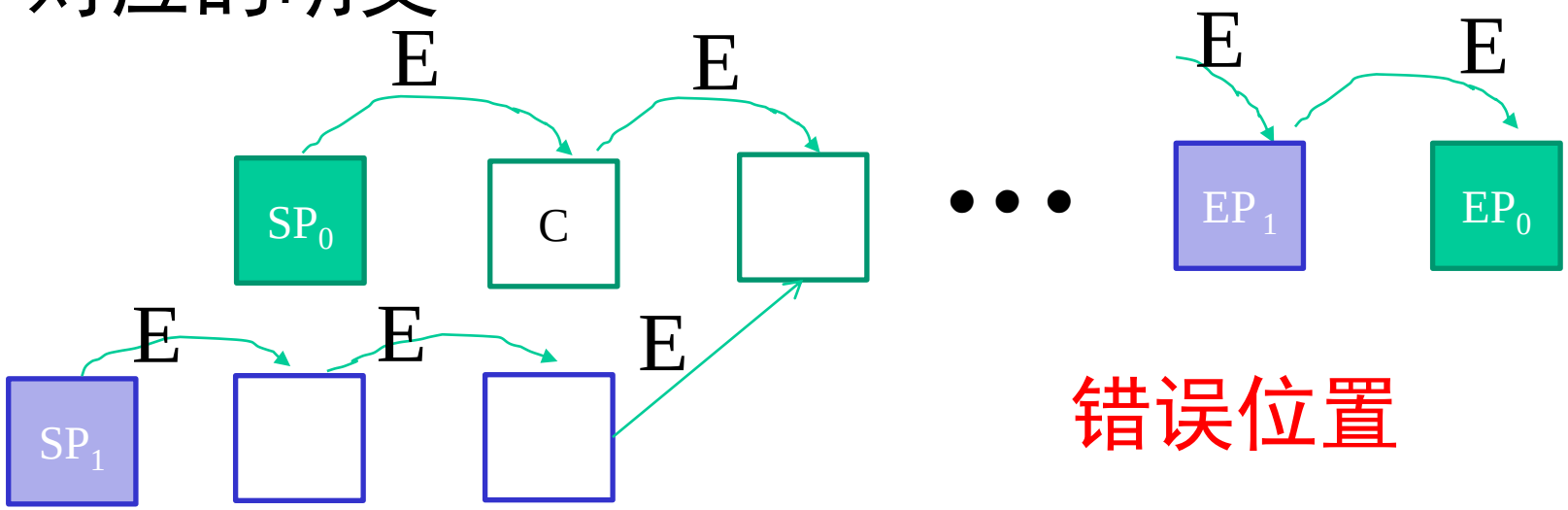
$$Y_0=SP_j$$
$$Y_1=E(P,Y_0)$$
$$Y_2=E(P,Y_1)$$
$$\cdots\cdots$$
$$Y_{t-i-1}=E(P,Y_{t-i-2})$$
$$Y_{t-i}=X_0=E(P,Y_{t-i-1})$$



TMTO

- 考虑以下几个问题：

1. C, E(C),……E^{t-1}(C)没有一个包含在尾节点中
2. 若存在$E^j(C)=EP_i=E^t(SP_i)$, 是否一定能找到C对应的明文



TMTO

第二个问题解决办法：计算所有的C，$E(C)$，…，$E^{t-1}(C)$,找到所有可能的尾节点，一一验证

3. 在一个s.t的表中，有多少个不同的点？

当第i行产生时，SP_i 是一个新的点的概率

$$1-(i-1)\bullet t/N$$

$E(SP_i)$是一个新的点的概率

$$1-((i-1)\bullet t+1)/N$$

TMTO

$$\Pr[SP_i \text{ is new}] \bullet \Pr[E(SP_i) \text{ is new} | SP_i \text{ is new}]$$

$$\geq (1 - \frac{(i-1)\bullet t}{N}) \bullet (1 - \frac{(i-1)\bullet t + 1}{N})$$

$$> (1 - \frac{(i-1)\bullet t + 1}{N})^2$$

$$\Pr[E^{t-1}(SP_i) \text{ is new}]$$

$$\geq (1 - \frac{i \cdot t}{N})^t = \left[(1 - \frac{i \cdot t}{N})^{\frac{N}{i \bullet t}} \right]^{\frac{i \cdot t^2}{N}} \approx e^{-it^2/N}$$

TMTO

- 当$it^2>N$时上式概率较高。 设$st^2=N/2$, 则表中不同的点的个数为

$$\sum_{i=1}^{s}\sum_{j=0}^{t-1}\Pr[E^j(SP_i)\text{ is new}]\geq\sum_{i=1}^{s}\sum_{j=0}^{t-1}\frac{1}{2}=\frac{st}{2}$$

- 则任意的点C在表中的概率至少为$\frac{st}{2N}=\frac{1}{4t}$
- 产生T=4t个独立的表，则C在其中一个表中的概率为

$$1-\Pr[\text{no table contains C}]=1-(1-\frac{1}{4t})^{4t}\approx 1-e^{-1}=0.63$$

TMTO

- 如何产生T=4t个独立的表？

对每个表应用一个不同的函数F_i：F_1，…F_T

对第i个表，定义$E_i=F_i$。E，如Fi(x)=x+ci

复杂度计算：

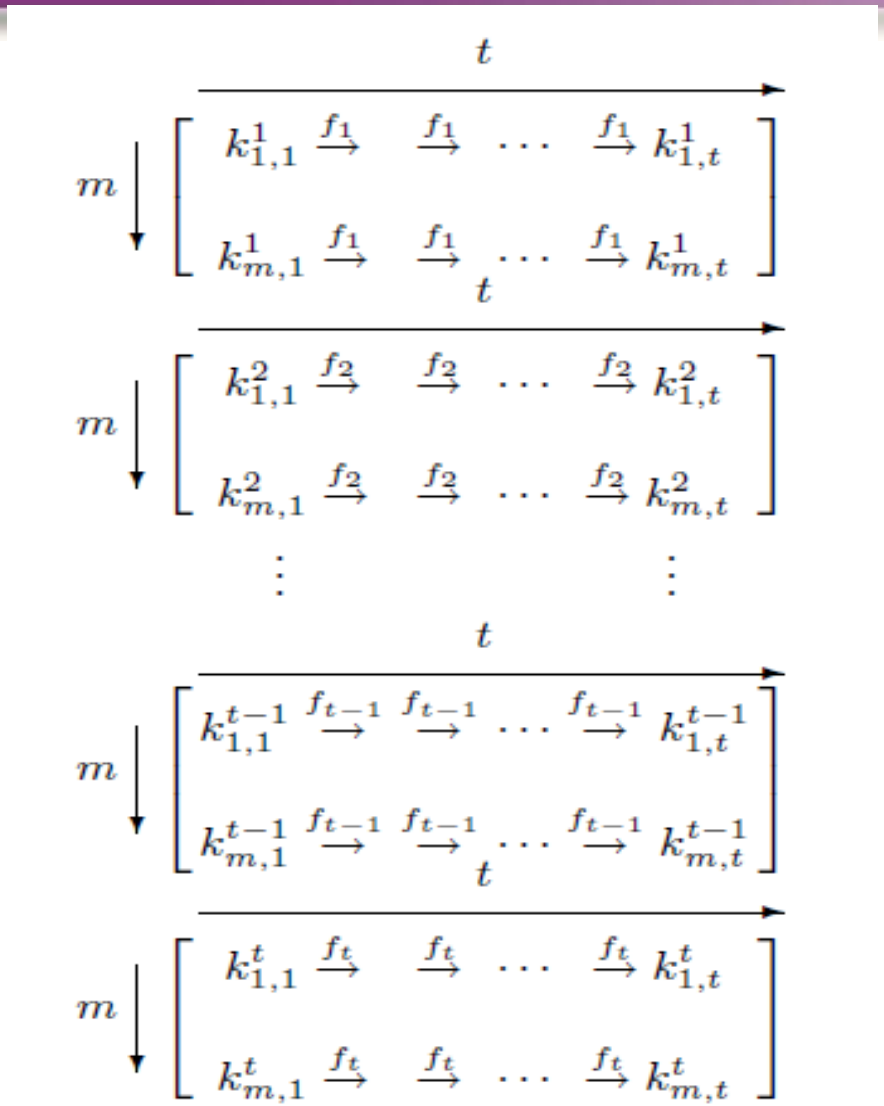
若$st^2=N/2$, 存储复杂度O(s.T)=O(S.t)=O(N/t)

时间复杂度$O(tT)=O(t^2)$

取$t=N^{1/3}=2^{l/3}$

存储和计算复杂度都为$O(2^{2l/3})$

TMTO



- 实际攻击中
  - 预计算$O(2^n)$
  - 存储$O(2^{2/3n})$
  - 计算$O(2^{2/3n})$
- 对于DES，预计算2^{56}，时间和存储2^{38}

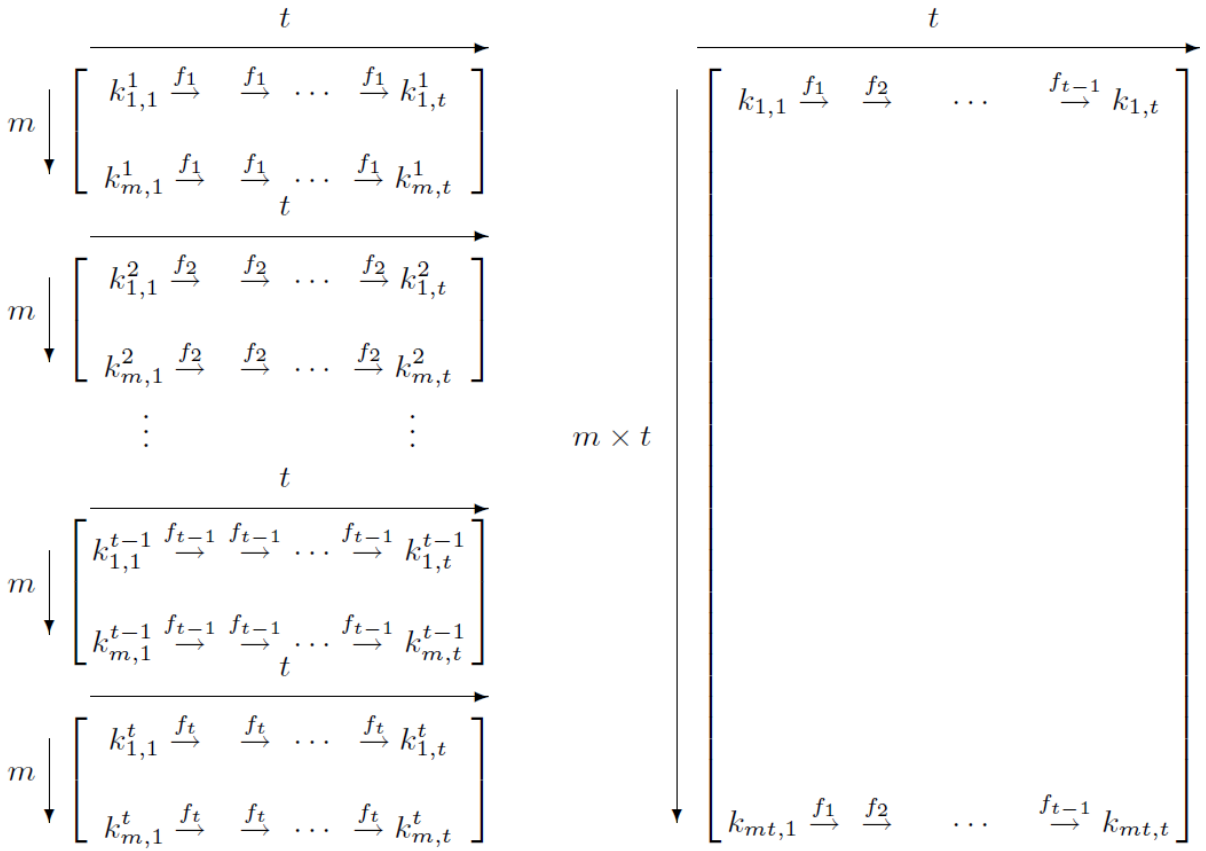
TMTO

- 如何处理输入和输出长度不同？如用于寻找Hash函数的原像（口令）

定义一个函数fi, 将$\{0,1\}^l$影射到一个口令空间。

Rainbow表（彩虹表）

- 2003年由Philippe Oechslin 提出,其效率是经典方法的2倍



分组密码的分析方法

- 差分分析
  - 1990年Crypto，E.Biham和A.Shamir提出，是一种选择明文的攻击，攻击到15轮DES
  - 1992年Crypto，改进的分析，攻击到16轮DES，复杂度2^{47}选择明文
- 线性分析
  - Matsui M, Eurocrypt 提出
  - 线性分析（linear cryptanalysis）是一种已知明文攻击
  - 对16轮DES，复杂度为2^{43}已知明文攻击

DES差分密码分析

- 1990年Crypto，E.Biham和A.Shamir提出，攻击到15轮
- 1992年Crypto，改进的分析，攻击到16轮

分组密码的差分攻击

- 基本观点：给定加密算法，比较两个明文的异或与相应的两个密文的异或的概率分布情况

  记 $P_1 \oplus P_2=\Delta P$，$C_1 \oplus C_2=\Delta C$

  任给$X=\Delta P$, $Y=\Delta C$,

  研究差分（$\Delta P \rightarrow \Delta C$） 的概率分布问题

  特别研究是否有高于平均概率的差分成立

- 如果加密算法明文/密文长度为l，则平均差分概率为$1/2^l$

寻找高概率差分的一般方法

- 通常差分$\Delta P \rightarrow \Delta C$概率主项为一些差分特征概率的乘积

$$\Delta P\ (\Delta R_0) \rightarrow \Delta R_1 \rightarrow \Delta R_2 \rightarrow \ldots\ldots\ldots \rightarrow \Delta R_{n-1} \rightarrow \Delta C$$

- 差分特征（轮差分）

$$\Delta P \rightarrow \Delta R_1,\quad \Delta R_1 \rightarrow \Delta R_2,\quad \ldots\ldots\ldots\ldots,\quad \Delta R_{n-1} \rightarrow \Delta C$$

DES线性和非线性部件

- 线性部件：已知输入差分，能够计算输出差分
  - XOR
  - P置换: $P(x) \oplus P(y)=P(x \oplus y)$
  - E扩展: $E(x) \oplus E(y)=E(x \oplus y)$
- 非线性部件
  - S-box: $S(x) \oplus S(y) \neq S(x \oplus y)$

S-box差分分布

- 对于每个S-box,共有64*64个输入对，每一个有一个输入差分对对应着64组值，共有16个输出差分，故平均每一个输入输出差分对 对应着4个值

S-box差分分布（续）

例题 设第一个 S-盒 S_1 的输入异或为 110100，那么 $\Delta(110100)=\{(000000, 110100),(000001,110101),\cdots,(111111,001011)\}$。现在我们对集合 $\Delta(110100)$ 中的每一个有序对，计算 S_1 的输出异或。例如，$S_1(000000)=E_{16}=1110$，$S_1(110100)=9_{16}=1001$，所以有序对(000000,110100)的输出异或为 0111。

对 $\Delta(110100)$ 中的每一个对，都做这样的处理后，可获得下列的输出异或分布：

0000	0001	0010	0011	0100	0101	0110	0111
0	8	16	6	2	0	0	12

1000	1001	1010	1011	1100	1101	1110	1111
6	0	0	0	0	8	0	6

具有输入异或110100的所有可能输入

输出异或	可能的输入
0000	
0001	000011,001111,011110,011111 101010,101011,110111,111011
0001	000011,001111,011110,011111 101010,101011,110111,111011
0010	000100,000101,001110,010001 010010,010100,011010,011011 100000,100101,0101110,101110 101111,110000,110001,111010
0011	000001,000010,010101,100001 110101,110110
0100	010011,100111
0101	
0110	
0111	000000,001000,001101,0101111 011000,011101,100011,101001 101100,110100,111001,111100
1000	001001,001100,011001,101101 111000,111101
1001	
1010	
1011	
1100	
1101	000110,010000,010110,011100 100010,100100,101000,110010
1110	
1111	000111,001010,001011,110011 111110,111111

S-1盒差分分布表

Input XOR	Output XOR															
	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	A_x	B_x	C_x	D_x	E_x	F_x
0_x	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_x	0	0	0	6	0	2	4	4	0	10	12	4	10	6	2	4
2_x	0	0	0	8	0	4	4	4	0	6	8	6	12	6	4	2
3_x	14	4	2	2	10	6	4	2	6	4	4	0	2	2	2	0
4_x	0	0	0	6	0	10	10	6	0	4	6	4	2	8	6	2
5_x	4	8	6	2	2	4	4	2	0	4	4	0	12	2	4	6
6_x	0	4	2	4	8	2	6	2	8	4	4	2	4	2	0	12
7_x	2	4	10	4	0	4	8	4	2	4	8	2	2	2	4	4
8_x	0	0	0	12	0	8	8	4	0	6	2	8	8	2	2	4
9_x	10	2	4	0	2	4	6	0	2	2	8	0	10	0	2	12
A_x	0	8	6	2	2	8	6	0	6	4	6	0	4	0	2	10
B_x	2	4	0	10	2	2	4	0	2	6	2	6	6	4	2	12
C_x	0	0	0	8	0	6	6	0	0	6	6	4	6	6	14	2
D_x	6	6	4	8	4	8	2	6	0	6	4	6	0	2	0	2
E_x	0	4	8	8	6	6	4	0	6	6	4	0	0	4	0	8
F_x	2	0	2	4	4	6	4	2	4	8	2	2	2	6	8	8
									⋮							
30_x	0	4	6	0	12	6	2	2	8	2	4	4	6	2	2	4
31_x	4	8	2	10	2	2	2	2	6	0	0	2	2	4	10	8
32_x	4	2	6	4	4	2	2	4	6	6	4	8	2	2	8	0
33_x	4	4	6	2	10	8	4	2	4	0	2	2	4	6	2	4
34_x	0	8	16	6	2	0	0	12	6	0	0	0	0	8	0	6
35_x	2	2	4	0	8	0	0	0	14	4	6	8	0	2	14	0
36_x	2	6	2	2	8	0	2	2	4	2	6	8	6	4	10	0
37_x	2	2	12	4	2	4	4	10	4	4	2	6	0	2	2	4
38_x	0	6	2	2	2	0	2	2	4	6	4	4	4	6	10	10
39_x	6	2	2	4	12	6	4	8	4	0	2	4	2	4	4	0
$3A_x$	6	4	6	4	6	8	0	6	2	2	6	2	2	6	4	0
$3B_x$	2	6	4	0	0	2	4	6	4	6	8	6	4	4	6	2
$3C_x$	0	10	4	0	12	0	4	2	6	0	4	12	4	4	2	0
$3D_x$	0	8	6	2	2	6	0	8	4	4	0	4	0	12	4	4
$3E_x$	4	8	2	2	2	4	4	14	4	2	0	2	0	8	4	4
$3F_x$	4	8	4	2	4	0	2	4	4	2	4	8	8	6	2	2

对一个固定的输入差分,输出差分可能值的比例

S box	Percentage
S1	79.4
S2	78.6
S3	79.6
S4	68.5
S5	76.5
S6	80.4
S7	77.2
S8	77.1

利用S-box恢复密钥

- 已知E和E^*，S1-盒输出差分为△C， 则S1盒的输入为E ⊕ K和E’ ⊕ K. 如何恢复密钥K?

例：设E=000001，E^*=110101， △C=1101。

查表可得N_1(110100,1101)=8, 所以在集合

S(110100,1101)={000110,010000,010110,011100,100010,100100,101000,110010}

故密钥可能的集合为

K={000111, 010001, 010111, 011101, 100011, 100101, 101001, 110011}

3轮DES

$R_3=L_2 \oplus f(R_2,K_3)$

$=R_1 \oplus f(R_2,K_3)$

$=L_0 \oplus f(R_0,K_1) \oplus f(R_2,K_3)$

同理,

$R_3'= L_0' \oplus f(R_0',K_1) \oplus f(R_2',K_3)$

选择$R_0=R_0'$,则

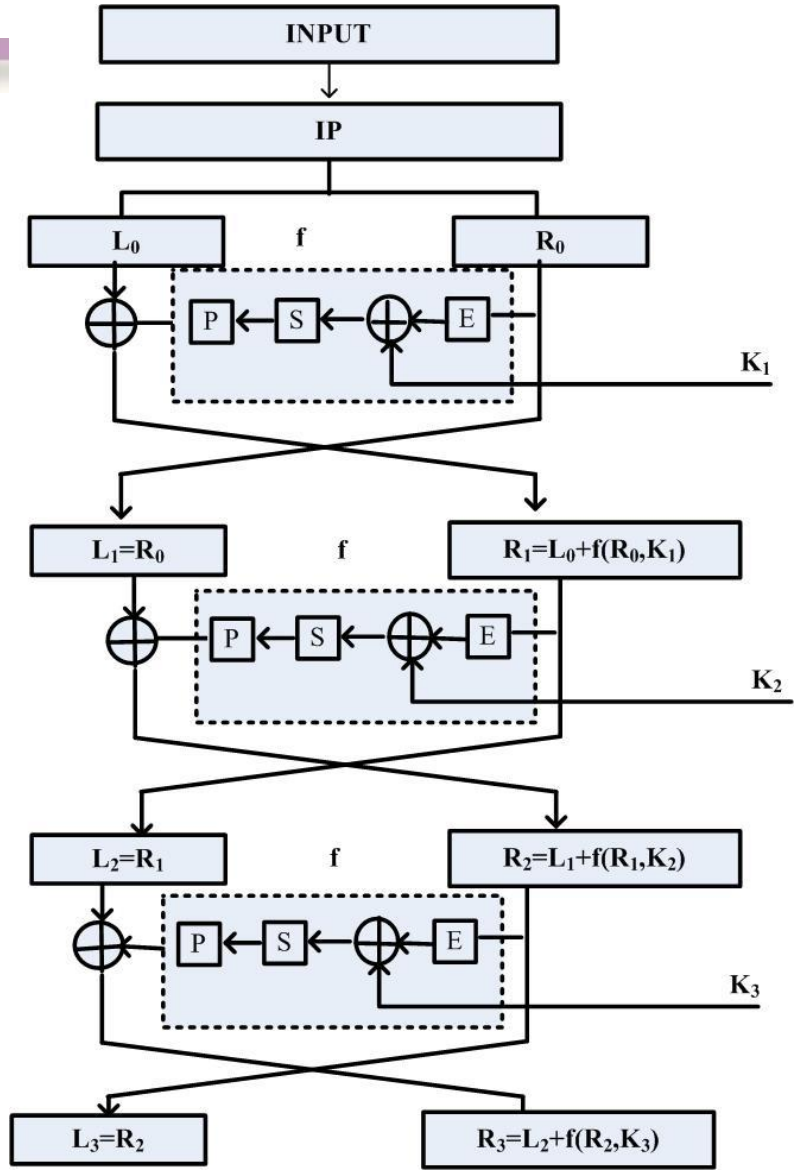
$\triangle R_3= \triangle L_0 \oplus f(R_2,K_3) \oplus f(R_2',K_3)$

则

$f(R_2,K_3) \oplus f(R_2',K_3) =\triangle L_0 \oplus \triangle R_3$

S-box输入为$E(L_3) \oplus K_3$和$E(L'_3) \oplus K_3$,

输出差分为$P^{-1}(\triangle L_0 \oplus \triangle R_3)$



3轮DES差分分析

明文	密文
748502CD38451097 3874756438451097	03C70306D8A09F10 78560A0960E6D4CB
486911026ACDFF31 375BD31F6ACDFF31	45FA285BE5ADC730 134F7915AC253457
357418DA013FEC86 12549847013FEC86	D8A31B2F28BBC5CF 0F317AC2B23CB944

3轮DES差分分析

- 对第一对明密文第3轮S盒的输入

$$E = E(L_3) = 000000000111111000001110100000000110100000001100$$

$$E' = E(L_3') = 101111110000001010101100000001010100000001010010$$

S盒的输出差分

$$\Delta C = C \oplus C' = P^{-1}(\Delta R_3 \oplus \Delta L_0) = 10010100101110101011011011001111$$

查表可得$N_1(101111,1001)=4$, 所以在集合

$S(101111,1001)=\{000000, 000111, 101000, 101111\}$

故密钥可能的集合为

$K=\{000000, 000111, 101000, 101111\}$

3轮DES差分分析

- 建立8个可能值是1~64的计数器，用以统计可能的密钥出现的次数

J_1															
1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0
0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

J_2															
0	0	0	1	0	3	0	0	1	0	0	1	0	0	0	0
0	1	0	0	0	2	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0	1	0	1	0	0	0	1	0
0	0	1	1	0	0	0	0	1	0	1	0	2	0	0	0

3轮DES差分分析

- 构造出48比特的子密钥

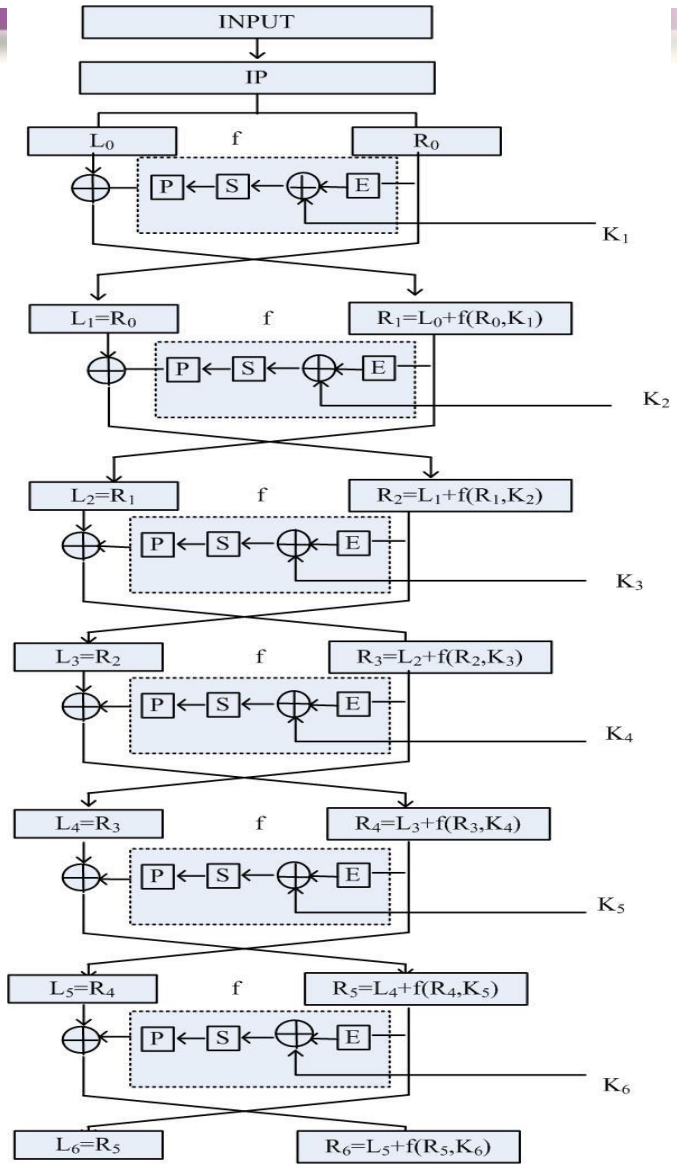
$$J_1 = 101111 \qquad J_2 = 000101$$
$$J_3 = 010011 \qquad J_4 = 000000$$
$$J_5 = 011000 \qquad J_6 = 000111$$
$$J_7 = 000111 \qquad J_8 = 110001$$

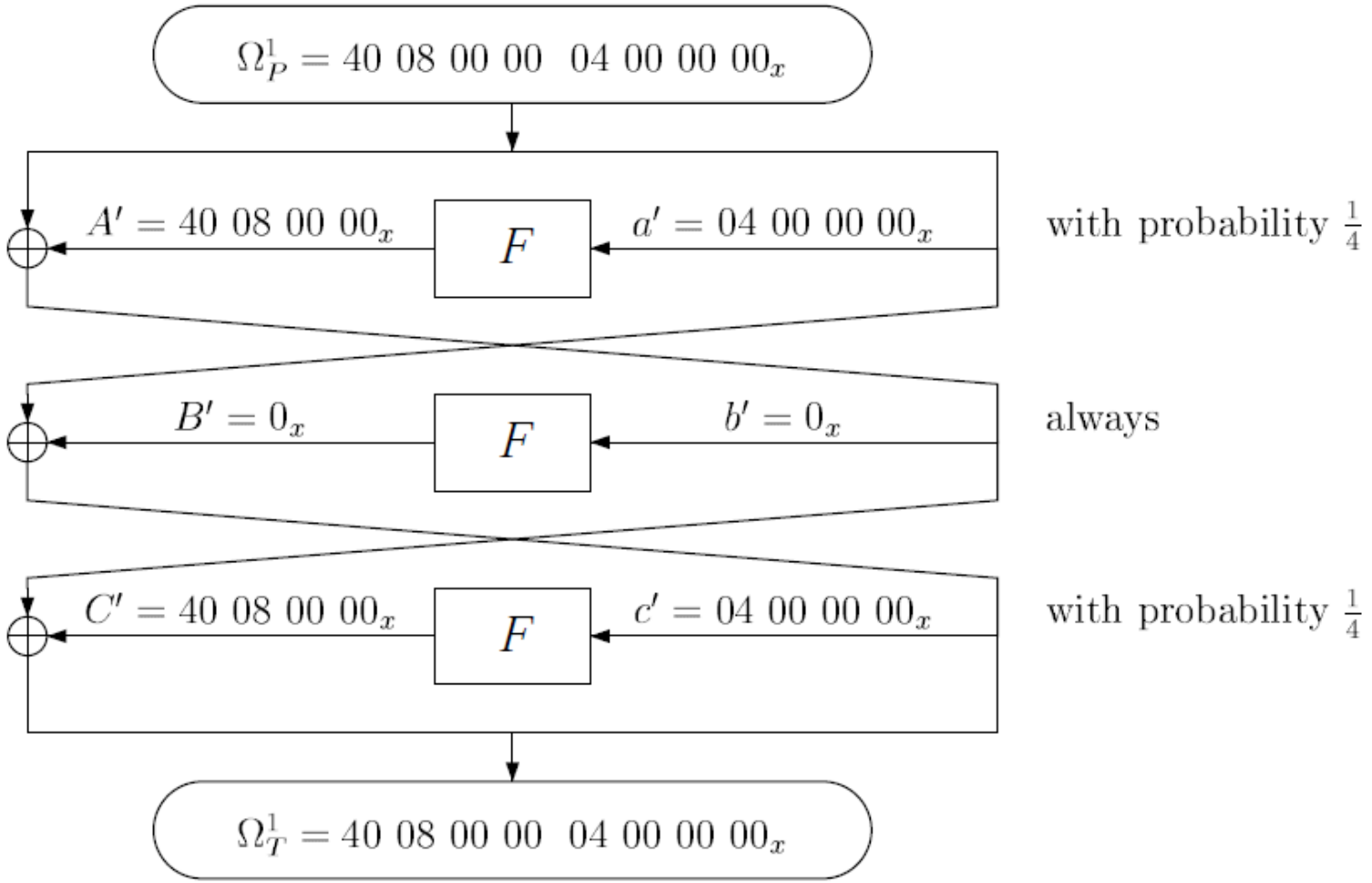
- 查找第三轮密钥方案，构造出种子密钥的48比特

0001101 0110001 01?01?0 1?00100

0101001 0000??0 111?11? ?100011

6轮DES差分分析


INPUT
IP
L_0
f
R_0
P
S
E
K_1
$L_1=R_0$
f
$R_1=L_0+f(R_0,K_1)$
K_2
$L_2=R_1$
f
$R_2=L_1+f(R_1,K_2)$
K_3
$L_3=R_2$
f
$R_3=L_2+f(R_2,K_3)$
K_4
$L_4=R_3$
f
$R_4=L_3+f(R_3,K_4)$
K_5
$L_5=R_4$
f
$R_5=L_4+f(R_4,K_5)$
K_6
$L_6=R_5$
$R_6=L_5+f(R_5,K_6)$


$\Omega_P^1 = 40\ 08\ 00\ 00\ \ 04\ 00\ 00\ 00_x$
$A' = 40\ 08\ 00\ 00_x$
F
$a' = 04\ 00\ 00\ 00_x$
with probability $\frac{1}{4}$
$B' = 0_x$
F
$b' = 0_x$
always
$C' = 40\ 08\ 00\ 00_x$
F
$c' = 04\ 00\ 00\ 00_x$
with probability $\frac{1}{4}$
$\Omega_T^1 = 40\ 08\ 00\ 00\ \ 04\ 00\ 00\ 00_x$

6轮DES密钥恢复过程

- 选择满足$\triangle L_0$=40080000和 $\triangle R_0$=04000000的明密文对(L_0R_0,L_6R_6),$(L'_0R'_0, L'_6R'_6)$, 则

$$R_6=L_5 \oplus f(R_5,K_6)=R_4 \oplus f(R_5,K_6)$$

$$=L_3 \oplus f(R_3,K_4) \oplus f(R_5,K_6)$$

同理 $R_6'= L_3' \oplus f(R_3',K_4) \oplus f(R_5',K_6)$

则$\triangle R_6= \triangle L_3 \oplus f(R_3,K_4) \oplus f(R'_3,K_4) \oplus f(R_5,K_6) \oplus f(R'_5,K_6)$

$\triangle R_3$=40080000和 $\triangle L_3$=04000000成立的概率为1/16。

6轮DES密钥恢复过程（续）

- 第4轮中S盒的输入异或为40080000，
  - E(40080000)=200050000000

    (0010 00|00 0000| 0000 01|01 0000| 0000 00|00 0000| 0000 00|00 0000)

    (08 00 01 10 00 00 00 00)
  - 经过E扩展后，S_2,S_5,S_6,S_7,S_8的输入差分都为0，则输出差分也为0。

1. 第6轮S盒输出差分$\triangle C=P^{-1}(\triangle R_6 \oplus 04000000)$
2. 第6轮S盒输入$E=E(L_6)$, $E'=E(L'_6)$
3. 对j={2,5,6,7,8}, 计算Set_j

密钥过滤过程

- 若5个$Set_j(j=\{2,5,6,7,8\})$中有一个$|Set_j|=0$, 则该对为错误对。对一个错误对，$|Set_j|=0$的概率大约为1/5。 5个$|Set_j|>0$的概率为$(4/5)^5\approx0.33$。大约有1/3的错误对剩下，正确对所占的比率大约为3N/16。
- 如何识别正确的密钥，攻击所需要的数据复杂度是多少？
  - 方法：计数器的方法
  - 复杂度：$(NP)^{-1}$

6轮DES差分分析

- 例：《分组密码的设计和分析》

  给定120个明文/密文对，其中73对为可识别出来的错误对，剩下的47对是“可能的”正确对

  故大约有9-10对正确对

1. 如何识别正确对？

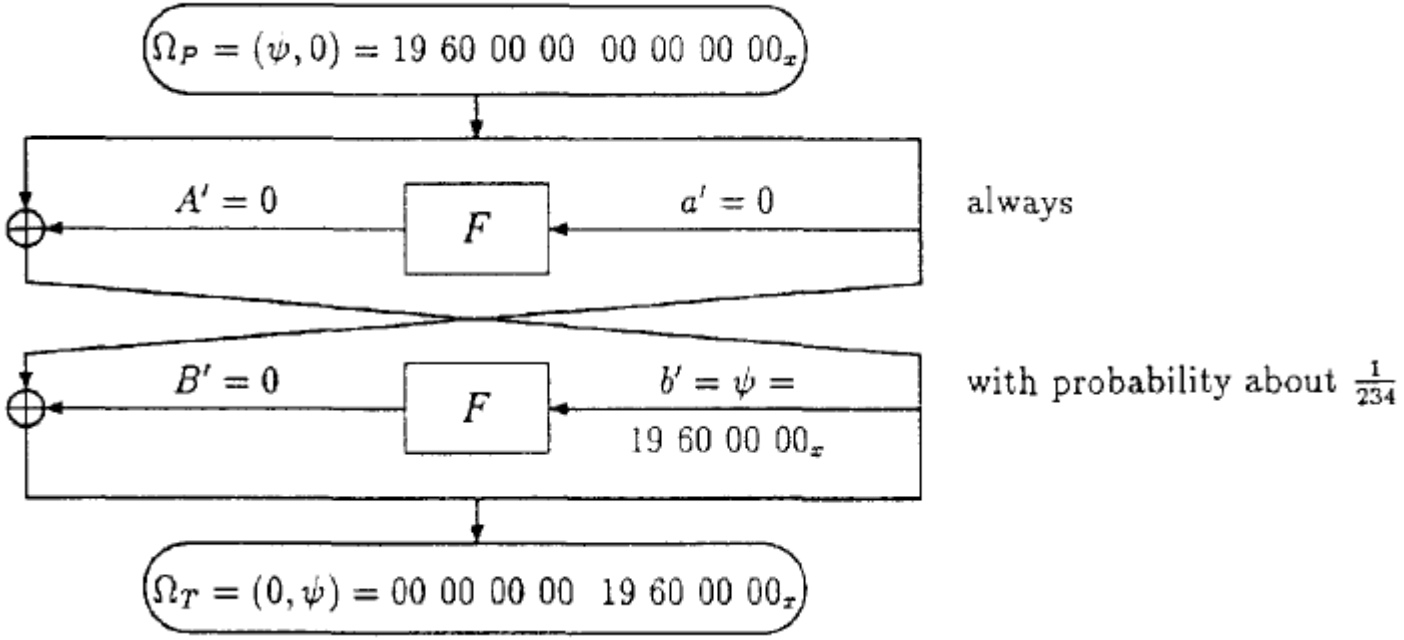
   构造可允许集

2. 如何恢复密钥？

   由正确对容易恢复密钥

16轮DES差分分析

- 第2到14轮使用迭代差分特征.



AES算法的安全性分析

- AES算法简介
- AES的积分攻击
- AES的碰撞攻击
- AES的不可能差分攻击

AES算法简介

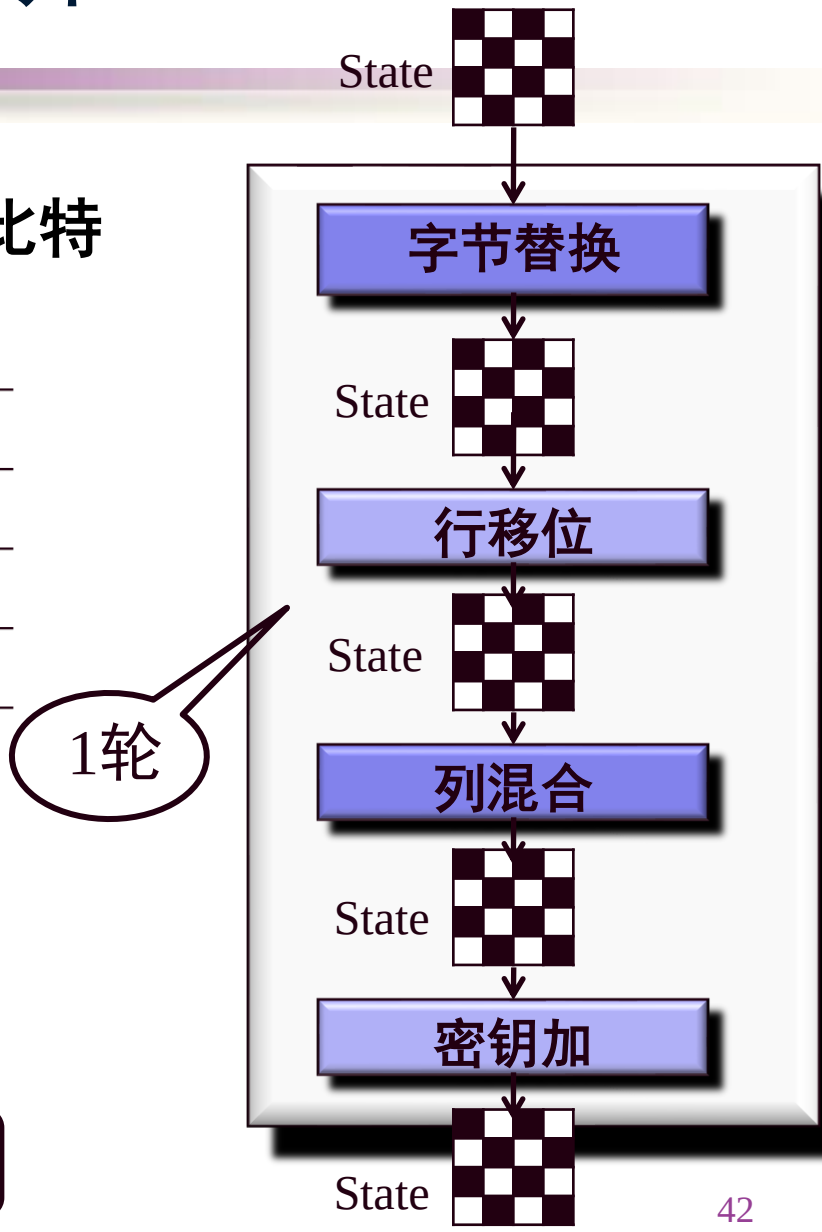
- AES算法采用SPN结构
- AES分组长度（Block Size）为128比特
- 密钥长度：128/192/256

	Key size (bit)	# rounds
AES-128	128	10
AES-192	192	12
AES-256	256	14

状态（state）

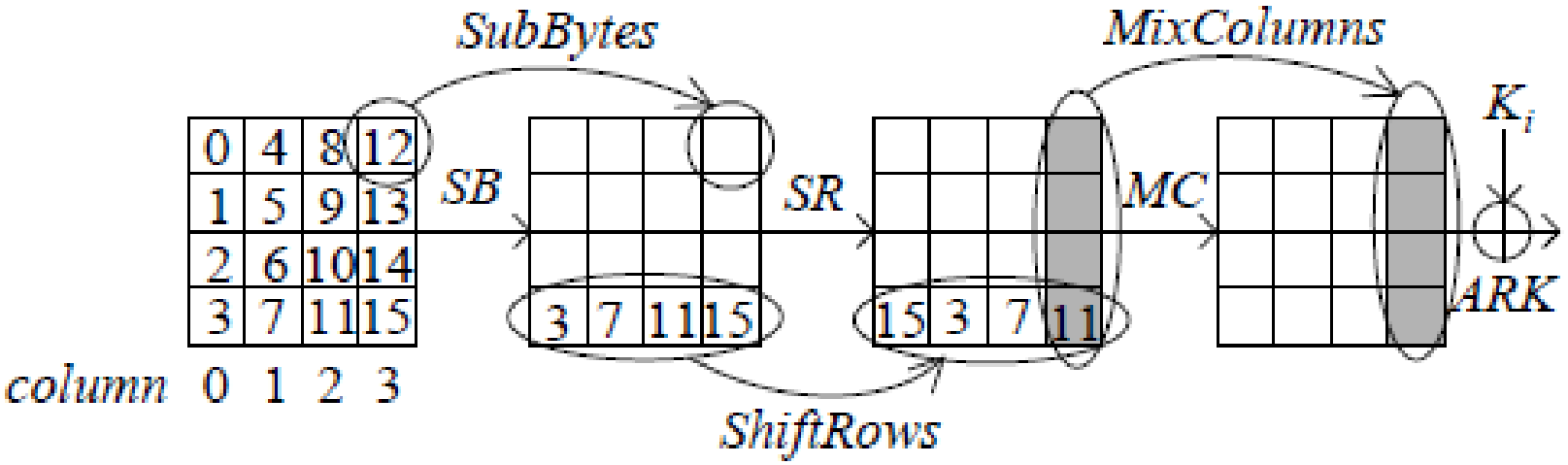
p_{00}	p_{01}	p_{02}	p_{03}
p_{10}	p_{11}	p_{12}	p_{13}
p_{20}	p_{21}	p_{22}	p_{23}
p_{30}	p_{31}	p_{32}	p_{33}

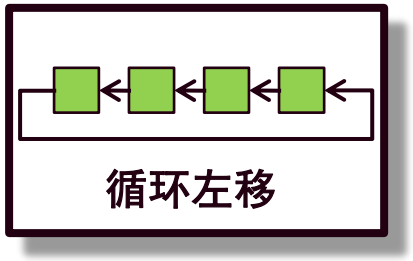
字节

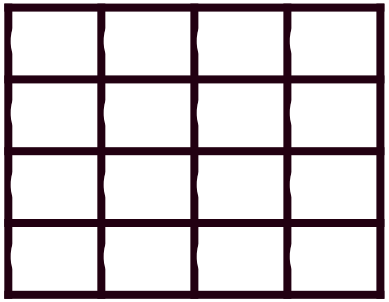


AES算法简介

轮函数

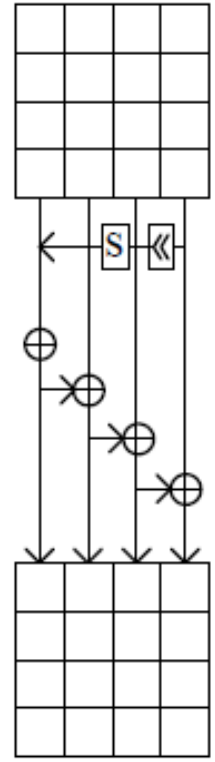

SubBytes
MixColumns
K_i
0 4 8 12
1 5 9 13
2 6 10 14
3 7 11 15
SB
SR
MC
ARK
3 7 11 15
15 3 7 11
column 0 1 2 3
ShiftRows


循环左移

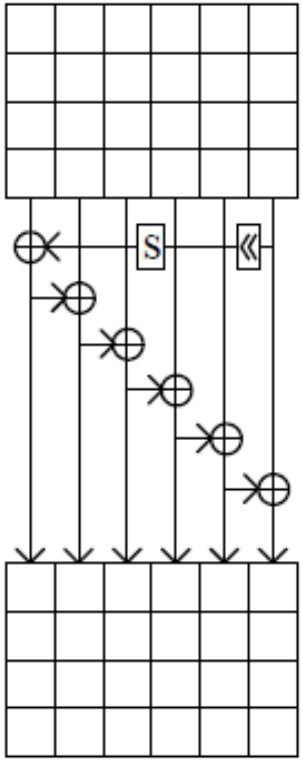


AES算法简介

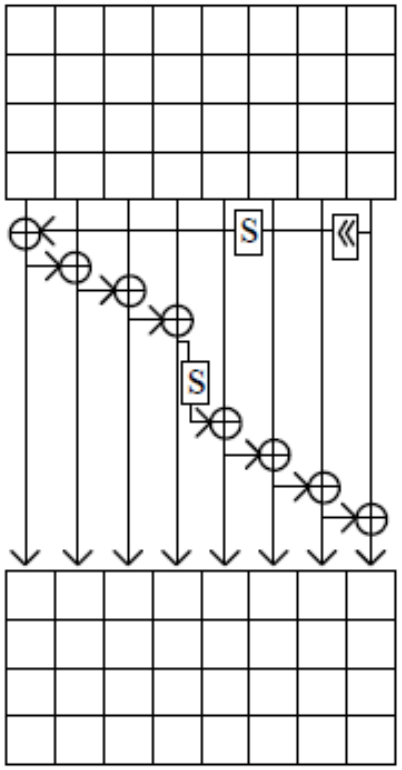
密钥调度



AES – 128



AES – 192



AES – 256

积分攻击

简 介

- 积分攻击（Integral Attack）由Daemen, Knudsen and Rijmen在1997年FSE提出的，用于分析Square算法，因此最初被称为Square Attack。
- 随着Square Attack思想的延伸，饱和度攻击（Saturation Attack）、碰撞攻击（Collision Attack）、多集合攻击（Multiset Attack）以及积分攻击（Integral Attack）等方法相继出现。

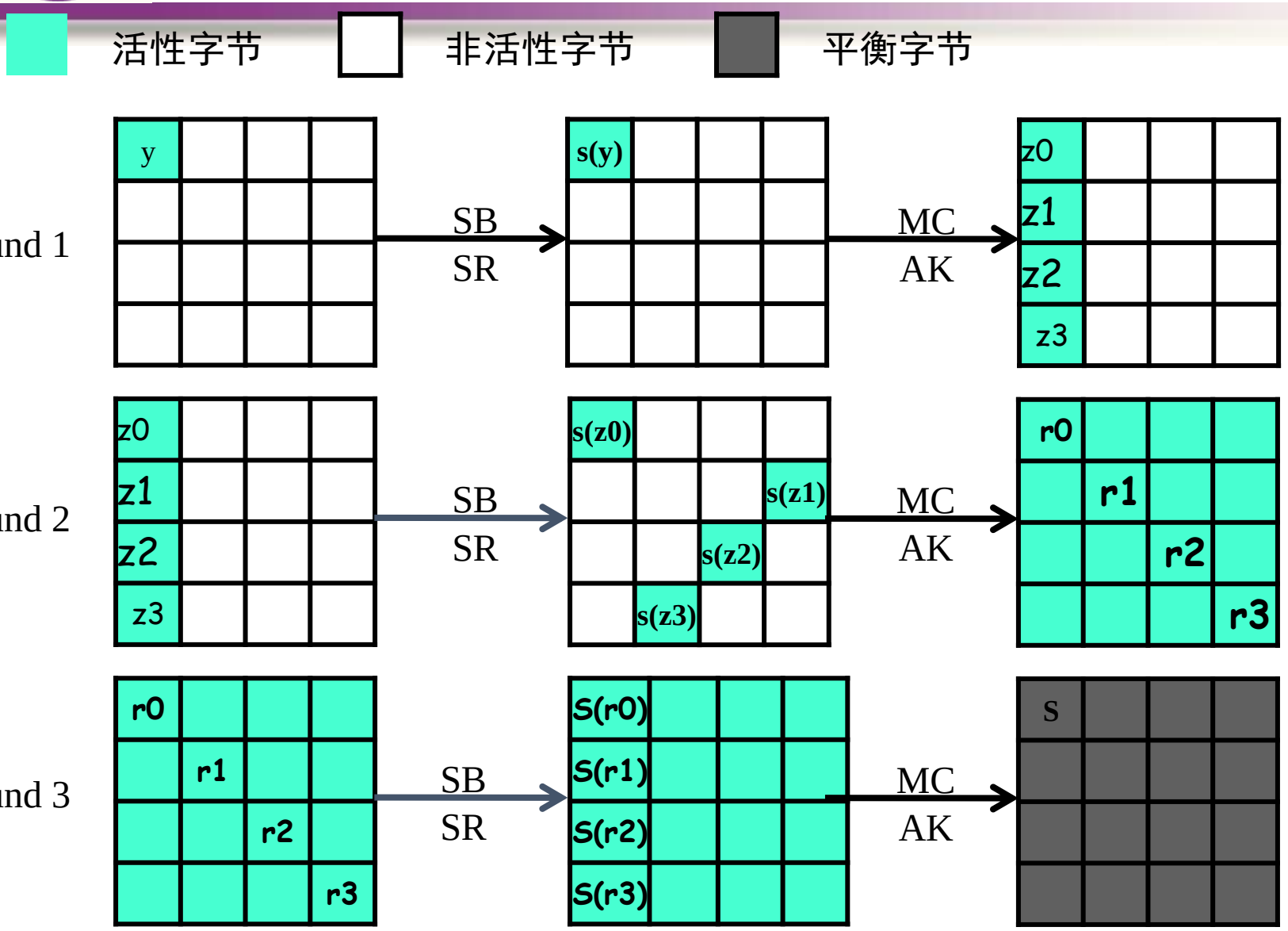
基本思想

- **积分攻击属于选择明文攻击（Chosen-plaintext）范畴，利用线性层扩散的不完全性，考虑的是某些值的和经过固定轮数加密后的特征。**

积分攻击

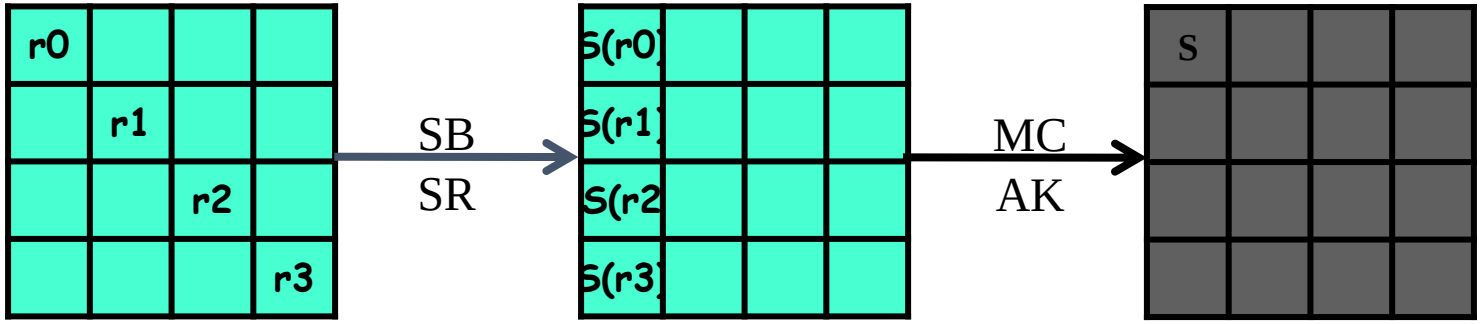
- 针对一般的分组密码（Block Cipher），假设其分组有n个子分组（Subblocks）：
  - 选择一个或几个子分组**遍历**，其余的子分组为常数。
  - 预测某些子分组经过R轮加密后的性质：
    - ◆常数：所有数据在某个子分组为相同的常值。
    - ◆活性：所有数据集合被划分为子集，在每个子集中某个子分组取遍所有可能的值称为**活性子分组**
    - ◆平衡：所有数据在这个子分组的和（异或和）为零。
    - ◆未知：不确定的值

AES的积分攻击(square)区分器



AES积分攻击

Round 3



$$\sum_{y=0}^{255} s[y] = \sum_{y=0}^{255} [02S(r_0[y]) + 03S(r_1[y]) + 01S(r_2[y]) + 01S(r_3[y]) + k_{0,0}^3]$$

$$= \sum_{y=0}^{255} 02S(r_0[y]) + \sum_{y=0}^{255} 03S(r_1[y]) + \sum_{y=0}^{255} 01S(r_2[y]) + \sum_{y=0}^{255} 01S(r_3[y]) + \sum_{y=0}^{255} k_{0,0}^3$$

$$= 0$$

AES的积分攻击

- Daemen和Rijmen给出6轮的AES分析结果（ FSE97 ）
- Lucks给出7轮AES-192 /256的攻击（3rd AES Candidate Conference）
- Ferguson等提出部分和技术，减少FSE97结果的时间复杂度，将积分攻击与相关密钥结合，分析9轮的AES-256（FSE2000）

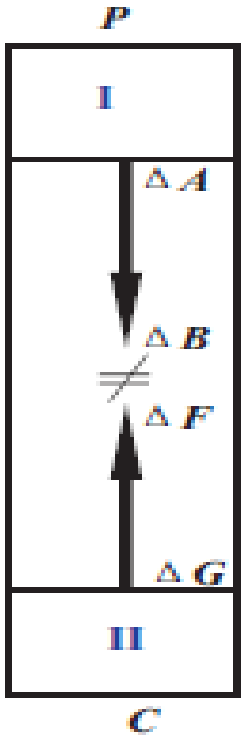
Cipher	Key size	Complexity [Data]	[Time]	Comments
Rijndael-6	(all)	2^{32} CP	2^{72}	[DR98] (previously known)
Rijndael-6	(all)	$6 \cdot 2^{32}$ CP	2^{44}	partial sums (new)
Rijndael-7	(192)	$19 \cdot 2^{32}$ CP	2^{155}	partial sums (new)
Rijndael-7	(256)	$21 \cdot 2^{32}$ CP	2^{172}	partial sums (new)
Rijndael-7	(all)	$2^{128} - 2^{119}$ CP	2^{120}	partial sums (new)
Rijndael-8	(192)	$2^{128} - 2^{119}$ CP	2^{188}	partial sums (new)
Rijndael-8	(256)	$2^{128} - 2^{119}$ CP	2^{204}	partial sums (new)
Rijndael-9	(256)	2^{85} RK-CP	2^{224}	related-key attack (new)

CP – chosen plaintext, RK-CP – related-key chosen plaintext.

不可能差分分析

简 介

不可能差分分析是Biham和Knudsen在1998年分别独立提出。

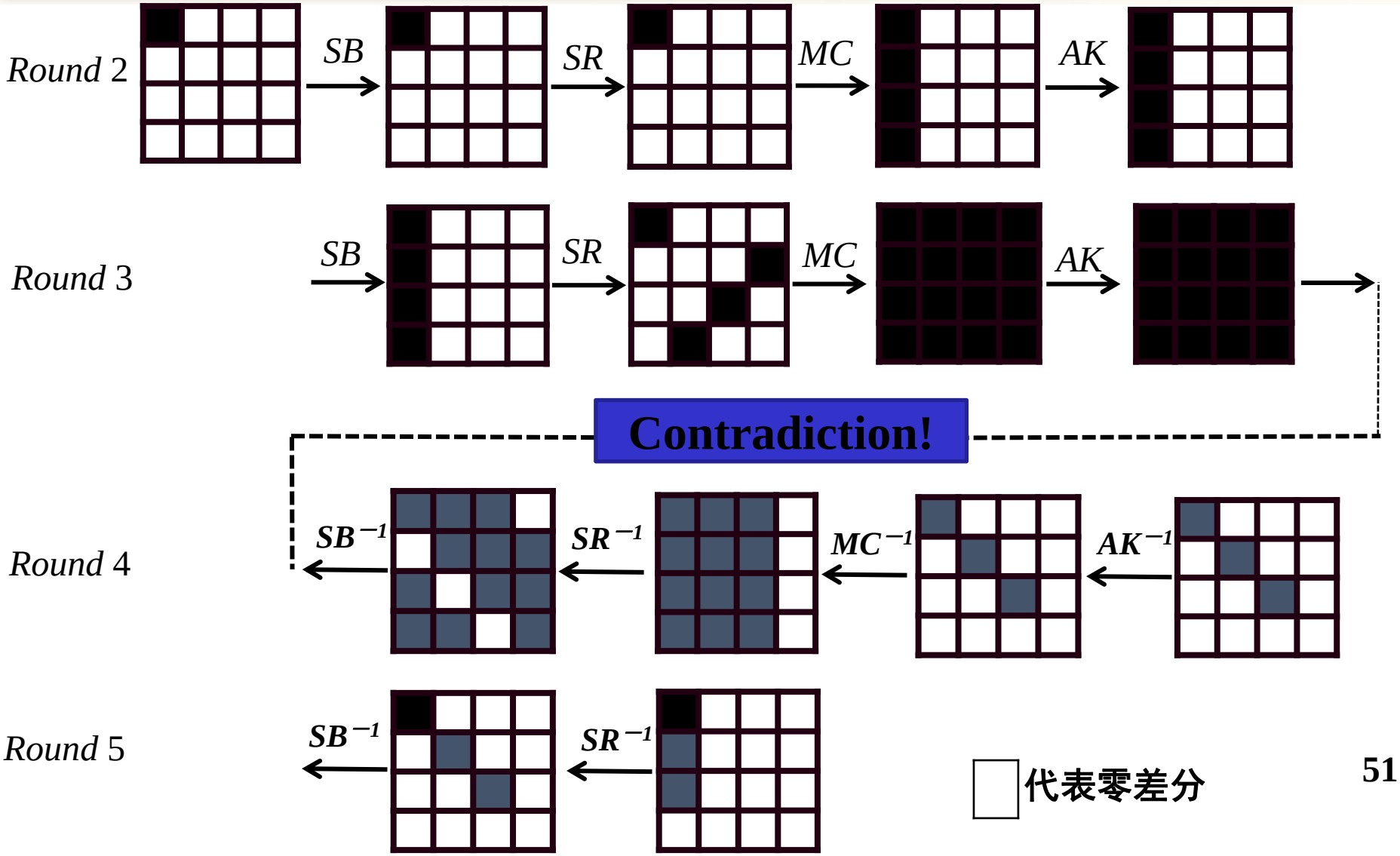


基本思想

不可能差分（概率为0的差分路径）作为密码算法与随机置换的区分器：

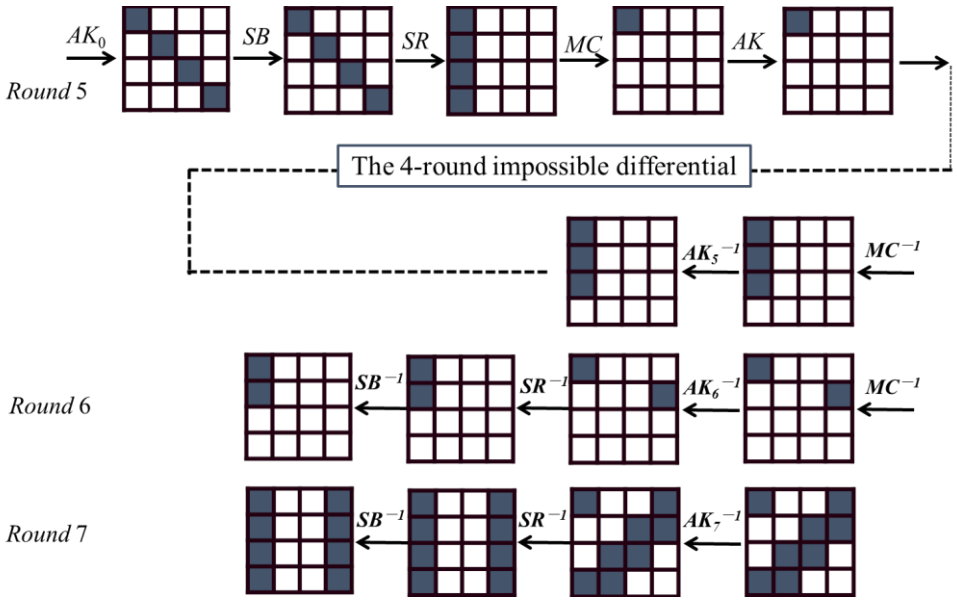
- 正确密钥加密解密后一定会满足该不可能差分
- 错误密钥加密解密后应以一定概率满足该不可能差分

3.5轮AES-128不可能差分区分器



7轮AES-128不可能差分攻击

- 头部扩1轮，尾部扩2.5轮分析7轮



- 复杂度分析
  - 数据复杂度$2^{115.5}$，时间复杂度2^{119}，存储复杂度2^{109}字节
  - 预计算部分的计算复杂度是$2^{40.5}$，存储复杂度是2^{45}

谢谢！