MasterMath Cryptology 2015 - 2/2 - Cryptanalysis

Thursday, 2 April, 2015 16:37

8. Linear Cryptanalysis (v1)

This is an introduction to linear cryptanalysis.
We'll explain the basic techniques by demonstrating how to use them against a toy cipher built similar to modern block ciphers.
(examples taken from http://www.engr.mun.ca/~howard/PAPERS/ldc_tutorial.pdf)

Modern block ciphers (like AES) proceed in rounds.
Every round uses a separate round key (or subkey) derived from the (master) key using an algorithm called the *key schedule.*

A simple structure commonly used to design block ciphers is the *substitution-permutation-network (SPN)*.

8.1. Substitution-Permutation Networks Ciphers

SPN Ciphers consist of a number of rounds.
Every round consists of the following operations:

- *key mixing*: add the round key to the current state (linear)
- *substitutions*: the state is split into words, each word is substituted using a one-to-one value mapping called an *S-BOX*. (non-linear)
- *permutation*: a linear operation performing inter-word mixing, e.g., a simple permutation over the bits of the state. (linear)

The very last round does no permutation, but instead does a *final key mixing*.
A permutation does not add security there as any attacker can immediately revert it independently of the key.
Without the final key mixing, the same would hold for the substitution.

As an example for expositition of the techniques we will consider a SPN Cipher of the following bit-oriented form (e.g., in contrast AES is byte-oriented as it uses $GF(2^8)$).
Let $L, M \in \mathbb{N}^+$, where L is the word size in bits and M is the number of words in the state.
Then the block size, and thus state size, is $L \cdot M$.
For simplicity we will use only one S-BOX for all substitution:

$\pi_S: \{0,1\}^L \to \{0,1\}^L, \pi_S \in Sym(\{0,1\}^L)$

and one permutation:

$\pi_P: \{1, \dots, LM\} \to \{1, \dots, LM\}, \pi_P \in Sym(\{1, \dots, LM\})$

Also, we will assume independent random keys for every round, instead of round keys derived from a master key.
The SPN Cipher takes as input a plaintext $P \in \{0,1\}^{LM}$ which is used as the initial state $S_0 = P$.
For round $r = 1, \dots, R$:

- Start with state S_{r-1}
- XOR round key K_r with S_{r-1}: $S_r^{(K)} = S_{r-1} \oplus K_r$
- Split state into M L-bit words $S_r^{(K)} = S_{r,1}^{(K)} | \cdots | S_{r,M}^{(K)}$ and apply π_S on each word:
  $S_r^{(KS)} = \pi_S\left(S_{r,1}^{(K)}\right) | \cdots | \pi_S\left(S_{r,M}^{(K)}\right)$
- If $r < R$ then permute bits of $S_r^{(KS)} = s_1 \dots s_{LM}$:
  $S_r^{(KSP)} = s_{\pi_P(1)} s_{\pi_P(2)} s_{\pi_P(3)} \dots s_{\pi_P(LM)}$
- If $r = R$ then do the final key mixing:
  $S_r^{(KSP)} = S_r^{(KS)} \oplus K_{R+1}$
- Return state $S_r = S_r^{(KSP)}$

The ciphertext block is the last state: $C = S_R \in \{0,1\}^{LM}$.

E.g., suppose we have a cipher with 16 bits $(s_1 \cdots s_{16})$ split into 4 words of 4-bits $(w_1 = s_1 s_2 s_3 s_4, w_2 = s_5 s_6 s_7 s_8, \dots)$, with $R = 4$ rounds, the following S-BOX π_S on $\{0,1\}^4$:

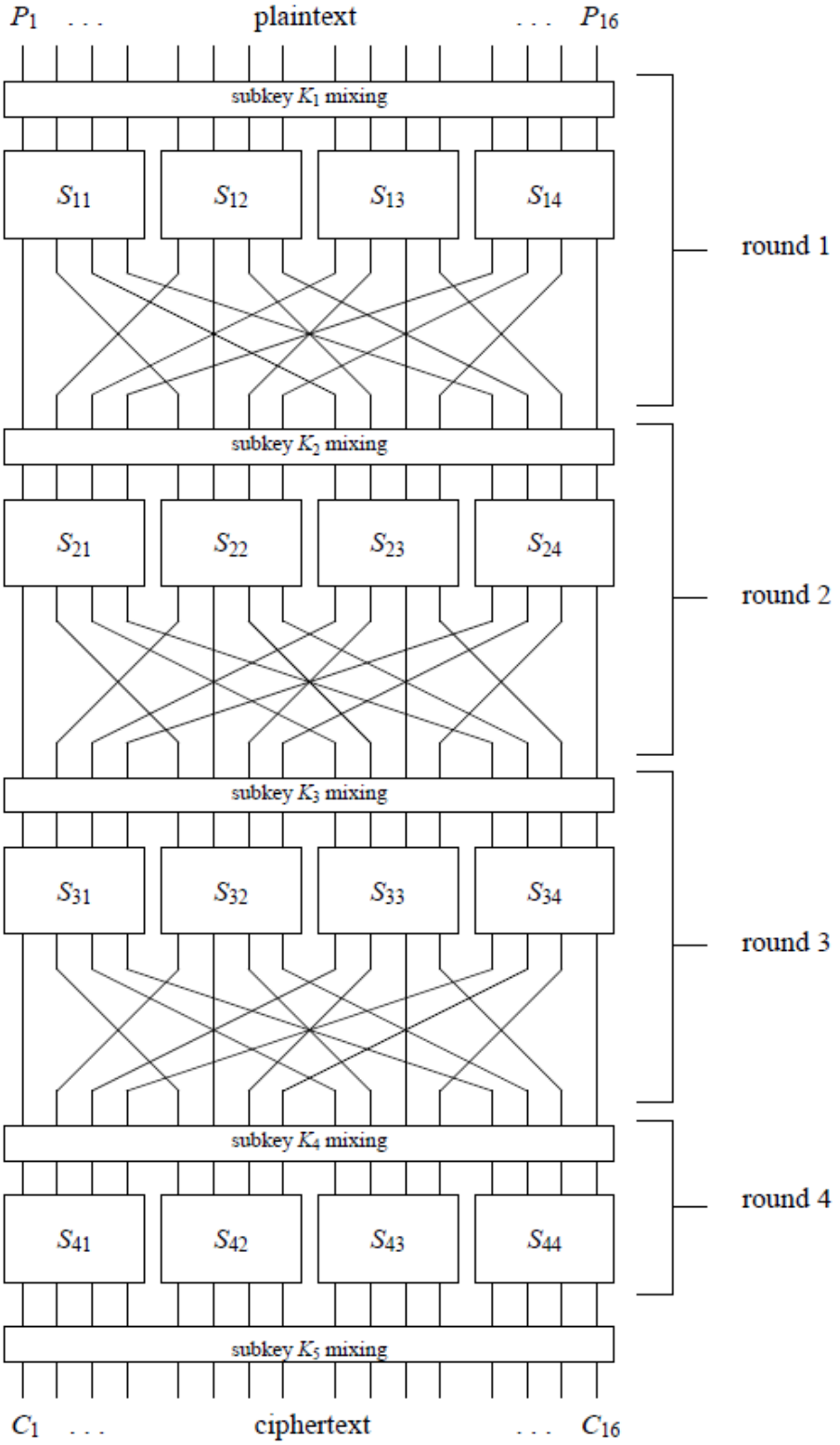
w	0	1	2	3	4	5	6	7
$\pi_S(w)$	14	4	13	1	2	15	11	8
w	8	9	10	11	12	13	14	15
$\pi_S(w)$	3	10	6	12	5	9	0	7

(4 bits are represented as integers, the most significant bit corresponds to the left most bit, e.g., bit s_1 in w_1)

and using the following bit-wise permutation π_P on $\{1, \dots, LM\}$:

i	1	2	3	4	5	6	7	8
$\pi_P(i)$	1	5	9	13	2	6	10	14
i	9	10	11	12	13	14	15	16
$\pi_P(i)$	3	7	11	15	4	8	12	16

The image below is a schematic description of this cipher:



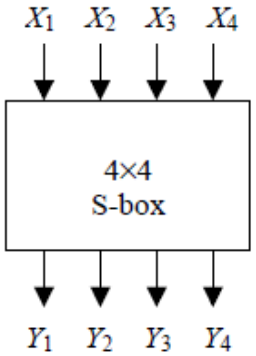
As you can see, SPN Ciphers are simple and can be very efficient, especially in hardware. Decryption is analogous to encryption, each operation is simply reversed.

8.2. Linear cryptanalysis

Linear cryptanalysis tries to linearly approximate the cipher, in particular that means a linear approximation of the S-BOXes as these are the only non-linear component.
The main attack idea is to find a linear relation between a subset of the plaintext bits and a subset of the bits before the last substitution that holds with a probability that has a significant *bias* away from 1/2.
As we show later on, the overall attack uses the probability bias of the linear relation to statistically distinguish when the relation holds (in order to derive information about a subset of the final round key bits).
Therefore the attack will require a lot of plaintext-ciphertext pairs in a known-plaintext scenario.
The bigger the magnitude of the bias, the fewer pairs are required.

8.3. S-BOX Linear Approximation Table (LAT)

Let X_1, X_2, X_3, X_4 be random variables for the input bits assumed to be independent and uniformly random and let Y_1, Y_2, Y_3, Y_4 be random variables for the output bits.



$X_1X_2X_3X_4$	0000	0001	0010	0011	0100	0101	0110	0111
$Y_1Y_2Y_3Y_4$	1110	0100	1101	0001	0010	1111	1011	1000
$X_1X_2X_3X_4$	1000	1001	1010	1011	1100	1101	1110	1111
$Y_1Y_2Y_3Y_4$	0011	1010	0110	1100	0101	1001	0000	0111

We are interested in linear relations for the SBOX, i.e., relations of the form

$$\sum_i a_i X_i + \sum_j b_j Y_j = c$$

In the current case of $\mathbb{F}_2$ (bits), $a_i, b_j, c \in \{0,1\}$.
There are $2^4 = 16$ different values for $X_1X_2X_3X_4$ all equally likely (by assumption).
Hence the probability that such a linear relation holds can be determined by counting the number of input-output pairs $(X_1X_2X_3X_4, Y_1Y_2Y_3Y_4)$ satisfy the relation, divided by 2^{16}.
E.g., consider the relations $X_2 + X_3 + Y_1 + Y_3 + Y_4 = 0$, $X_1 + X_4 + Y_2 = 0$ and $X_3 + X_4 + Y_1 + Y_4 = 0$:

$X_1X_2X_3X_4$	$Y_1Y_2Y_3Y_4$	$X_2 + X_3$	$Y_1 + Y_3 + Y_4$	$X_1 + X_4$	Y_2	$X_3 + X_4$	$Y_1 + Y_4$
0000	1110	0	0	0	1	0	1
0001	0100	0	0	1	1	1	0
0010	1101	1	0	0	1	1	0
0011	0001	1	1	1	0	0	1
0100	0010	1	1	0	0	0	0
0101	1111	1	1	1	1	1	0
0110	1011	0	1	0	0	1	0
0111	1000	0	1	1	0	0	1
1000	0011	0	0	1	0	0	1
1001	1010	0	0	0	0	1	1
1010	0110	1	1	1	1	1	0
1011	1100	1	1	0	1	0	1
1100	0101	1	1	1	1	0	1
1101	1001	1	0	0	0	1	0

1110	0000	0	0	1	0	1	0
1111	0111	0	0	0	1	0	1

Thus $\Pr[X_2 + X_3 + Y_1 + Y_3 + Y_4 = 0] = \frac{12}{16} = \frac{3}{4}$, $\Pr[X_1 + X_4 + Y_2 = 0] = \frac{8}{16} = \frac{1}{2}$ and $\Pr[X_3 + X_4 + Y_1 + Y_4 = 0] = \frac{2}{16} = \frac{1}{8}$.
For *independent* X_i and Y_j a linear relation has probability 1/2 to hold.
The stronger the probability bias away from 1/2 for a relation, the more useful the relation becomes.
We can describe the probability bias for all possible relations in the *Linear Approximation Table (LAT)*,
which is a table whose rows (respectively columns) describe the possible input (respectively output) variable sums.
The cell at row I and column O contains the number of matches between the sum of input bits and the sum of output bits:

$$\sum_{i \in I} X_i = \sum_{j \in J} Y_j$$

minus half the number of possible input values (for a 4-bit SBOX: $\frac{1}{2}2^4 = 8$):

Input Sum \ Output Sum	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	+8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	−2	−2	0	0	−2	+6	+2	+2	0	0	+2	+2	0	0
2	0	0	−2	−2	0	0	−2	−2	0	0	+2	+2	0	0	−6	+2
3	0	0	0	0	0	0	0	0	+2	−6	−2	−2	+2	+2	−2	−2
4	0	+2	0	−2	−2	−4	−2	0	0	−2	0	+2	+2	−4	+2	0
5	0	−2	−2	0	−2	0	+4	+2	−2	0	−4	+2	0	−2	−2	0
6	0	+2	−2	+4	+2	0	0	+2	0	−2	+2	+4	−2	0	0	−2
7	0	−2	0	+2	+2	−4	+2	0	−2	0	+2	0	+4	+2	0	+2
8	0	0	0	0	0	0	0	0	−2	+2	+2	−2	+2	−2	−2	−6
9	0	0	−2	−2	0	0	−2	−2	−4	0	−2	+2	0	+4	+2	−2
A	0	+4	−2	+2	−4	0	+2	−2	+2	+2	0	0	+2	+2	0	0
B	0	+4	0	−4	+4	0	+4	0	0	0	0	0	0	0	0	0
C	0	−2	+4	−2	−2	0	+2	0	+2	0	+2	+4	0	+2	0	−2
D	0	+2	+2	0	−2	+4	0	+2	−4	−2	+2	0	+2	0	0	+2
E	0	+2	+2	0	−2	−4	0	+2	−2	0	0	−2	−4	+2	−2	0
F	0	−2	−4	−2	−2	0	+2	0	0	−2	+4	−2	−2	0	+2	0

Linear approximation table: input and output variable sums are described in hexidecimal, e.g.,
$X_2 + X_3 = 0X_1 + 1X_2 + 1X_3 + 0X_4 \rightarrow 0110 \rightarrow 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 6$,
$Y_1 + Y_3 + Y_4 \rightarrow 11 = 0xB$.
Each table cell contains the count of inputs for which the corresponding linear approximation holds minus 8.
Note that the sum for each row and column is either +8 or -8.

Thus the probability bias for any relation can be found by looking up the corresponding number in the LAT and dividing by 16.
Such a linear approximation table can be easily computed using SAGE:

```
sage: S=mq.SBox(14,4,13,1,2,15,11,8,3,10,6,12,5,9,0,7);
sage: S.linear_approximation_matrix()
[ 8  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
[ 0  0 -2 -2  0  0 -2  6  2  2  0  0  2  2  0  0]
[ 0  0 -2 -2  0  0 -2 -2  0  0  2  2  0  0 -6  2]
[ 0  0  0  0  0  0  0  0  2 -6 -2 -2  2  2 -2 -2]
[ 0  2  0 -2 -2 -4 -2  0  0 -2  0  2  2 -4  2  0]
[ 0 -2 -2  0 -2  0  4  2 -2  0 -4  2  0 -2 -2  0]
[ 0  2 -2  4  2  0  0  2  0 -2  2  4 -2  0  0 -2]
[ 0 -2  0  2  2 -4  2  0 -2  0  2  0  4  2  0  2]
[ 0  0  0  0  0  0  0  0 -2  2  2 -2  2 -2 -2 -6]
[ 0  0 -2 -2  0  0 -2 -2 -4  0 -2  2  0  4  2 -2]
[ 0  4 -2  2 -4  0  2 -2  2  2  0  0  2  2  0  0]
[ 0  4  0 -4  4  0  4  0  0  0  0  0  0  0  0  0]
[ 0 -2  4 -2 -2  0  2  0  2  0  2  4  0  2  0 -2]
[ 0  2  2  0 -2  4  0  2 -4 -2  2  0  2  0  0  2]
[ 0  2  2  0 -2 -4  0  2 -2  0  0 -2 -4  2 -2  0]
[ 0 -2 -4 -2 -2  0  2  0  0 -2  4 -2 -2  0  2  0]
```

8.4. Piling-Up Lemma

In the attack we will combine several linear relations over individual Sboxes to obtain a linear relation over the first 3 rounds. To determine the probability bias for the resulting linear relation we can use the Piling-Up Lemma (see the original linear cryptanalysis paper by Matsui: http://link.springer.com/chapter/10.1007%2F3-540-48285-7_33).

Let X_1, X_2 be two independent binary random variables and $p_1 = \Pr[X_1 = 0], p_2 = \Pr[X_2 = 0]$, then

$$\Pr[X_1 \oplus X_2 = 0] = \Pr[X_1 = X_2] = \Pr[X_1 = 0, X_2 = 0] + \Pr[X_1 = 1, X_2 = 1] = p_1 p_2 + (1 - p_1)(1 - p_2)$$

Let ϵ_1, ϵ_2 be the probability biases of $X_1 = 0$ and $X_2 = 0$: $p_1 = \frac{1}{2} + \epsilon_1, p_2 = \frac{1}{2} + \epsilon_2$, then

$$\Pr[X_1 \oplus X_2 = 0] = \frac{1}{2} + 2\epsilon_1\epsilon_2$$

Hence, the probability bias of $X_1 \oplus X_2 = 0$ is $\epsilon_{12} = 2\epsilon_1\epsilon_2$.
This can be generalized to n independent binary random variables.

Piling-Up Lemma (Matsui)

For n independent binary random variables $X_1, \dots, X_n$, with probability biases $\epsilon_i = \Pr[X_i = 0] - \frac{1}{2}$:

$$\Pr[X_1 \oplus \cdots \oplus X_n = 0] = \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} \epsilon_i$$

Or equivalently,

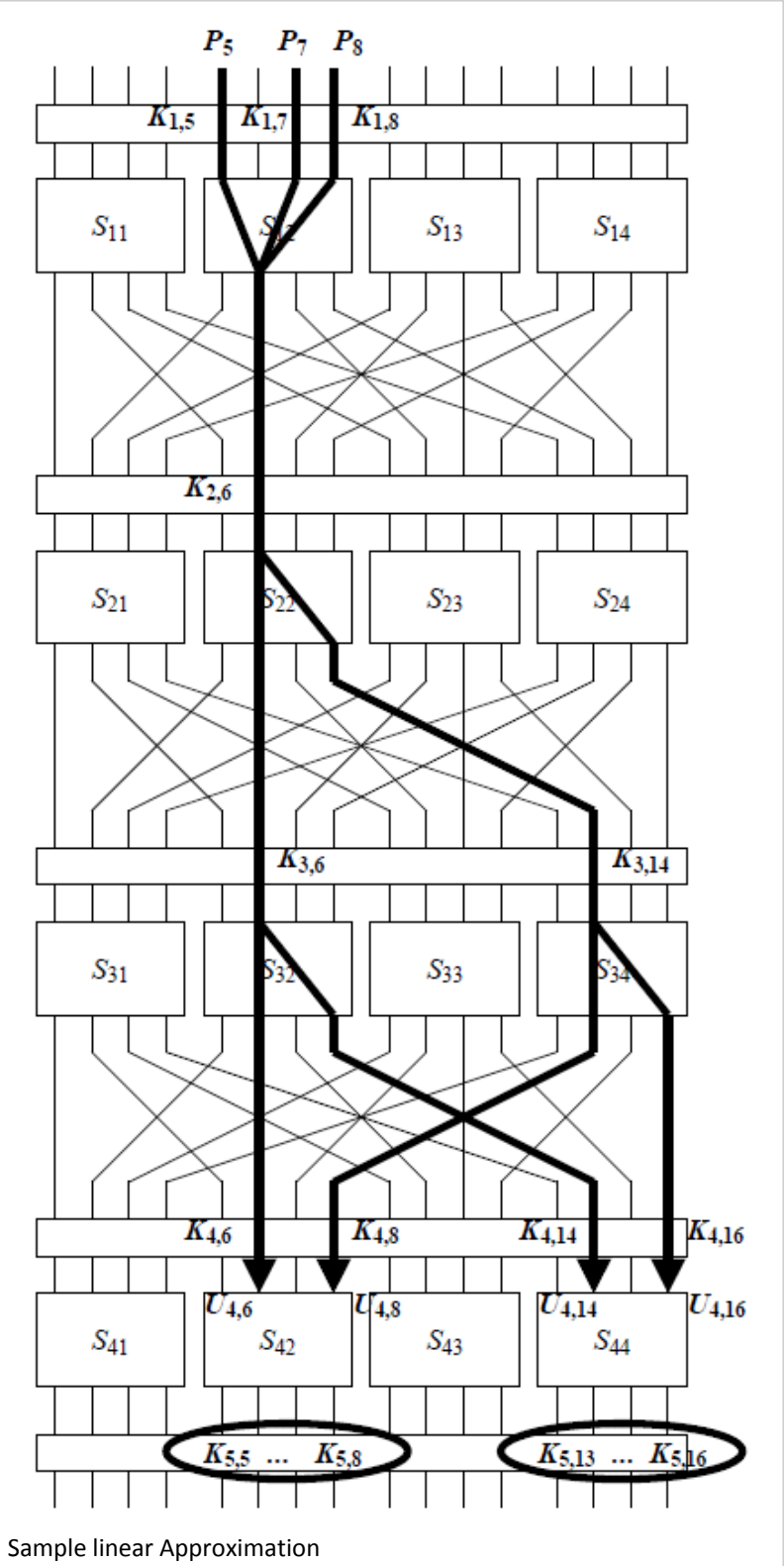
$$\epsilon_{1,\dots,n} = 2^{n-1} \prod_{i=1}^{n} \epsilon_i$$

Note that if $p_i = 0$ for all i, then $\Pr[X_1 \oplus \cdots \oplus X_n = 0]$ is either 0 or 1.
Also, if only one $p_i = \frac{1}{2}$ then $\Pr[X_1 \oplus \cdots X_n = 0] = \frac{1}{2}$.

8.5. Constructing a linear approximation over several rounds

Using the SBOX LAT and the piling-up lemma we will show an example how to use them to obtain a linear relation over the first three rounds of the toy cipher, i.e., only the first three key-mixings, substitutions and permutations, and not the last round key-mixing, substitution and final key-mixing.
We will use P_i the denote the i-th bit of the plaintext, $K_{r,i}$ to denote the i-th bit of the round key K_r, $X_{j,i}$ and $Y_{j,i}$ to denote the i-th input and output bit of Sbox $j \in \{11, \dots, 14, 21, \dots, 44\}$.

- We start with the linear approximation $X_{12,1} \oplus X_{12,3} \oplus X_{12,4} \oplus Y_{12,2} = 0$ over Sbox S_{12} with has probability bias $+\frac{1}{4}$.
  Note that $X_{12,1} = P_5 \oplus K_{1,5}, X_{12,3} = P_7 \oplus K_{1,7}$ and $X_{12,4} = P_8 \oplus K_{1,8}$.
  Although key-bits are unknown, for a given problem instance they're fixed. Therefore we gather sums of key-bits on the right-hand side:
  (1) $P_5 \oplus P_7 \oplus P_8 \oplus Y_{12,2} = K_{1,5} \oplus K_{1,7} \oplus K_{1,8}$ holds with bias $+1/4$.
- Output bit $Y_{12,2}$ is the 6th-bit and remains the 6th-bit through the permutation, key-mixing is applied and $Y_{12,2} \oplus K_{2,6} = X_{22,2}$ is the 2nd-bit input of Sbox S_{22}.
- We use the linear approximation $X_{22,2} \oplus Y_{22,2} \oplus Y_{22,4} = 0$ over Sbox S_{22} with bias $-1/4$ and obtain
  (2) $Y_{12,2} \oplus Y_{22,2} \oplus Y_{22,4} = K_{2,6}$ holds with bias $-1/4$.
- With the Piling-Up Lemma we can combine (1) and (2) to obtain
  (3) $P_5 \oplus P_7 \oplus P_8 \oplus Y_{22,2} \oplus Y_{22,4} = K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6}$ holds with bias $2\frac{-1}{4}\frac{1}{4} = -1/8$.
  Note that we make an assumption of independence here which isn't necessarily true.
  In truth we simply hope that in reality they behave close enough to independence such that the bias is very close to $-1/8$. In practice it turns out to work well enough.
- Output bits $Y_{22,2}$ and $Y_{22,4}$ pass through another permutation and key-mixing such that
  $Y_{22,2} \oplus K_{3.6} = X_{32,2}$ and $Y_{22,4} \oplus K_{3,14} = X_{34,2}$
- For both Sboxes S_{32} and S_{34} we use the same linear relation as for Sbox S_{22} with bias $-1/4$:
  $X_{32,2} \oplus Y_{32,2} \oplus Y_{32,4} = 0$ with bias $-1/4$
  $X_{34,2} \oplus Y_{34,2} \oplus Y_{34,4} = 0$ with bias $-1/4$
  Substituting the input bits with the previous output bits we get:
  (4) $Y_{22,2} \oplus Y_{32,2} \oplus Y_{32,4} = K_{3,6}$ with bias $-1/4$
  (5) $Y_{22,4} \oplus Y_{34,2} \oplus Y_{34,4} = K_{3,14}$ with bias $-1/4$
- Applying the Piling-Up Lemma we can combine the three linear relations (3), (4) and (5) to obtain
  $P_5 \oplus P_7 \oplus P_8 \oplus Y_{32,2} \oplus Y_{32,4} \oplus Y_{34,2} \oplus Y_{34,4} = K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6} \oplus K_{3,6} \oplus K_{3,14}$
  with bias $4\frac{-1}{8}\frac{-1}{4}\frac{-1}{4} = -1/32$



Sample linear Approximation

Note that the final bias of $-1/32$ depends on the biases of the invidual relations and the number of Sboxes actively involved in the relation.
To strengthen a cipher one can thus try to use Sboxes that have very small biases and a structure that tries to maximize the minimum number of active Sboxes.

8.6. Extracting final round key bits

Once we have a linear approximation over all but the last round for a given cipher that has a suitably large enough probability bias, we can try to exploit it to recover some bits of the final round key bits from the final key-mixing.
The linear approximation we obtained above

$$P_5 \oplus P_7 \oplus P_8 \oplus Y_{32,2} \oplus Y_{32,4} \oplus Y_{34,2} \oplus Y_{34,4} = K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6} \oplus K_{3,6} \oplus K_{3,14} \quad \text{with bias } -1/32$$

can be extended with $Y_{32,2} \oplus K_{4,6} = X_{42,2}, Y_{32,4} \oplus K_{4,8} = X_{42,4}, Y_{34,2} \oplus K_{4,14} = X_{44,2}, Y_{34,4} \oplus K_{4,16} = X_{44,4}$:

$$P_5 \oplus P_7 \oplus P_8 \oplus X_{42,2} \oplus X_{42,4} \oplus X_{44,2} \oplus X_{44,4} = K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6} \oplus K_{3,6} \oplus K_{3,14} \oplus K_{4,6} \oplus K_{4,8} \oplus K_{4,14} \oplus K_{4,16}$$

Let's replace the sum of key bits by zero:

$$P_5 \oplus P_7 \oplus P_8 \oplus X_{42,2} \oplus X_{42,4} \oplus X_{44,2} \oplus X_{44,4} = 0$$

This linear approximation holds with bias $-1/32$ if the sum of key bits equals zero
and with bias $1/32$ otherwise.
Note that only the sign of the bias changes, the magnitude remains the same.
In our attack the sign has no importance, only the magnitude is important to be able to statistically distinguish a correct guess.
Hence we can simplify to:

$$P_5 \oplus P_7 \oplus P_8 \oplus X_{42,2} \oplus X_{42,4} \oplus X_{44,2} \oplus X_{44,4} = 0 \quad \text{with bias} \pm 1/32$$

We will assume that we have access to a lot of (plaintext,ciphertext)-pairs obtained in a known-plaintext attack scenario.
To be able to check whether the linear relation holds we need to guess the key bits corresponding to the output bits of the two Sboxes S_{42} and S_{44}: $K_{5,5}, K_{5,6}, K_{5,7}, K_{5,8}, K_{5,13}, K_{5,14}, K_{5,15}, K_{5,16}$, we call these bits the final round *subkey*.
The idea is to guess the final round subkey bits and that we hope that for the correct guess the bias is visible, whereas for incorrect guesses we hope to see fractions very close to 1/2.

partial subkey $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$	| bias |	*partial subkey* $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$	| bias |
1 C	0.0031	2 A	0.0044
1 D	0.0078	2 B	0.0186
1 E	0.0071	2 C	0.0094
1 F	0.0170	2 D	0.0053
2 0	0.0025	2 E	0.0062
2 1	0.0220	2 F	0.0133
2 2	0.0211	3 0	0.0027
2 3	0.0064	3 1	0.0050
2 4	**0.0336**	3 2	0.0075
2 5	0.0106	3 3	0.0162
2 6	0.0096	3 4	0.0218
2 7	0.0074	3 5	0.0052
2 8	0.0224	3 6	0.0056
2 9	0.0054	3 7	0.0048

Experimental results for a linear attack with 10000 (plaintext,ciphertext)-pairs.
It shows partial subkey guesses and the determined $|bias| = |count - 5000|/10000$,
subkeys are written down in hexidecimal.
The correct subkey 24 is listed in bold and has the highest bias magnitude not only in the showed listing, but among all guesses for the subkey.

Matsui also showed in his paper that the number of required (plaintext,ciphertext)-pairs for the attack is proportional to ϵ^{-2}, where ϵ is the probability bias of the linear relation over $R - 1$ rounds.
In practice, it is generally reasonable to use a small multiple of ϵ^{-2} pairs.

Once we have determined the correct value for the final round subkey bits (or at least a short list of highly-likely values that we can go through), we can continue in the following manner:

1. Try to exploit other linear relations in order to obtain all final round key bits,
   guess all remaining unknown final round key bits.
2. Strip the last round of all known (plaintext,ciphertext)-pairs,
   i.e., revert the last cipheroperations till the second-last key-mixing.
3. One can view the cipher as having $R - 1$ rounds, continue to attack the cipher using a linear relation over $R - 2$ rounds.
4. Iterate 1-3 until all round keys have been broken.

To experiment with linear cryptanalysis on this toy cipher one can use SAGE and the sage files found on the course website.
Install SAGE from http://www.sagemath.org/doc/installation/ or use SAGE in the cloud https://cloud.sagemath.com/.
(The files are also available at cloud.sagemath.com:
https://cloud.sagemath.com/projects/c682ad80-f351-4902-b096-aa2c2ae21613/files/)

The files ptctlist.txt and ptctlist.sobj (SAGE binary format) contain a plaintext-ciphertext list generated with:

```
sage: load('toycipher_definition.sage')
sage: load('toycipher_gendata')
```

Be careful: executing this yourself will overwrite the file ptctlist.sobj.

To execute a linear cryptanalysis attack using the above linear relation to recover bits 5,6,7,8,13,14,15,16 from final round key K5 do the following in sage with access to the *.sage and ptctlist.sobj files:

```
sage: load('toycipher_definition.sage')
sage: load('linearcryptanalysis.sage')
sage: result
[[0,208]
...
,[192,236]
,[237,60]]
sage: int2K5sub(60)
[0,0,0,0,0,0,1,1,0,0,0,0,1,1,0,0]
```

Read the comments in the *.sage files to see what's exactly happening.
'result' is a list of (|bias|, k5sub) sorted for increasing bias.
The function int2K5sub maps an 8-bit integer to the corresponding guess of bits 5,6,7,8,13,14,15,16 of K5, other bits of K5 are set to 0.
In this case the largest bias indeed belongs to the correct K5 sub key, this may not always be the case.