Linear and Differential Cryptanalysis

Benjamin Toft Jakobsen* Mehdi Abyar† Peter Sebastian Nordholt‡
20022151 20023623 20022601

University of Aarhus, Denmark

December 15, 2006

*bentoft@daimi.au.dk
†mehdi@daimi.au.dk
‡sebastian@daimi.au.dk

Contents

1 Introduction

The process of studying methods of encryption to obtain information from encrypted data without knowing the *secret key* is called cryptanalysis. It is usually a deep analysis and attacking of an encryption method to find the secret key.

In this report we are going to introduce two cryptanalysis techniques, namely *Linear* and *Differential* cryptanalysis. In the first section we describe linear cryptanalysis, with a small example *Substitution Permutation Network* (SPN), in section 3 we explain the differential cryptanalysis and the idea of extracting key bits of a SPN by looking at the differences between input and output pairs. In section 4 we explain how the linear cryptanalysis can be applied to the Data Encryption Standard (DES). First by a small 3 rounds DES and then the full 16 rounds DES. In section 5 we describe what can be done to prevent linear and differential cryptanalysis attacks and make the SPN resistant.

2 Linear Cryptanalysis

A linear cryptanalysis is a known plain text attack, against a block cipher. The attack was first described by Matsui in 1994 as an attack against DES [M93]. But linear cryptanalysis can be used against many other block ciphers, and must therefore be considered when designing new block ciphers.

In this section, we will describe the steps of the general linear cryptanalysis attack. But first we will introduce an SPN cipher which will be used as an example cipher.

2.1 Substitution permutation network

We will now describe the general substitution permutation network (SPN), and introduce an example SPN that will be used in this section and in section 3.

A SPN is a block cipher that consist of a number of rounds, each containing a substitution and a permutation. The key will normally be expanded into round keys, that will be XOR'ed at the beginning of each round.

Our small example SPN[1] takes as input 16 bits, and outputs 16 bits. The key is also 16 bits, but they are expanded into 5 round keys of 16 bits. In each round the 16 bits are XOR'ed with the round key, then they are split into four blocks of four bit and each block goes through a substitution box (S-box). At the end of each round, the 16 bits are permuted. In our case we use the same S-box for all blocks and all rounds. The permutation can be seen in figure 1, which shows all of the example cipher. Our concrete S-box can bee seen in figure 2.

2.2 Overview of the attack

A linear cryptanalysis can be split into two steps. First you have to find a linear approximation for the SPN you want to break. The idea is to find an approximation that only contains plaintext bits and input bits to the last round. After we have found an approximation, we are ready to make the actual attack. Notice that you don't need any ciphertext or plaintext to do the first step. It is only an analysis of the SPN, and therefore independent of the key you want to find. This means that you only have to make the analysis once for a certain SPN.

In the actual attack we try to find some of the bits of the last round key. The idea is to calculate the input bits to the last round, that are in our approximation. We

[1] Borrowed from [H02]

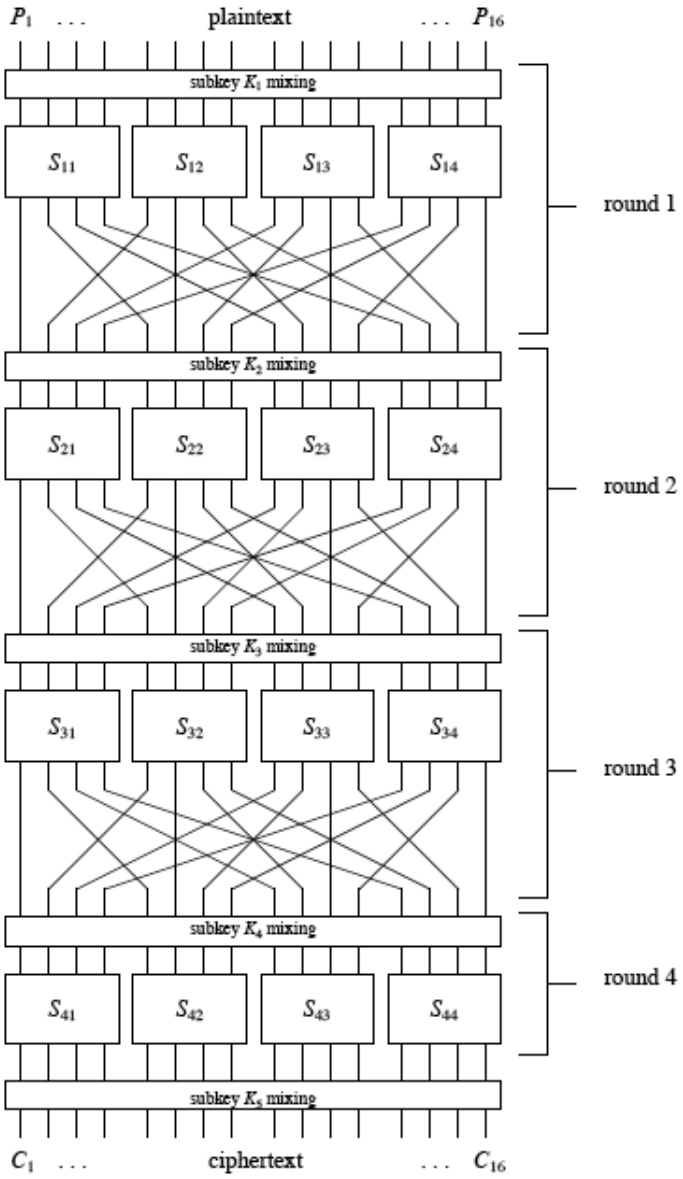

Figure 1: An overview of our example SPN.

do that by inverting the last round, hopefully using only a subset of the round key bits. This can be done since we only need some of the input bits to the last round. Now we try all possible combinations of these key bits, and checks for what key bits the approximation is true most often, over all our plaintext/ciphertext pairs. This combination must be the right one since we assume that for wrong choices of key bits, the result will be random.

The linear approximation we are looking for in step one, is an equation of the form:

$$P^* \oplus X^* = 0$$

where P^* is an XOR combination of some plaintext bits and the X^* is an XOR combination of some input bits to the last round. It is clear that if there exists a

X_1	X_2	X_3	X_4	Y_1	Y_2	Y_3	Y_4
0	0	0	0	1	1	1	0
0	0	0	1	0	1	0	0
0	0	1	0	1	1	0	1
0	0	1	1	0	0	0	1
0	1	0	0	0	0	1	0
0	1	0	1	1	1	1	1
0	1	1	0	1	0	1	1
0	1	1	1	1	0	0	0
1	0	0	0	0	0	1	1
1	0	0	1	1	0	1	0
1	0	1	0	0	1	1	0
1	0	1	1	1	1	0	0
1	1	0	0	0	1	0	1
1	1	0	1	1	0	0	0
1	1	1	0	0	0	0	0
1	1	1	1	0	1	1	1

Figure 2: Our example S-box. The X_i's are input to the S-box and the Y_i's are output.

linear equation, that always is true, then the block cipher is not good, since non linearity is very important for an SPN. It is also clear that if the bits in the equation were all uniformly random then the probability that the equation would come true is 0.5. So what we are looking for, is an equation where the probability that it will be true is as far from 0.5 as possible.

2.3 Finding a linear approximation

In this subsection we will try to explain how to find a good linear approximation. To do so we have to introduce some notation.

Definition 2.1. If L is a random variable of the form $X_1 \oplus X_2 \oplus ... \oplus X_n$ with probability P_L of being 0, then we define the bias of the random variable to be

$$\varepsilon_L = P_L - \frac{1}{2}$$

Where ε_L is the bias of L.

This means that if X is a uniformly random variable, then the bias of X is 0. The Piling-Up Lemma [M93], tell us that the bias of a random variable $X_1 \oplus X_2 \oplus ... \oplus X_n$, where the $X_i's$ are independent, is equal

$$2^{n-1} \prod_{i=1}^{n} \varepsilon_i$$

where ε_i is the bias of X_i. We will use this lemma, even though the X_i's are not independent. This however turns out to be a good approximation in practice.

Now we are ready to describe how to find a linear approximation. The only nonlinear part of our example block cipher is the S-box. This is also the case in most other SPN's like DES. So the first thing to do, is to analyze the S-boxes of the cipher. In our example cipher we only have one S-box, while an analysis of DES would require an analysis of each of the eight different S-boxes.

2.3.1 Linear Approximation for a single S-box

We want to find a linear equations in the input and output bits of the S-box. In our case there are 8 variables, namely $X_1, ..., X_4, Y_1, ..., Y_4$. We can then construct a Linear Approximation Table with the bias of all possible combinations of the variables. In our case there will be $2^8 = 256$ entries in the table. We will write this table as a 16×16 table with the 16 combinations of input as rows and the 16 combinations of output as columns. We will represent the input combination $X_1 \oplus X_3$ as the binary number 1010, and write it as a decimal: 10. We will represent the outputs in the same way. The table can be seen in figure 3. The bias values are represented as integers that should be divided by 16 to get the actual bias.

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
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
10	0	4	-2	2	-4	0	2	-2	2	2	0	0	2	2	0	0
11	0	4	0	-4	4	0	4	0	0	0	0	0	0	0	0	0
12	0	-2	4	-2	-2	0	2	0	2	0	2	4	0	2	0	-2
13	0	2	2	0	-2	4	0	2	-4	-2	2	0	2	0	0	2
14	0	2	2	0	-2	-4	0	2	-2	0	0	-2	-4	2	-2	0
15	0	-2	-4	-2	-2	0	2	0	0	-2	4	-2	-2	0	2	0

Figure 3: Linear Approximation table. The rows are S-box input and the columns are output.

When we have made the Linear Approximation tables for all S-boxes used in the cipher, we are ready to find the linear approximation for the whole cipher.

2.3.2 Linear Approximation for the complete cipher

I will start this subsection by introducing some notation[2]:

X_i	The i'th input bit to the S-box.
Y_i	The i'th output of the S-box.
P_i	The i'th bit of the plaintext.
C_i	The i'th bit of the ciphertext.
U_i^j	The i'th bit of the input to the S-boxes in the j'th round.
V_i^j	The i'th bit of the output of the S-boxes in the j'th round.
K_i^j	The i'th bit of key in the j'th round.
$A[a_1, a_2, ..., a_n]$	The XOR between bit $a_1 - a_n$, like $U^3[1, 5, 8] = U_1^3 \oplus U_5^3 \oplus U_8^3$.

To get an overview of how we construct the linear approximation, see figure 4. The first thing to do is to find a good linear approximation for an S-box in the first round. We choose S-box S_{12} (see Figure 1), and the equation $X[1, 3, 4] \oplus Y_2 = 0$, since the corresponding entry (11,4) in the Linear Approximation table has a high

[2]The notation is borrowed from [M93]

value (in our case 4, which means a bias at 0.25). Notice that when this equation is applied to S-box S_{12}, we get the equation

$$U^1[5,7,8] \oplus V_6^1 = 0$$

But $U^1[5,7,8] = P[5,7,8] \oplus K^1[5,7,8]$, and we get the equation

$$P[5,7,8] \oplus V_6^1 \oplus K^1[5,7,8] = 0 \tag{1}$$

where the only unknown besides the key bits is V_6^1. So now we will trace this bit through the cipher. Remember that the left side of the equation still have the same bias, namely 0.25.

We follow the unknown bit V_6^1 through the permutation. In figure 4 we see that $V_6^1 = K_6^2 \oplus U_6^2$, so now we need a linear approximation for S-box S_{22}, containing only one input bit, namely U_6^2 which is the second input to S_{22}, X_2. When looking at the Linear Approximation table we find that $X_2 \oplus Y[2,4] = 0$ has a good bias (-0.25). Output 2 and 4 for S-box S_{22} are the bits V_6^2 and V_8^2, and we get the equation

$$\begin{aligned} U_6^2 \oplus V^2[6,8] &= 0 \\ \Updownarrow & \\ V_6^1 \oplus K_6^2 \oplus V^2[6,8] &= 0 \end{aligned} \tag{2}$$

where the left side still have the bias -0.25.

Again we follow the bits $V^2[6,8]$ through the permutation, and find that $V^2[6,8] = K^3[6,14] \oplus U^3[6,14]$. U_6^3 and U_{14}^3 are both send as the second input bit, X_2, for the S-boxes S_{32} and S_{34} respectively. This means that we can use the same linear approximation for both S_{32} and S_{34} as we just did for S_{22}, namely $X_2 \oplus Y[2,4] = 0$. Output 2 and 4 for S-box S_{32} are the bits V_6^3 and V_8^3, and for S-box S_{34} they are V_{14}^3 and V_{16}^3. This gives us that

$$\begin{aligned} U_6^3 \oplus V^3[6,8] &= 0 \\ \Updownarrow & \\ V_6^2 \oplus K_6^3 \oplus V^3[6,8] &= 0 \end{aligned} \tag{3}$$

and

$$\begin{aligned} U_{14}^3 \oplus V^3[14,16] &= 0 \\ \Updownarrow & \\ V_{14}^2 \oplus K_{14}^3 \oplus V^3[14,16] &= 0 \end{aligned} \tag{4}$$

where the left side of the equations both have bias -0.25 again.

Now we can combine equation (1)-(4), by taking the XOR of all left sides and all the right sides. By eliminating duplicated bits, we get the equation

$$P[5,7,8] \oplus V^3[6,8,14,16] \oplus K^1[5,7,8] \oplus K_6^2 \oplus K_[^3 6,14] = 0 \tag{5}$$

Once more we follow the bits V_6^3, V_8^3, V_{14}^3 and V_{16}^3 through the permutation and find that $V^3[6,8,14,16] = U^4[6,8,14,16] \oplus K^4[6,8,14,16]$. If we insert that into equation (5), we finally get the linear approximation we are looking for

$$P[5,7,8] \oplus U^4[6,8,14,16] \oplus \Sigma_K = 0$$

where Σ_K is an XOR of some key bits. Remember that the left side, is the XOR of four random variables, with bias 0.25, -0.25, -0.25 and -0.25 respectively. Now the Piling Up lemma tells that the bias of such an XOR is

$$2^3 \cdot \frac{4}{16} \cdot (-\frac{4}{16})^3 = -\frac{1}{32}$$

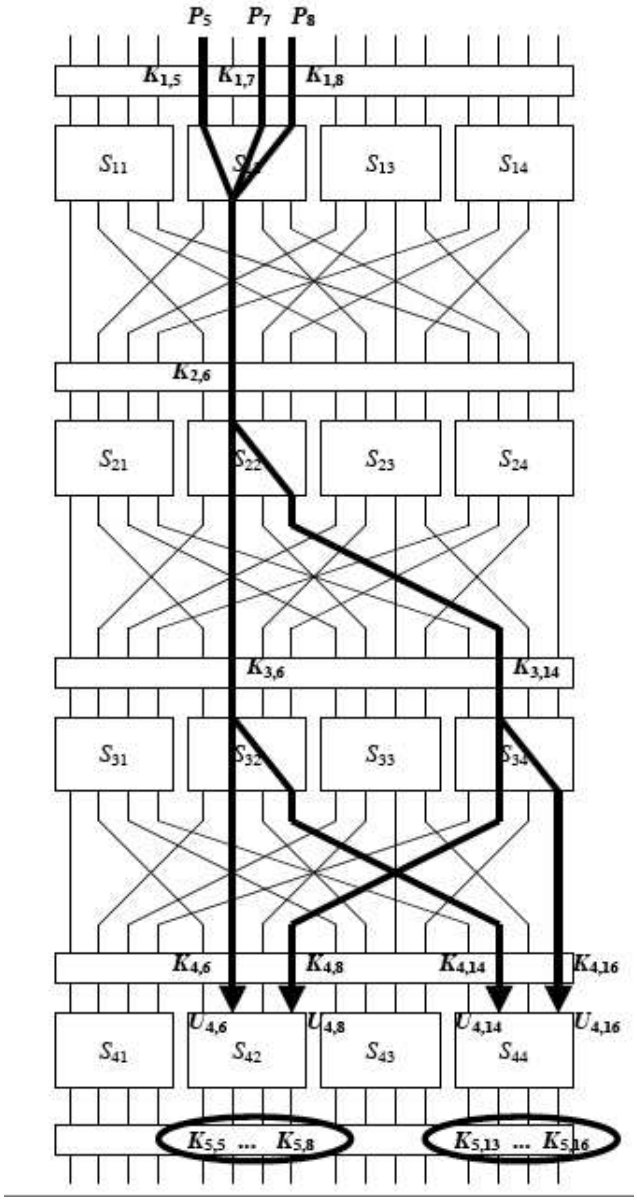

Figure 4: A path through our example SPN.

We don not know the key bits, but we know they are fixed, meaning that the XOR Σ_K is equal to 0 or 1. Therefore the bias of the random variable

$$P[5, 7, 8] \oplus U^4[6, 8, 14, 16]$$

will be either $-\frac{1}{32}$ or $\frac{1}{32}$. In other words we could say that the equation

$$P[5, 7, 8] \oplus U^4[6, 8, 14, 16] = 0$$

will be true with probability $\frac{15}{32}$ or $\frac{17}{32}$. So now we have found a linear approximation, and we shall see how to use that in the next subsection.

2.4 The actual attack

Now that we have found a Linear Approximation, we are ready to do the actual attack. We have to invert the last round to check if the equation is true or false. In our case we have to XOR the cipher text bits with the last round key, and then send the bits through the inverted S-box. Then we can check if the equation

$$P[5,7,8] \oplus U^4[6,8,14,16] = 0$$

holds true or not.

Notice that to check the equation we only need the bits $U_6^4, U_8^4, U_{14}^4, U_{16}^4$, but to get them we need all the bits V_5^4, ... , V_8^4 and V_{13}^4, ... , V_{16}^4 because of the S-box. But that is also fine, since we only needs 8 keys bits from the last round key to find those. This means that we should only test $2^8 = 256$ possible keys instead of $2^{16} = 65536$, which is a huge difference. This also means that we can just do exhaustive search.

Let $K_{<1>}$ and $K_{<2>}$ be two 4 bit key blocks, and let $C_{<5-8>}$ and $C_{<13-16>}$ be the cipher text bits from 5 to 8 and from 13 to 16 of the ciphertext C. Let S^{-1} be the inverted S-box and let Γ be the set of all ciphertext/plaintext pairs we have. The algorithm for the attack will then be:

T[16][16] $\leftarrow 0$
for all $K_{<1>}$, $K_{<2>}$ **do**
  for all plain text / cipher text pairs $(P, C) \in \Gamma$ **do**
    $V_{<5-8>} \leftarrow C_{<5-8>} \oplus K_{<1>}$
    $V_{<13-16>} \leftarrow C_{<13-16>} \oplus K_{<2>}$
    $U_{<5-8>} \leftarrow S^{-1}(V_{<5-8>})$
    $U_{<13-16>} \leftarrow S^{-1}(V_{<13-16>})$
    if $U_{4,6} \oplus U_{4,8} \oplus U_{4,14} \oplus U_{4,16} \oplus P_5 \oplus P_7 \oplus P_8 = 0$ **then**
      T[$K_{<1>}$][$K_{<2>}$] $\leftarrow$ T[$K_{<1>}$][$K_{<2>}$] + 1
    end if
  end for
end for
max $\leftarrow 0$
for all $K_{<1>}$, $K_{<2>}$ **do**
  if |size of(Γ)/2 - T[$K_{<1>}$][$K_{<2>}$]| > max **then**
    max $\leftarrow$ |size of(Γ)/2 - T[$K_{<1>}$][$K_{<2>}$]|
    result $\leftarrow K_{<1>}$ || $K_{<2>}$
  end if
end for
return result

In other words, check for what key bits, the number of times the equation holds, is most far from half the times. Those key bits should be the right ones. So now we have hopefully found 8 bits of the last round key.

2.5 Conclusion on Linear Cryptanalysis

One might say that 8 bits of the last round key is not very useful. But if we got half of the bits, we can easily find the rest by exhaustive key search. If we got all of the last round key, we might have all of the key, depending on how the key expanding is done. In all cases there will be only few keys with the same last round key, and therefore we would be able to find it.

Another question is how many plaintext/ciphertext pairs do we need to make this attack? In [S06] the author claims that we would need about 8000 pairs to make the attack on our example cipher. For a full 16-rounds DES one would need

2^{47} pairs [M94]. Could it ever happen that an adversary got access to such data? That is hard to imagine, but never say never. The fact is that we can no longer achieve a chosen cipher text secure cryptographic system as defined in [D04], if a Linear Cryptanalysis attack is possible.

In all cases this attack is one of the reasons, that one should never use the same key too much. The attack also causes several considerations on designs of new block ciphers (see section 5).

3 Differential Cryptanalysis

Differential cryptanalysis is a so called chosen plaintext attack, that is to say the attacker has the possibility to choose a plaintext and get the corresponding ciphertext [S06]. The main idea in differential cryptanalysis is comparing the XOR of two plaintexts with the XOR of corresponding two ciphertexts. In differential cryptanalysis the input and output difference of the S-boxes are considered to determine a high probability difference pair[3], which leads to some information about the plaintext difference and the difference of the input to the last round [H02].

Considering the SPN described in section 2 we try to explain the differential cryptanalysis in three steps , first analyzing the cipher pairs, secondly using cipher analysis to construct differential characteristics and lastly making out how the key bits can be extracted.

3.1 Difference Distributions

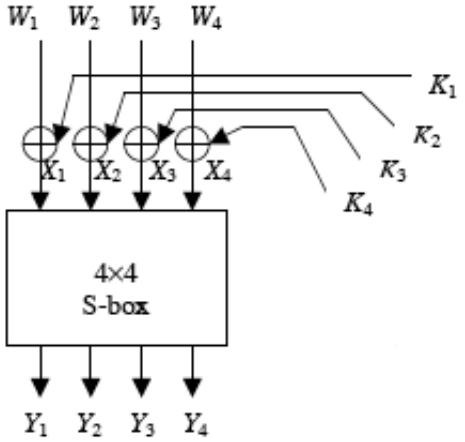

Figure 5: S-box.

Consider the S-box from figure 5 with input $X = [X_1X_2X_3X_4]$ and output $Y = [Y_1Y_2Y_3Y_4]$. We define $\Delta X = X' \oplus X''$ to be the input difference and $\Delta Y = Y' \oplus Y''$ to be the corresponding output difference. In the following the pair $(\Delta X, \Delta Y)$ is referred to as a differential pair. We can explore the probability of a ΔY given ΔX for all S-boxes considering all different input pairs (X', X'') where $\Delta X = X' \oplus X''$. Remember the S-box introduced in section 2, figure 6 lists all the possible output differences ΔY given that $\Delta X \in \{1011, 1000, 0100\}$, for example if $\Delta X = 1011$ and

[3] A high probability difference pair is, the difference of an input and output that often occurs

$X' = 0000$ then $Y' = 1110$, thus

$$X'' = \Delta X \oplus X' = 1011 \oplus 0000 = 1011$$

then $Y'' = 1100$ and $\Delta Y = 0010$ comes from figure 6.

X'	**Y'**	ΔY when $\Delta X = 1011$	ΔY when $\Delta X = 1000$	ΔY when $\Delta X = 0100$
0000	1110	0010	1101	1100
0001	0100	0010	1110	1011
0010	1101	0111	0101	0110
0011	0001	0010	1011	1001
0100	0010	0101	0111	1100
0101	1111	1111	0110	1011
0110	1011	0010	1011	0110
0111	1000	1101	1111	1001
1000	0011	0010	1101	0110
1001	1010	0111	1110	0011
1010	0110	0010	0101	0110
1011	1100	0010	1011	1011
1100	0101	1101	0111	0110
1101	1001	0010	0110	0011
1110	0000	1111	1011	0110
1111	0111	0101	1111	1011

Figure 6: Sample difference pairs of the S-box.

From figure 6 we can see that the number of occurrence of $\Delta Y = 0010$ given $\Delta X = 1011$ is 8 out of 16. In the same way we can compute the probability of the other differentials, that is the number of occurrence of a differential pair $(\Delta X, \Delta Y)$ divided by 16, see figure 7. Each element in the table of figure 7 presents the number of occurrence of an output difference ΔY given an input difference ΔX[4].

Before we can describe the attack we have to discuss the influence of the key on the S-box differential (considering figure 5). The input to a S-box is represented by X, the output by Y, and let $W = [W_1 W_2 W_3 W_4]$ denote the input to round keys, where by input the round key W_i we mean the output from the previous round before being XOR 'ed with the round key, W_i see figure 2. So the input difference to a round key is

$$\Delta W = [W_1' \oplus W_1'' \, W_2' \oplus W_2'' \, W_3' \oplus W_3'' \, W_4' \oplus W_4'']$$

where $W' = [W_1' W_2' W_3' W_4']$ and $W'' = [W_1'' W_2'' W_3'' W_4'']$.
Lets now take a look at input difference to the round keys, since the key is the same for both inputs we have

$$\Delta W_i = W_i' \oplus W_i'' = (X_i' \oplus K_i) \oplus (X_i'' \oplus K_i) = X_i' \oplus X_i'' = \Delta X_i$$

since $K_i \oplus K_i = 0$. Thus the input difference of the input to the round key is the same as input difference to the S-box, this is to say that the key bits do not have any influence on the input difference.

[4]It is worth to mention that the S-box could be ideal if the probability distribution for occurring a particular ΔY given ΔX was $\frac{1}{16}$, however it can be proved that this is not possible [H02].

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	2	0	0	0	2	0	2	4	0	4	2	0	0
2	0	0	0	2	0	6	2	2	0	2	0	0	0	0	2	0
3	0	0	2	0	2	0	0	0	0	4	2	0	2	0	0	4
4	0	0	0	2	0	0	6	0	0	2	0	4	2	0	0	0
5	0	4	0	0	0	2	2	0	0	0	4	0	2	0	0	2
6	0	0	0	4	0	4	0	0	0	0	0	0	2	2	2	2
7	0	0	2	2	2	0	2	0	0	2	2	0	0	0	0	4
8	0	0	0	0	0	0	2	2	0	0	0	4	0	4	2	2
9	0	2	0	0	2	0	0	4	2	0	2	2	2	0	0	0
10	0	2	2	0	0	0	0	0	6	0	0	2	0	0	4	0
11	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
12	0	2	0	0	2	2	2	0	0	0	0	2	0	6	0	0
13	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
14	0	0	2	4	2	0	0	0	6	0	0	0	0	0	2	0
15	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

Figure 7: Difference distribution.

3.2 Differential Characteristic

After handling the differential data for S-boxes in an SPN, we have enough information to find useful differential characteristic of the overall cipher. The idea is to attack the cipher by finding a subset key bits following the last round. This is done by constructing a differential characteristic of certain S-box difference pairs in each round, such that a differential involves plaintext bits and input data to the last round of S-boxes.

Consider the SPN illustrated in figure 8. In this diagram the active S-boxes S_{12}, S_{23}, S_{32}, and S_{33} have non-zero differentials. This forms a characteristic for the first three rounds of the cipher. The difference pairs of the active S-boxes are

$$S_{12} : \Delta X = 11 \equiv 1011 \rightarrow \Delta Y = 2 \equiv 0010 \qquad \text{with probability } \frac{8}{16}$$

$$S_{23} : \Delta X = 4 \equiv 0100 \rightarrow \Delta Y = 6 \equiv 0110 \qquad \text{with probability } \frac{6}{16}$$

$$S_{32} : \Delta X = 2 \equiv 0010 \rightarrow \Delta Y = 5 \equiv 0101 \qquad \text{with probability } \frac{6}{16}$$

$$S_{33} : \Delta X = 2 \equiv 0010 \rightarrow \Delta Y = 5 \equiv 0101 \qquad \text{with probability } \frac{6}{16}$$

which can be verified from the table of figure 7. The other S-boxes have input difference zero.
In the following we use U_i to denote the input to the i'th round S-box and V_i to denote the output of the i'th round S-box. The input difference to the first round is given by

$$\Delta U_1 = [0000\,1011\,0000\,0000]$$

and the corresponding output difference is

$$\Delta V_1 = [0000\,0010\,0000\,0000]$$

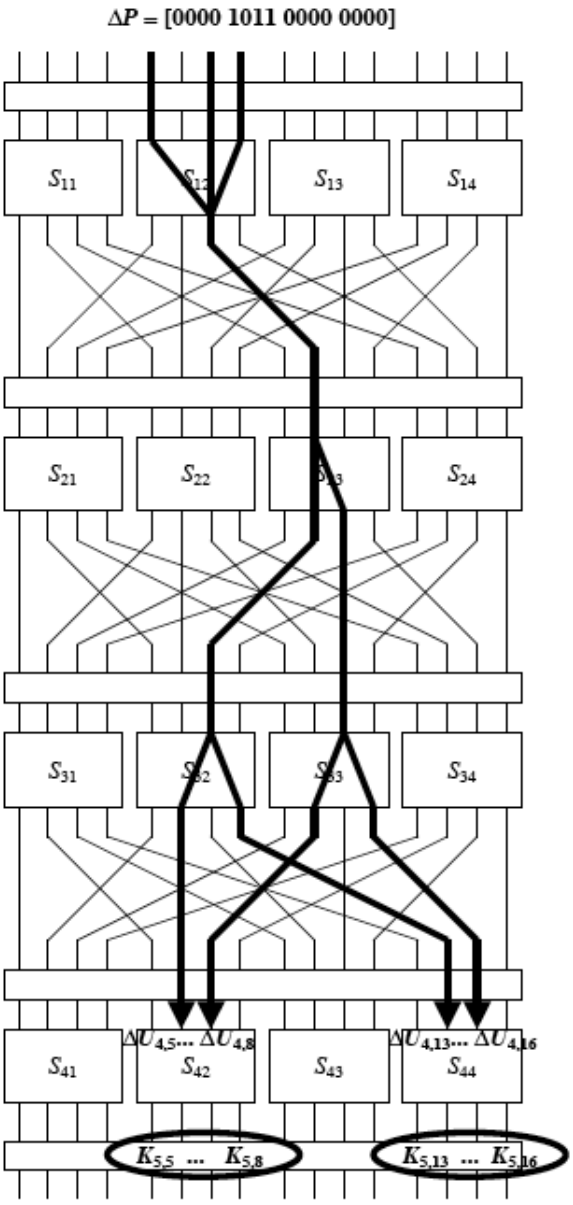

Figure 8: Differential Characteristic with ΔP as the input difference

By looking at difference pair for S_{12} and the following round permutation, we realize that the input difference to the second round S-box is

$$\Delta U_2 = [0000\,0000\,0100\,0000]$$

with probability $\frac{1}{2}$.
The second round differential with respect to difference pair for S_{23} is

$$\Delta V_2 = [0000\,0000\,0110\,0000]$$

and the permutation of round two gives

$$\Delta U_3 = [0000\,0010\,0010\,0000]$$

with probability $\frac{6}{16}$. In the same way the output of the third round is

$$\Delta V_3 = [0000\,0101\,0101\,0000]$$

and the corresponding input to round four is

$$\Delta U_4 = [0000\,0110\,0000\,0110]$$

In other words

$$\Delta U_1 = [0000\,1011\,0000\,0000] \longrightarrow \Delta V_3 = [0000\,0101\,0101\,0000]$$

and

$$\Delta V_3 = [0000\,0101\,0101\,0000] \longrightarrow \Delta U_4 = [0000\,0110\,0000\,0110]$$

Hence it follow that

$$\Delta U_1 = [0000\,1011\,0000\,0000] \longrightarrow \Delta U_4 = [0000\,0110\,0000\,0110]$$

The distribution of differentials illustrated in figure 7 can interpret as independent conditional probabilities (which mathematically is not true, but we assume that even so because practically it works)

$$Pr[\Delta Y|\Delta X] = (\Delta X, \Delta Y)$$

considering this and the computation above, differentials can be combined. For example in this case we have

$$(0000\,1011\,0000\,0000, 0000\,0110\,0000\,0110) = \frac{8}{16} \cdot \frac{6}{16} \cdot \frac{6}{16} \cdot \frac{6}{16} = \frac{27}{1024}$$

which means that with probability $\frac{27}{1024}$ the differential characteristic will occur. Now we are ready to give the idea of how to extract the key bits.

3.3 Key Bit Extraction

Now after having discussed the differential characteristic of rounds it is time to attack the cipher and obtain key bits. In a cipher of N rounds the idea is to attack the $N-1$'th round cipher. In the example of SPN network it is possible to extract bits from subkey K_5 by attacking the fourth round key. The attack involves pair vise decryption of the last round cipher and testing the corresponding input to find out that the *right/good* pair is occurred, where meaning by *good/right* pairs is the pairs that result in none zero input differential to the expected S-boxes i.e. the pairs for which the differential characteristics holds.

Lets look at the decryption of the last round, it involves the XOR of the cipher with the subkey, which is influenced by the non-zero differentials, and running data backward through the S-boxes, where all possible values of the subkey should be tried. Put in a different way we XOR the output cipher with every possible candidate subkey and apply the inverse S-boxes.

The exhaustive search for finding key bits can be limited by just taking the *good* pairs. For example in the example of SPN a good pair has

$$U_{41} = U_{43} = 0000$$

and feeds U_{42} and U_{44} by non zero input differential, where U_{ij} is input to the j'th S-box in the i'th round .

A decryption is done for all pairs of ciphertexts corresponding to the pairs of palintexts used to generate the input difference (ΔP in the example of figure 8).

Moreover a counter is maintained for every subkey (which are those key bits that are influenced by a non-zero differences in the differentials), the counters are incremented when the input to the last round corresponds to the expected value from differential characteristic. At the end of the procedure the subkey which has the greatest value is considered to be the correct subkey, then we do an exhaustive search to find the rest of the key bits. In example of figure 8 the differential characteristic has influence on S-boxes S_{42} and S_{44} of the last round, therefor for each pair of ciphertexts we try all 256 possibilities for $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$, and whenever the input difference to the final round determined by the decryption is $\Delta U_4 = [0000\,0110\,0000\,0110]$ then the counter is incremented.

4 Linear Attack against DES

In this section we will present an Linear attack on DES as described in [M94]. Before we do that we will give a brief description of DES.

4.1 The Data Encryption Standard

DES has been one of the most used block ciphers since 1976, but is now considered insecure, because of the small key size (56 bits). DES is a so called Feistel cipher, and consist of 16 rounds with a round 48 bits key for each round. The F function of is the same in all rounds and contains eight different S-boxes. Each S-box has six input bits and four output bits, which implies that the S-boxes are not invertible. But because of the structure of the Feistel cipher, invertible S-boxes are not necessary as we shall see.

4.2 Linear Approximations for DES

The best Linear Approximation for any DES S-box, is $X_2 \oplus Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4 = 0$ for S-box number five. This equation has probability $\frac{12}{64}$. If one analyzes the linear parts of the F function, one finds that the equation implies that

$$X_{15} \oplus F(X,K)_7 \oplus F(X,K)_{18} \oplus F(X,K)_{24} \oplus F(X,K)_{29} \oplus K_{22} = 0 \qquad (6)$$

have the same probability. We will use this equation to describe an attack against a 3 rounds DES cipher.

To attack the full 16 rounds DES cipher, one should use all of the following Linear Approximations:

$$X_{15} \oplus F(X,K)_7 \oplus F(X,K)_{18} \oplus F(X,K)_{24} \oplus F(X,K)_{29} \oplus K_{22} = 0$$

$$X_{27} \oplus X_{28} \oplus X_{30} \oplus X_{31} \oplus F(X,K)_{15} \oplus K_{42} \oplus K_{43} \oplus K_{45} \oplus K_{46} = 0$$

$$X_{29} \oplus F(X,K)_{15} \oplus K_{44} = 0$$

$$X_{15} \oplus F(X,K)_7 \oplus F(X,K)_{18} \oplus F(X,K)_{24} \oplus K_{22} = 0$$

$$X_{12} \oplus X_{16} \oplus F(X,K)_7 \oplus F(X,K)_{18} \oplus F(X,K)_{24} \oplus K_{19} \oplus K_{23} = 0$$

4.3 The attack

First we will describe an attack on the 3 rounds DES cipher (see figure 9). We will use equation (6) to do so. The equation and the figure tells us that

$$P.L_{15} \oplus (P.H_7 \oplus X2_7) \oplus (P.H_{18} \oplus X2_{18}) \oplus (P.H_{24} \oplus X2_{24}) \oplus (P.H_{29} \oplus X2_{29}) \oplus K^1_{22} = 0$$

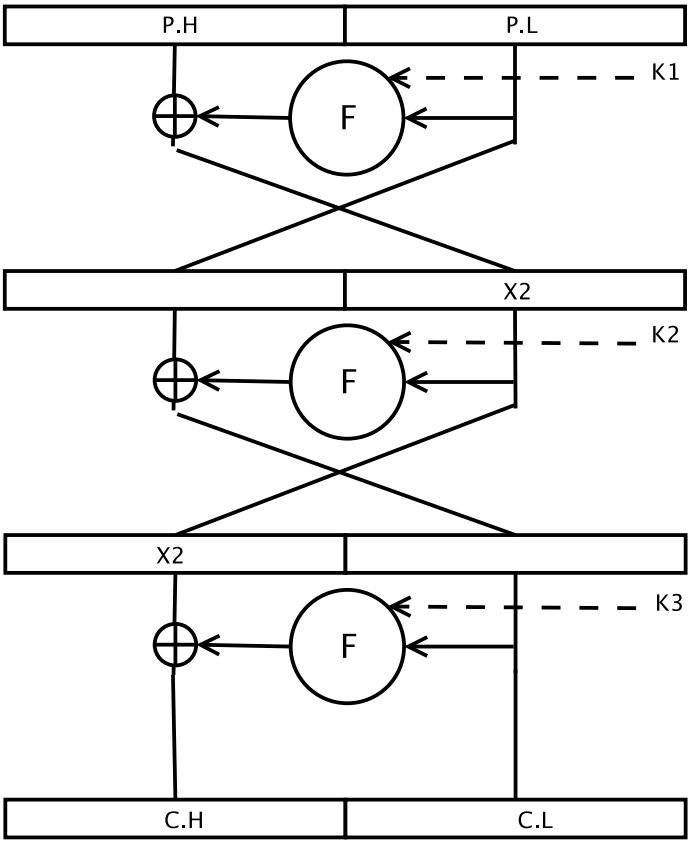

Figure 9: An overview of the 3 rounds DES cipher.

with the same probability $\frac{12}{64}$, where K^i is the round key for the i'th round. We can remove the key bit K^1_{22}, since we consider it to have a fixed value, and it therefore just determine if the probability is $\frac{12}{64}$ or $1 - \frac{12}{64}$. We dont mind knowing which case we are in.

Notice that $X2$ goes through the second round without being changed, and $X2$ is therefore input to the last round. So now we should invert the last round, and compute the four needed bits of $X2$, so we can check when the equation holds. $X2$ is equal to $C.H \oplus F(C.L, K^3)$, and the only unknown is K^3. We have to find the four bits of $X2$ using only a subset of the key bits. This is possible, since we only need four of the bit of $X2$. If we study the F function we find that the four bits come from a permutation, and before that the S-boxes. But since there are only four bits they can not come from more than four different S-boxes. To compute these S-boxes we need 6 key bits for each, which means that we only need 24 bits of K^3 to find the needed four bits of $X2$.

So now we can go through all the 2^{24} key candidates all our plaintext/ciphertext pairs, and compute for which key the equation is true, furthest from half the times. That candidate would be our guess for the corresponding key bits of K^3.

4.3.1 The full 16 rounds DES

To attack the full 16 rounds DES, one would need the equations as stated in subsection 4.2. Using these equations as approximations of the F-functions in DES

reduced to 15 rounds we get the following linear approximation:

$$P.H[7,18,24] \oplus P.L[12,16] \oplus C.H^{15}[7,18,24,29] \oplus C.L^{15}_{15} \oplus \Sigma_{key} = 0 \qquad (7)$$

Where $C.L^{15}$ and $C.H^{15}$ are the right and left sides of the ciphertext achieved from this reduced DES respectively. Σ_{key} is the XOR of several round key bits from different rounds, but as explained above we will however not care very much about these bits since they are fixed and thus their XOR is also fixed. [**?**] states that this is the best approximation of 15 rounds of DES and that it has bias 1.19^{-22}.

We can then extend the reduced DES to a full 16 DES cipher by adding one more round. Now since the 15'th round will no longer be the last round this will swap the positions of $C.L^{15}$ and $C.H^{15}$. Let us write $C.H^{15} = X.L^{15}$ and $C.L^{15} = X.H^{15}$. We then get the following approximation from (7):

$$P.H[7,18,24] \oplus P.L[12,16] \oplus X.L^{15}[7,18,24,29] \oplus X.H^{15}_{15} = 0 \qquad (8)$$

Notice that we have now omitted Σ_{key} as argued above. (8) is then the approximation we would like to check by inverting the last round of the 16 DES cipher.

Now let C.L and C.H be the right and left part of the ciphertext obtained from 16 round DES. We can then decipher the ciphertext of 16 round DES by one round to get the input to the 16'th round $X.L^{15}$ and $X.H^{15}$. By simple definition of DES we have that:

$$X.L^{15} = C.L$$

and

$$X.H^{15} = C.H \oplus F(X.L^{15}, K^{16}) = C.H \oplus F(C.L, K^{16})$$

So applying this to (8), we can check if the approximation holds for a given key K^{16} by checking if the following holds:

$$P.H[7,18,24] \oplus P.L[12,16] \oplus C.L[7,18,24,29] \oplus C.H^{15}_{15} \oplus F(C.L, K^{16})_{15} = 0$$

The only part of this equation that is not given in the plaintext/ciphertext pair is then the bit $F(C.L, K^{16})_{15}$. This bit however is only affected by a single S-box of F so only 6 bits of K^{16} is needed to check if the equation holds (since S-boxes take 6bit inputs). We can now obtain the correct 6 key bits as described above. The remaining key bits can then be obtained by exhaustive search.

In [M94] Matsui uses the round symmetry of DES to create an other linear approximation corresponding to (7) with the same bias. By also using this approximation, we can obtain further 6 key bits. So a total of 12 key bits can be found in this way. Matsui uses a slightly different algorithm from the one we have presented. This also obtains the bit Σ_{key} for each approximation, and this way one could obtain as much as 14 key bits. In [**?**] Matsui improved this algorithm by instead using a approximation of 14 round approximation of DES. This way he obtained 26 key bits.

Matsui estimates that the original attack needs 2^{47} plaintext/chipertext pairs in order to be successful with high probability and the improved attack needs only 2^{43} pairs. However [J01] shows experimental results that suggest that only 2^{41} pairs are really needed for the improved version of the attack.

5 Block cipher design to prevent these attacks

The obvious question is now; what can we do to make our SPN resistant against linear and differential cryptanalysis. [HT94] describes three design principles to make an SPN stronger against linear and differential attacks. Two of the points focus on S-box design, the last point adds a linear transformation between S-boxes besides the permutation.

5.1 Good Diffusion

Let p_{R-1} be the probability of a high probability differential characteristic, as defined in section 3.2, for round $R-1$ in a SPN. The number of plaintexts needed for a differential analysis attack to be successful with high probability can then be approximated to be at least $N_D = \frac{1}{p_{R-1}}$. In this subsection we describe a principle to decrease the probability of all characteristics for an SPN, and thus increasing N_D.

In order to be resistant to differential analysis, S-boxes of the SPN should have good diffusion properties. That is given inputs that has small differences the corresponding outputs of an S-box should have large differences. For example if two inputs to an S-box only differ on one bit, say 1000 and 1001 for our example SPN in the above sections, we want the corresponding outputs to be different on at least two bits. This motivates the following definition of the *diffusion order* of an S-box, which is a lower bound on the output difference for small difference input. Here hw is the hamming weight, which is the number of 1's in a bit string. Notice that $hw(\Delta\mathbf{X})$ is exactly the number of bits that are different from X' in X'' when $X' \oplus X'' = \Delta\mathbf{X}$.

Definition 5.1 (Diffusion Order)**.** An S-box satisfies diffusion order $\lambda \geqslant 0$ if, for $hw(\Delta\mathbf{X} > 0)$,

$$hw(\Delta\mathbf{Y}) > \begin{cases} \lambda + 1 - hw(\Delta\mathbf{X}), & hw(\mathbf{X}) < \lambda + 1 \\ 0, & otherwise \end{cases}$$

DES S-boxes satisfies a diffusion order of one. That is for inputs to a DES S-box that are only different on one bit the outputs will be different on at least two bits.

Active S-boxes are the S-boxes that are involved in a characteristic. Having many active S-boxes in a characteristic decreases the probability of the characteristic occurring. Combining S-boxes that satisfy high diffusion orders, with a permutation so that no two outputs of a round r S-box are connected to the same round $r+1$ S-box, we get an SPN where characteristics have relatively large amounts of *active* S-boxes. Intutively this is because the high diffusion order garranties that either the inputs to an active S-box, S_a, are different on many bits or the output is. The permutation then tells us that if in the former case, there were many active S-boxes in the round before S_a, and in the latter case that there will be many active S-boxes in the round after S_a.

From [HT94] we have the following upper bound, on the probability of characteristics in such an SPN:

Theorem 5.1. *Given an SPN with R rounds such that R is a multiple of 4, using a permutation as described above, and S-boxes satisfying diffusion order λ. The probability p_{R-1} of a $(R-1)$-round characteristic is bounded in the following way:*

$$p_{R-1} \leqslant p^{\frac{\lambda+2}{2}R-(\lambda+1)}$$

Where p is the best probability of input/output XOR's over an S-box.

This means that the higher diffusion order S-boxes of the SPN satisfy the lower is the probability that any characteristic holds.

So to make an SPN resistant to differential analysis we just need to make sure the S-boxes have high diffusion order. However such S-boxes are very rare. Assume that one XOR pair, $(\Delta\mathbf{X}, \Delta\mathbf{Y})$, for an S-box violating diffusion order 1 is independent of other XOR pairs violating the same diffusion order. [HT94] shows that under this assumption, the probability of randomly choosing an $n \times n$ bijective

S-box that satisfies diffusion order 1 is only:

$$(\frac{2^n - 1 - n}{2^n - 1})^{n2^{n-1}}$$

Of course since S-boxes that satisfy diffusion order $\lambda > 1$, by definition, must satisfy order $\lambda - 1$, it must also be at least as hard to find such S-boxes as S-boxes that only satisfy order $\lambda - 1$. Thus the above probability is an upper bound on the probability of finding S-boxes that satisfy diffusion orders above 1.

5.2 High Nonlinearity

The bias of a linear approximation for a linear analysis attack, gives us an estimate of how many plaintext/ciphertext pairs, we need to expect the attack to be successful. If the absolute value of the bias is high we will need many plaintext/ciphertext pairs. Thus insuring small absolute values of the bias of all linear approximation will make an SPN resistant to linear cryptanalysis.

In this subsection we give an upper bound on the absolute value of the bias of linear approximations of an SPN, based on the nonlinearity of the S-boxes. We will call the approximation maximizing the absolute value of the bias for the *best approximation*. Notice that this is not necessarily the approximation with the highest bias. Since

[HT94] describes the following measure of nonlinearity.

Definition 5.2 (Nonlinearity)**.** The nonlinearity of a boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$ the nonlinearity is:

$$NL(f) = min_{U_1,...,U_n,V \in \{0,1\}}(\#\{X | f(X) \neq \oplus_{i=1}^{n} U_i X_i \oplus V\})$$

For X_i being the i'th bit of X.

In other words the nonlinearity measures the amount of elements in $\{0,1\}^n$ for which the best linear approximation of f does not hold. The nonlinearity of an $n \times n$ bijective S-box, S, is then:

$$NL(S) = min_{W_1,...,W_n \in \{0,1\}, \Sigma_{i=1}^{n} W_i \neq 0}(NL(\oplus_{i=1}^{n} W_i f_i))$$

Where f_i is the function that on input x gives the i'th output bit of S on x.

For an R-round SPN where the S-box with the lowest nonlinearity has nonlinearity NL_{min}. The best approximation of an S-box then has bias ε where $|\varepsilon| \leqslant \frac{2^{n-1} - NL_{min}}{2^n}$. Let the best approximation of the entire SPN have bias γ. It can then be shown that:

$$|\gamma| \leqslant 2^{R-1}|\varepsilon|^R \leqslant 2^{R-1}|\frac{2^{n-1} - NL_{min}}{2^n}|^R$$

This means that if the nonlinearity, NL_{min}, of the S-box with minimum nonlinearity is high, the bias of the best approximation of is low. As stated above, for small biases we need a large amount plaintext/ciphertext pairs, thus by having S-boxes of high minimum nonlinearity an SPN will be more resistant against linear cryptanalysis.

Furthermore if the size of S-boxes increase the ratio of S-boxes of high nonlinearity grows. That is for large S-boxes we have high probability of choosing an S-box with high nonlinearity at random. For instance [HT94] describes making 200 randomly generated 8×8 bijective S-boxes of which more than half had high nonlinearity between 94 and 98.

5.3 Linear Transformation

In order to achieve diffusion properties we already use a linear transformation between rounds of S-boxes, namely the permutations of each round. In this subsection we describe a new kind of linear transformation that can be used between rounds to achieve better resistance against both differential and linear cryptanalysis.

We consider a SPN with an even number N of S-boxes in each round. Let $U = (U_{<1>}, U_{<2>}, \ldots, U_{<N>})$ be the output of some round of S-boxes, where $U_{<i>}$ is the output of the i'th S-box. We then define L to be the linear transformation so that

$$L(U) = (L_1(U), L_2(U), \ldots, L_N(U))$$

where

$$L_i(U) = (\oplus_{j=1}^{N} U_{<j>}) \oplus U_{<i>} = U_{<1>} \oplus U_{<2>} \oplus \ldots \oplus U_{<i-1>} \oplus U_{<i+1>} \oplus \ldots \oplus U_{<N>}$$

That is $L_i(U)$ is the XOR of all $U_{<j>}$'s so that $j \neq i$. Since we assume N to be even $L(U)$ can be inverted by, for each $L_i(U)$, XOR'ing all $L_j(U)$'s so that $j \leqslant i$, that is L is its own inverse.

If we assume P is the permutation function of the SPN, we can now modify the SPN using L, by making the input to the next round $P(L(U))$, instead of just $P(U)$. Using an SPN modified with such a linear transformation gives us the following bounds on the effectiveness of differential and linear cryptanalysis respectively:

Theorem 5.2. *Let an SPN be given as in theorem 5.1, modified with the linear transformation, L, as described above. If each $n \times n$ S-box of the SPN has $n \geqslant 3$, and satisfies diffusion order $\lambda \leqslant \frac{n-1}{2}$, the bound of theorem 5.1, still holds and for $\lambda = 0$ we have that:*

$$p_{R-1} \leqslant p^{\frac{3}{2}R-2}$$

This means that although we do not have any improvements over the result in subsection 5.1 for $\lambda > 0$, for $\lambda = 0$ we have as good a bound as we had for $\lambda = 1$ in subsection 5.1.

Theorem 5.3. *Let an SPN be given of an even number of rounds, R, and M $n \times n$ S-boxes, in each round, such that $M \geqslant n$. For such an SPN, modified by a linear transformation as described above, we have that the bias of the best possible $R-1$ round linear approximation, γ, satisfies:*

$$|\gamma| \leqslant 2^{\frac{3}{2}(R-1)} |\varepsilon|^{\frac{3}{2}R}$$

Where ε is the bias of the best linear approximation of any single S-box in the SPN.

In 5.2, we said that it could be shown that $|\gamma| \leqslant 2^{R-1}|\varepsilon|^R$, that however was based on the assumption that the best linear approximation could be one that involved only one S-box in each round. This tighter bound is due to the fact that modifying the SPN with the linear transformation, makes sure that at least three S-boxes must be involved for any two consecutive rounds in a linear approximation.

5.4 Conclusions On SPN Design Principles

Using these principles we can get very good bounds on the expected effectiveness of attacks based on differential and linear cryptanalysis. One should notice that the principles are not exclusive, and all of them could and should be used to give an SPN resistance against both forms of attack.

The table 10 borrowed from [HT94] shows estimates on the number of known and chosen plain texts that is expected to be needed for successful attacks using linear or differential cryptanalysis respectively, for different variations and combinations of the principles. The estimates are for an eight round SPN with 64bit blocks and a 64bit key. The SPN uses 8×8 bijective S-boxes with a minimum nonlinearity 96, and the probability of the best differential over any S-box is 2^{-4}. N_D^{min} is the estimated expected minimum of chosen plaintexts needed for a differential cryptanalysis attack, and N_L^{min} is the estimated expected minimum of plaintext/chipertext pairs needed for a linear cryptanalysis attack.

TYPE	λ	N_D^{min}	N_L^{min}
Permutation $\pi(\cdot)$	0	2^{28}	2^{34}
	1	2^{40}	
	2	2^{52}	
Linear Transform $\pi(\mathcal{L}(\cdot))$	0	2^{40}	2^{50}
	1	2^{40}	
	2	2^{52}	

Figure 10: Resitance of different SPN's

6 Conclusion

In this report we have presented and discussed two famous cryptanalysis methods, namely Linear and Differential cryptanalysis. Due to the extensive theory, we have decided to concentrate on the Linear Cryptanalysis, describing some details of the theory and explaining how it can be applied to the DES cipher. The theory of Differential Cryptanalysis has been described briefly for more information about how to determine better differential characteristics we refer to [M95].

The two attacks both have some very hard preconditions, in form of big amounts of chosen or known plaintext data. Yet one must consider these attacks when designing new block ciphers. Therefore one should always keep principles, such as the ones described in our last section, in mind when designing ciphers.

References

[D04] Ivan Damgård. Definitions and results for cryptosystems. pages 1–20, 2004.

[H02] Howard M. Heys. A tutorial on linear and differential cryptanalysis. *Cryptologia*, XXVI(3):189–221, 2002.

[HT94] H. M. Heys and S. E. Tavares. The design of substitution-permutation networks resistant to differential and linear cryptanalysis. In *CCS '94: Proceedings of the 2nd ACM Conference on Computer and communications security*, pages 148–155, New York, NY, USA, 1994. ACM Press.

[J01] Pascal Junod. On the complexity of matsui's attack. In *SAC '01: Revised Papers from the 8th Annual International Workshop on Selected Areas in Cryptography*, pages 199–211, London, UK, 2001. Springer-Verlag.

[M93] Mitsuru Matsui. Linear cryptanalysis method for des cipher. In *EUROCRYPT '93: Workshop on the theory and application of cryptographic techniques on Advances in cryptology*, pages 386–397, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[M94] Mitsuru Matsui. The first experimental cryptanalysis of the data encryption standard. In *CRYPTO '94: Proceedings of the 14th Annual International Cryptology Conference on Advances in Cryptology*, pages 1–11, London, UK, 1994. Springer-Verlag.

[M95] Mitsuru Matsui. On correlation between the order of s-boxes and the strength of des. In *EuroCrypt '94: Lecture Notes in Computer Sience no 950*, pages 366–375, London, UK, 1995. Springer-Verlag.

[S06] Douglas Stinson. *Cryptography: Theory and Practice,Third Edition*. CRC/C&H, 2006.