Information Technology Journal 3 (1): 49-56, 2004
ISSN 1682-6027

Theoretical Analysis of Linear Cryptanalysis Against DES (Data Encryption Standard)

Zheng-Quan Xu and Dereje Yohannes
Huazhong University of Science and Technology, Department of Computer Science, Wuhan, 430074, China

Abstract: Cryptanalysis is the art of deciphering encrypted communications without knowing the proper keys. Linear cryptanalysis against DES is one of the best and modern known-plaintext attacks. This paper describes the DES algorithm and we theoretically analyze the linear cryptanalysis of DES according to Matsuit's experimentation.

Key words: Cryptanalysis, encryption, decryption, cipher, key, s-box

INTRODUCTION

One of the primary reasons that intruders can be successful is that most of the information they acquire from a system is in a form that they can read and comprehend. When you consider the millions of electronic messages that traverse the Internet each day, it is easy to see how a well-placed network sniffer might capture a wealth of information that users would not like to have disclosed to unintended readers. Intruders may reveal the information to others, modify it to misrepresent an individual or organization, or use it to launch an attack[1]. One solution to this problem is, through the use of cryptography, to prevent intruders from being able to use the information that they capture.

Cryptography is mostly concerned with keeping messages secret[2]. The idea is that a plain text message is transformed to a cipher text message by means of a secret key. Nobody then should be able to obtain the plain text from the cipher text without knowing the key. The method of transforming plaintext to cipher text is called encryption and the method itself is called the encryption algorithm.

Cryptography plays an important role in modern data security. One of the first advances in cryptography was due to the publication of the Data Encryption Standard (DES). DES algorithm is designed to encipher (decipher) data in block format. It is based on non linear functions and is accepted as a secure algorithm. DES was adopted by the National Bureau of Standards[3], now the National Institute of Standards and Technology (NIST). It is widely used and at the same time widely studied to find a way of breaking it[2].

There are many cryptanalytic techniques, like Cipher text-only attack, Chosen-plaintext attack, Faults in crypto systems, Man-in-the-middle attack and known-plaintext attacks. Until this era, the successful approach to break DES cipher was differential cryptanalysis but Matsui[4] proposed a linear cryptanalysis technique which is superior to differential cryptanalysis. The method breaks DES faster instead of exhaustively testing of all possible keys. The attacker knows or can guess the plaintext for some parts of the cipher text and then he/she will decrypt the rest of the cipher text blocks using this information. This may be done by determining the key used to encrypt the data, or via some shortcut.

Description of des algorithm: DES is a block cipher that operates on 64-bit data using a 56-bit key. It is a symmetric cipher, which means that the key used for encryption will also be required for decryption (Kwan). The algorithm consists of two main functions the key schedule and the cipher functions; the key schedule produces sixteen 48-bit subkeys. A subkey is a selection of bits from the 56-bit key. The cipher function scrambles the data block that passes through 16 iterations or rounds of the cipher function and a different 48-bit subkey is applied in each round or iteration Fig. 1.

As you can see from the (Fig. 1) DES operates on a 64-bits block of plaintext. After an initial permutation (denoted IP), the block is split into a right half R_0 and a left half L_0, each 32-bits long. Then, following the Feistel cipher concept, there are 16 rounds of identical operations, called function F, in which the data are combined with 16 different subkeys K_i, which are derived from the key K using the key scheduling algorithm. At the end of the 16 rounds, the two parts L and R are combined and the inverse of IP (denoted IP^{-1}) finishes the algorithm.

Corresponding Author: Dereje Yohannes, Huazhong University of Science and Technology, Foreign students building, P.O. Box-430074, Hubie Province, Wuhan City

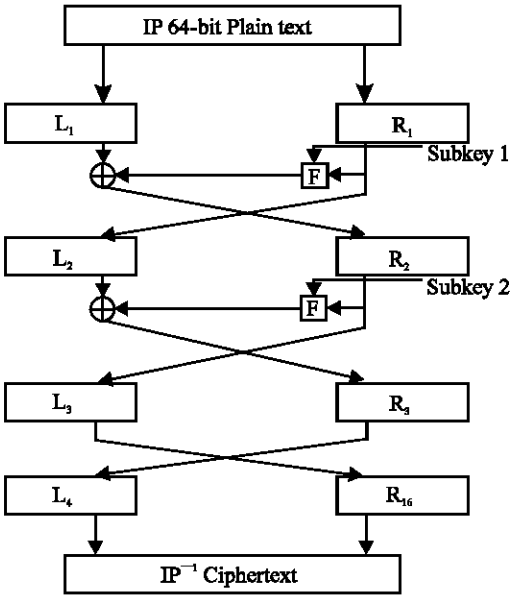

Fig. 1: The classical Feistel Network DES encryption algorithm

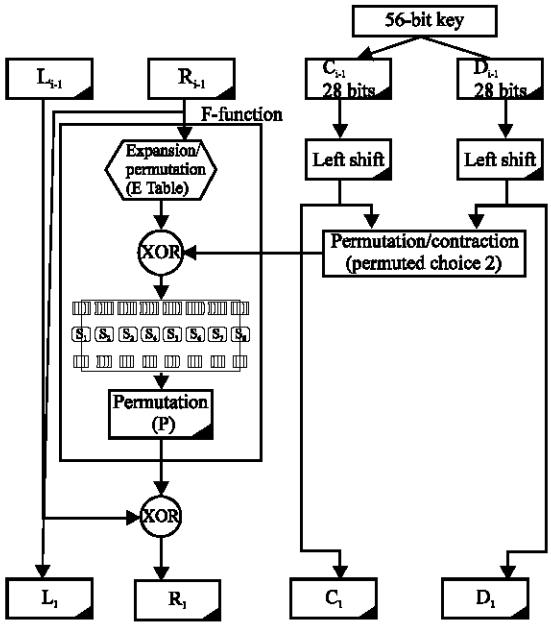

Fig. 2: F-function, Key schedule and S-box algorithms

From (Fig. 1) that illustrates the classical Feistel Network DES encryption algorithm we can deduce the following:

$$L_i = R_{i-1},$$
$$R_i = L_{i-1} \oplus F(R_{i-1}, K_i),$$

where each subkey K_i is derived from the key K.

The Feistel cipher structure is guaranteed to be reversible (or, in other words, one can use the same function to encrypt and to decrypt the data)[2]. Because XOR is used to combine the left half with the output of the round function, following equality holds:

$$L_{i-1} \oplus F(R_{i-1}, K_i) \oplus F(R_{i-1}, K_i) = L_{i-1}$$

Key schedule, the f function and s-boxes: As we described above, the DES key is often expressed as a 64-bits block, where the least significant bits of each bytes are ignored and used as parity check to ensure that the key is error-free. This operation is implemented by the so-called permuted choice, denoted PC1, which eliminates the superfluous bits and permutes the remaining ones. After this operation, a different 48-bits subkey is generated for each of the 16 rounds of DES in the following manner: first, the 56-bits key is divided into two 28-bits halves. Then, the halves are circularly shifted left by either one or two bits, depending of the round. After being shifted, 48 out of the 56 bits are selected by a compression permutation, often denoted PC2.

Because of the shifting, a different subset of key bits is used in each subkey. Each bit is used in approximately 14 of the 16 rounds, but not all bits are used exactly the same number of times. The key scheduling algorithm is illustrated in Fig. 2.

As you can see from (Fig. 2) one round of the F-function consists of the following operations: first, an expansion permutation, denoted E, expands the 32 bits of the right half of the data Ri to 48 bits, which are XORed with the corresponding subkey; this sum will be the input of the substitution stage (S-boxes). This operation changes the order of the bits as well as repeating certain bits. The goals of E are multiple: firist, it makes the right half the same size as the key for the XOR operation and second, it provides longer results that can be compressed during the substitution operation. Furthermore, it allows one bit to affect two substitutions, so the dependency of the output bits on the input bits spreads faster.

The substitution stage is composed of eight different S-boxes. Each S-box has an input of 6 bits and a 4 bits output. The 48 bits are divided into eight 6-bits subblocks. Each separate block is operated on by a separate S-box. S-box is a table of 4 rows and 16 columns (Table 1). The first and the last bit of the 6 input bits specify which row is used and the four inner bits specify the corresponding column.

Here under we theoretically analyzed the F-function algorithm by only selecting the third S-box (S3) as an example, the analysis of the rest of the s-boxes are quit identical so one can use the same procedure so as to determine the characteristics of the other boxes.

Let the Expansion Permutation (E) denote a function which takes a block of 32 bits as input and yields a block of 48 bits as output. Let E be such that the 48 bits of its output, written as 8 blocks of 6 bits each, are obtained by selecting the bits in its inputs in order according to the following table:

E bit-selection table

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Thus the first three bits of E(R) are the bits in positions 32, 1 and 2 of R while the last 2 bits of E(R) are the bits in positions 32 and 1. Each of the unique selection functions S1,S2,...,S8, takes a 6-bit block as input and yields a 4-bit block as output and is illustrated by using a table containing the recommended S3 as an example:

Table 1: S-box selection table for S3

Row No.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

If S3 is the function defined in this table and B is a block of 6 bits, then S3(B) is determined as follows: The first and last bits of B represent in base 2 a number in the range 0 to 3. Let that number be i. The middle 4 bits of B represent in base 2 a number in the range 0 to 15. Let that number be j. Look up in the table the number in the i'th row and j'th column. It is a number in the range 0 to 15 and is uniquely represented by a 4 bit block. That block is the output S3(B) of S3 for the input B. For example, for input 110001 the row is 11, that is row 3 and the column is determined by 1000, that is column 8. In row 3 column 8 appears 4 so that the output is 0100. Selection functions S1,S2,...,S8 of the algorithm table appear in any Cryptography book.

The permutation function P yields a 32-bit output from a 32-bit input by permuting the bits of the input block. Such a function is defined by the following table:

P

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

The output P(L) for the function P defined by this table is obtained from the input L by taking the 16th bit of L as the first bit of P(L), the 7th bit as the second bit of P(L) and so on until the 25th bit of L is taken as the 32nd bit of P(L). This permutation function P of the algorithm can be found in any cryptography book.

Now let S1,...,S8 be eight distinct selection functions, let P be the permutation function and let E be the function defined above. To define F(R,K) we first define B1,...,B8 to be blocks of 6 bits each for which

$$B1B2...B8 = K \oplus E(R)$$

The block F(R,K) is then defined to be

$$P(S1(B1)S2(B2)...S8(B8))$$

Thus $K \oplus E(R)$ is first divided into the 8 blocks as indicated in Fig. 2. Then each Bi is taken as an input to Si and the 8 blocks S1(B1), S2(B2), ..., S8(B8) of 4 bits each are consolidated into a single block of 32 bits which forms the input to P. The output is then the output of the function F for the inputs R and K.

Security considerations: The S-box substitution is the critical step in DES, regarding as well its implementation or its security. The algorithm's other operations are all linear and easy to analyze, while the S-boxes are the only non-linear steps.

The end of the F-function consists of a straight permutation, the P-box permutation P. This permutation maps each output bit of the substitution stage to an output position, i.e. no bits are used twice and no bits are ignored. Finally, the output of P is XORed with the left half of the initial 64-bits block. Then, the left and right halves are permuted, following the Feistel cipher concept and another round can begin.

The most interesting part of DES is its S-boxes. All the operations in DES, except For the S-boxes, are linear. If the S-boxes were also linear then we would be able to express the operation of DES as linear equation of the plaintext, ciphertext and key. We can demonstrate this by simulating the expression of 3-round DES. (derived from Fig. 1)

$$L_1 = R_o$$
$$R_1 = L_o \oplus F(R_o, K_1)$$
$$L_2 = R_1,$$
$$R_2 = L_1 \oplus F(R_1, K_2),$$
$$L_3 = R_2,$$
$$R_3 = L_2 \oplus F(R_2, K_3)$$
$$L_4 = R_3,$$
$$R_4 = L_3 \oplus F(R_3, K_4)$$

For simplicity expanding the round-3 equation we get,

$$L_3 = R_o \oplus F(L_o \oplus F(R_o, K_1), K_2),$$
$$R_3 = L_o \oplus F(R_o, K_1) \oplus F(R_o \oplus F(L_o \oplus F(R_o, K_1), K_2), K_3).$$

We are unable to simplify the equations because the linear (permutation) and nonlinear (Substitution) operations in the cipher function do not commute. The DES equation becomes increasingly complex as more rounds are added. Compare equation with the equation obtained from 3-round DES with linear S-boxes:

$$L3 = R_o \oplus P(S(E(L_o))) \oplus P(S(E(R_o))) \oplus P(S(K_1)) \oplus P(S(K_2)),$$
$$R_3 = L_o \oplus P(S(E(L_o))) \oplus P(S(E(R_o))) \oplus P(S(K_2)) \oplus P(S(K_3)).$$

The final linear equation always remains simple since some terms introduced from previous rounds cancel out (i.e., additional rounds would not increase security if the round function is linear).

We can calculate the S-boxes that give the result of the cumulative substitutions of all n rounds (because the S-boxes are linear). We let the inverse S-boxes S^{-1} be the inverse of these "n round S-boxes". Using just one plain text/ciphertext pair we obtain:

$$K_1 \oplus K_3 = S^{-1}(P^{-1}(L_o \oplus R_o \oplus L_3 \oplus R_3))$$

As the S-boxes are 6-bit to 4-bit mappings, we cannot solve them uniquely . We use the above equation as the basis for the attack on "linear DES". Inverting linear S-box S_1would give 4 linear equations (each equation corresponds to an output value in a row of the S-box, as each row is a permutation). Since we require 6 bits of S-box input, we fix the two LSBs of the S-box input and evaluate the other four using the S_1^{-1} linear expressions. The two bits which we fixed would be shared by 2 linear S-box S_2 because of the E-box, so we can automatically evaluate all six input bits to S_2 from the linear equations obtained by S_2^{-1}. Two bits are shared between the inputs of S_2 and S_3 and so we can evaluate the input to S_3 and so on. Thus, to invert all the S-boxes we only need to try possibilities for 2 input bits to S_1, so we would have 4 linear expressions (1 correct, 3 incorrect), each representing 48 key bits.

By guessing the value of those key bits which appear in either K1 or Kn, but not both, we can recover 48 bits of the "key". In case of 3-round linear DES, there are only 6 key bits not shared by K_1 and K_3 The remaining 8 key bits can be recovered by reduced exhaustive key search. Thus the complexity of an attack on DES if it had linear S-boxes (satisfying the conditions described above) would be (2^{16}). Although we have put restrictions on the structure of the S-boxes to make this analysis easier, the point is that linear ciphers, even DES-like linear ciphers, are solvable by known-plain text attacks.

Linear cryptanalysis of DES: The linear cryptanalysis is one of the most famous generic attack against block ciphers (AES). It was proposed by Matsui in[4]. where he shows that DES can be broken with the help of 247 known plaintext-ciphertext pairs faster than an exhaustive search. Later, in a following paper , he refines his technique and shows that it is sufficient to have 243 known plaintext-ciphertext pairs at disposal; furthermore, he implements it and breaks DES in 50 days with the help of 12 computers. Although this attack has only a theoretical importance, the linear cryptanalysis is the most powerful one on DES to date.

Linear cryptanalysis principles: The first step in Matsui's method is to find a linear approximation of DES cipher with some probability of $p \neq ½$ .That is:

$$P^{[i1,\ldots,ia]} \oplus C^{[j1,\ldots,jb]} = K^{[k1,\ldots,kc]} \qquad (2.1)$$

$$\left(\bigoplus_{i\in\{1\ldots64\}} P^{(i)}\right) \oplus \left(\bigoplus_{j\in\{1\ldots64\}} C^{(i)}\right) = \bigoplus_{k\in\{1\ldots64\}} K^{(i)}$$

Where P, C and K denote plaintext- ciphertext- and key-bits respectively and the Boolean operator XOR. The indices i, j and k denote fixed bit locations.

For some integer a, b and c, such that $0 \subseteq a, b \subseteq 64$ and $0 < c \subseteq 56$.

Each side of this equation represents one bit, therefore, the magnitude of [p _ ½] is equivalent to the probability of the equation to be true or false.

Algorithm 1: Let T be the number of plaintexts such that the left hand side of equation 2.1 is equal to Zero. If T >N/2 (N denotes the number of plaintexts), then guess

$$K^{[k1,\ldots kc]} = 0 \text{ (when } p > ½) \text{ or } 1 \text{ (when } p < ½),$$

else guess

$$K^{[k1,\ldots kc]} = 1 \text{ (when } p > ½) \text{ or } 0 \text{ (when } p < ½)$$

The success rate of Algorithm 1 can be determined by N and p and this rate clearly increases when N or [p-1/2] does.

In practice, for breaking n-round DES the linear expression of (n-2)-round DES is used. Therefore, the results of the F function in the first and the last rounds affect the linear expression of n rounds.
The result becomes:

$$P^{[i1,\ldots,ia]} \oplus C^{[j1,\ldots,jb]} \oplus F1^{[u1,\ldots,ud]} \oplus Fn^{[v1,\ldots,ve]} = K^{[k1,\ldots kc]}, \quad 2.2$$

for some integer a , b , c , d and e , such that $0 \subseteq a$; $b \subseteq 64$, $0 < c \subseteq 56$ and $0 \subseteq d$; $e \subseteq 32$. This type of linear expression makes the performance of the method easier.

Algorithm 2 is used to suggest the right side of equation 2.2, based on some candidate key bits.

Algorithm 2: Let $K_1^{(i)}$ (i = 1,2,...) and $K_n^{(j)}$ (j = 1,2,...) be possible candidates for K_1 and K_n, respectively. Then for each pair $(K_1^{(i)}, K_n^{(j)})$, let $T_{i,j}$ be the number of plaintext such that the left side of equation 2.2 is equal to zero. Let T_{max} be the maximal value and T_{min} be the minimal value of all $T_{i,j}$'s.

If $[T_{max}–N/2]>[T_{min}–N/2]$, then adopt the key candidate corresponding to T_{max} and guess

$$K^{\{k1,...kc\}} = 0 \text{ (when } p > ½) \text{ or } 1 \text{ (when } p < ½)$$

If $[T_{max}–N/2]<[T_{min}–N/2]$, then adopt the key candidate corresponding to T_{min} and guess

$$K^{\{k1,...kc\}} = 1 \text{ (when } p > ½) \text{ or } 0 \text{ (when } p < ½).$$

The success rate of this algorithm, similar to algorithm 1, increases when N or $[p - ½]$ does. The algorithm gives no suggestion when two absolute values are equal.

Now to find such a linear expression (equation 2.2), we start with the linear approximation of S-Boxes.

Then, these linear approximations are combined to make up the actual approximation of n rounds.

Linear approximation of s-boxes: Linear approximation of S-Boxes finds a linear relation between input bits and output bits of S-Boxes.

Notation: For a given S-Box Sa (a = 1,2,…,8), $1 \subseteq \alpha \subseteq 63$ and $1 \subseteq \beta \subseteq 15$, we define $N S_a(\alpha,\beta)$ to be the number of inputs, out of 64 possible inputs of Sa, that an XORed value of the input bits masked by α agreed with an XORed value of the output bits masked by β. That is ,

$$N S_a(\alpha,\beta) \overset{def}{=} \#\{x \mid 0 \le x < 64, (\bigoplus_{s=0}^{5} (x^{[s]} \bullet \alpha^{[s]})) = (\bigoplus_{s=0}^{3} (S_a^{[i]}(x) \bullet \beta^{[s]}))\}$$

This definition gives eight 63 by 15 tables that are called distribution tables . One can find the complete distribution table for all the eight S-boxes in any cryptography book. In the distribution tables, each element is decremented by 32, because only the difference between $N S_a(\alpha,\beta)$ and 32 is important. From these tables it is clear that only a few elements are far from 32.

Each element of the distribution tables gives a linear expression based on some bits of input and output of the F function and some bits of the subkey for that particular F function. Each linear expression is true with probability

$$p = \frac{N S_a(\alpha,\beta)}{64}$$

Hence, only those elements of distribution tables that are far from 32 are important. We note that, for those elements that are much bigger than 32, the corresponding linear expression will be true with some probability $p > ½$, but, for the elements that are much smaller than 32, the corresponding linear expression will be false with some probability $p > ½$. For example,

$$N S_5(16,15) = 12,$$

is true with probability $p = N S_5(16,15)/64 = 12/64 = 0.19$.

The important point is that $N S_5(16,15) \neq 12$ with probability of p= 1-0.19 = 0.81, which is a high probability.

Here we will show how a linear approximation for S_5 can be extracted from $N S_5(16,15)$based on the distribution table of S_5. $N S_5(16,15)$shows that the forth bit of the input of S_5 coincides with XOR of all four bits of outputs of S_5, with probability p = 12/64.

The four bits of the output of S_5 are bits 12, 13, 14 and 15 of the output block of S-Boxes. After P permutation, they become bits 7, 18, 24 and 29 of output of F function.

The fourth bit of the input of S_5 is bit 22 of the input block of S-Boxes, which is the XOR result of the bit 22 of the subkey and the output of the expansion function E. Finally, bit 22 in the output of the expansion function E, is bit 15 of the input of F function.

The above description starts from the correlation between input and output of S_5 and finds the correlation among the subkey and the input and output of F function.

This correlation is called linear approximation of F function. It can be written as:

$$X_i^{[15]} \oplus F_i^{[7,18,24,29]} = K_i^{[22]} \qquad 2.3$$

Now we extend the approximation to a number of rounds. We do this by compounding linear expressions. For example, to extend equation 2.3 which is a one-round linear approximation to 3-round DES, we start by expressing it for the first round. If P_H is the left half of the plain text block and P_L is the right half of the plain text (that enters the cipher function F), then we have:

$$F(P_L, K_1)^{[7,18,24,29]} = P_H^{[7,18,24,29]} \oplus X_2^{[7,18,24,29]} \qquad 2.4$$

Where X_2 is the data block as it enters F in the second round. Substituting equation 2.4 in equation 2.3, we obtain the linear approximation for the first round:

$$X_2^{[7,18,24,29]} \oplus P_H^{[7,18,24,29]} \oplus P_L^{[15]} = K_1^{[22]} \quad 2.5$$

Similarly, we apply the linear approximation (equation 2.3) to the final round:

$$X_2^{[7,18,24,29]} \oplus C_H^{[7,18,24,29]} \oplus C_L^{[15]} = K_3^{[22]} \quad 2.6$$

Where C_L and C_H are the right and left halves of the ciphertext, respectively. Adding equations 2.5 and 2.6 cancels the unknown X_2 term giving a linear approximation to a 3–round DES:

$$P_H^{[7,18,24,29]} \oplus C_H^{[7,18,24,29]} \oplus P_L^{[15]} \oplus C_L^{[15]} = K_1^{[22]} \oplus K_3^{[22]} \quad 2.7$$

The probability that equation 2.7 holds for random plaintext P and its corresponding ciphertext C is (12/64)2 + (1- 12/64) = 0.70. This is because it will hold if and only if either both one-round approximation hold, or if neither hold. The probability can also be calculated using equation 2.8:

$$p = \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} (ps - \frac{1}{2})$$

$$p = \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} (ps - \frac{1}{2}) = \frac{1}{2} + 2^{2-1} \prod_{i=1}^{2} (pi - \frac{i}{2})$$

$$= \frac{1}{2} + 2(\frac{12}{64} - \frac{1}{2})(\frac{12}{64} - \frac{1}{2}) = 0.70.$$

In the next section we will describe how these theoretical results are used to break 12- round DES.

The 12-rounds DES attack: In order to break 12-rounds DES, Matsui shows in[4] how it is possible to improve the attack described previously. First of all, he works with linear expression on 10 rounds of DES . Each equation has two active S-boxes (S-box 1 and S-box 5) and can recover each 13 bits of the key, or 26 in total. Here we will analayze the 12-rounds DES attack by using the previous theoretical results.

According to Matsui[4] principle we know that to break an n-round DES , we need the linear approximation of (n-2)-round DES. Therefore , for our case so as to break 12 round-DES we have to analyze the best expression of 10-round DES. They are the expression corresponding to N S_5(16,15), N S_1(4,4) and N S_5(16,14).

$$X_i^{[15]} \oplus F_i^{[7,18,24,29]} = K_i^{[22]} \quad 2.9$$

$$X_i^{[29]} \oplus F_i^{[15]} = K_i^{[44]} \quad 2.10$$

$$X_i^{[15]} \oplus F_i^{[7,18,24]} = K_i^{[22]} \quad 2.11$$

Using equation 2.9 in the second, eighth and tenths rounds, equation 2.10 in the third and seventh rounds and equation 2.11 in the forth and sixth rounds will give:

$$X_2^{[15]} \oplus F_2^{[7,18,24,29]} = K_2^{[22]}$$
$$X_8^{[15]} \oplus F_8^{[7,18,24,29]} = K_8^{[22]}$$
$$X_{10}^{[15]} \oplus F_{10}^{[7,18,24,29]} = K_{10}^{[22]}$$
$$X_3^{[29]} \oplus F_3^{[15]} = K_3^{[44]}$$
$$X_7^{[29]} \oplus F_7^{[15]} = K_7^{[44]}$$
$$X_4^{[15]} \oplus F_4^{[7,18,24]} = K_4^{[22]}$$
$$X_6^{[15]} \oplus F_6^{[7,18,24]} = K_6^{[22]}$$

As a general equation we have :

$$F_2 = P_L \oplus X_3,$$
$$F_i = X_{i-1} \oplus X_{i+1}, \ 3 \le I \le 9,$$
$$F_{10} = X_9 \oplus C_H,$$
$$X_{10} = C_L$$

If we apply these expressions to the above equation we will obtain:

$$X_2^{[15]} \oplus (P_L^{[7,18,24,29]} \oplus X_3^{[7,18,24,29]}) = K_2^{[22]}$$
$$X_8^{[15]} \oplus (X_7^{[7,18,24,29]} \oplus X_9^{[7,18,24,29]}) = K_8^{[22]}$$
$$C_L^{[15]} \oplus (X_9^{[7,18,24,29]} \oplus C_H^{[7,18,24,29]}) = K_{10}^{[22]}$$
$$X_3^{[29]} \oplus (X_2^{[15]} \oplus X_4^{[15]}) = K_3^{[44]}$$
$$X_7^{[29]} \oplus (X_6^{[15]} \oplus X_8^{[15]}) = K_7^{[44]}$$
$$X_4^{[15]} \oplus (X_3^{[7,18,24]} \oplus X_5^{[7,18,24]}) = K_4^{[22]}$$
$$X_6^{[15]} \oplus (X_5^{[7,18,24]} \oplus X_7^{[7,18,24]}) = K_6^{[22]}$$

If we XOR and simplify the above equations we will get

$$P_L^{[7,18,24,29]} \oplus C_H^{[7,18,24,29]} \oplus C_L^{[15]}$$
$$= K_2^{[22]} \oplus K_3^{[44]} \oplus K_4^{[22]} \oplus K_6^{[22]} \oplus K_7^{[44]} \oplus K_8^{[22]} \oplus K_{10}^{[22]} \quad 2.12$$

This is the first best expression of 10-round DES. To find the probability p of the equation we use equation 2.8

$$p = \frac{1}{2} + 2^{7-1} \prod_{i=1}^{7} (pi - \frac{1}{2})$$

$$= \frac{1}{2} + 2^6 (\frac{12}{64} - \frac{1}{2})^3 (\frac{30}{64} - \frac{1}{2})^2 (\frac{42}{64} - \frac{1}{2})^2 :$$

$$p = ½ - 1.53 \times 2^{-15}$$

By using the same method as used above we determined the second best expression of 10-round. We apply equation 2.9 to the first, third and ninth rounds, equation 2.10 to the forth and eighth rounds and equation 2.11 to the fifth and seventh rounds and we obtain the following result with the same probability

$$p = ½ - 1.53 \times 2^{-15}$$

$$C_L^{[7,18,24,29]} \oplus P_H^{[7,18,24,29]} \oplus P_L^{[15]}$$
$$= K_9^{[22]} \oplus K_8^{[44]} \oplus K_7^{[22]} \oplus K_5^{[22]} \oplus K_4^{[44]} \oplus K_3^{[22]} \oplus K_{11}^{[22]} \quad 2.13$$

Now, since we have the best expression of 10-round DES we can easily break 12-round DES by using equation

2.12 and 2.13. If we approximate the F function from second to eleventh round by using the above equations we can get:

$$X_2^{[7,18,24,29]} \oplus X_{12}^{[7,18,24,29]} \oplus X_{11}^{[15]} = K_3^{[22]} \oplus K_4^{[44]} \oplus K_5^{[22]} \oplus K_7^{[22]} \oplus K_8^{[44]} \oplus K_9^{[22]} \oplus K_{11}^{[22]} \quad 2.14$$

$$X_{11}^{[7,18,24,29]} \oplus X_1^{[7,18,24,29]} \oplus X_2^{[15]} = K_{10}^{[22]} \oplus K_9^{[44]} \oplus K_8^{[22]} \oplus K_6^{[22]} \oplus K_5^{[44]} \oplus K_4^{[22]} \oplus K_2^{[22]} \quad 2.15$$

Using the following expressions :

$$X_2 = P_H \oplus F_1,$$
$$X_{12} = C_L,$$

$$X_{11} = C_H \oplus F_{12}$$
$$X_1 = P_L$$

And substituting these values in 2.14 and 2.15 yields:

$$P_H^{[7,18,24,29]} \oplus F_1^{[7,18,24,29]} \oplus C_L^{[7,18,24,29]} \oplus C_H^{[15]} \oplus F_{12}^{[15]} = K_3^{[22]} \oplus K_4^{[44]} \oplus K_5^{[22]} \oplus K_7^{[22]} \oplus K_8^{[44]} \oplus K_9^{[22]} \oplus K_{11}^{[22]} \quad 2.16$$

$$C_H^{[7,18,24,29]} \oplus F_{12}^{[7,18,24,29]} \oplus P_L^{[7,18,24,29]} \oplus P_H^{[15]} \oplus F_1^{[15]} = K_{10}^{[22]} \oplus K_9^{[44]} \oplus K_8^{[22]} \oplus K_6^{[22]} \oplus K_5^{[44]} \oplus K_4^{[22]} \oplus K_2^{[22]} \quad 2.17$$

The probability of these equations are calculated using eq. 2.8 and it is $p = \frac{1}{2} - 1.53 \times 2^{-15}$, for randomly chosen plaintexts and their corresponding ciphertext.

We note that one decrypts two rounds instead of a single one. One can expect that this fact will amplify the randomization effect in case of wrong subkey candidates.

Let's define now the concept of effective text bits and of effective key bits. They are simply the bits which affect the left part of the linear approximations. We count the XORed value of several text- or key-bits affecting the left side of our expressions as one effective bit.

The effective bits of the two linear expressions are the following:

It is obvious that $P_H^{[7,18,24,29]} \oplus C_L^{[7,18,24,29]} \oplus C_H^{[15]}$ can be considered as one effective text bit. To find the effective bits for $F_{12}^{[15]}$. The input of S1 is the XOR result of $K_{12}^{[42]} \sim K_{12}^{[47]}$ and bits 42, 43, …., 47 of the expansion function E. Those bits are $X_{12}^{[27]} \sim X_{12}^{[31]}$ and $X_{12}^{[0]}$,or in fact $C_L^{[27]} \sim C_L^{[31]}$ and $C_L^{[0]}$. Thus, other six effective text bits are $C_L^{[27]} \sim C_L^{[31]}$ and $C_L^{[0]}$. Similarly for $F_1^{[7,18,24,29]}$ we have $P_L^{[11]} \sim P_L^{[16]}$ as effective text bits, so we have found totally 13 effective text bits for equation 2.16.

To find effective key bits, we use the same method and find out that $K_{12}^{[42]} \sim K_{12}^{[47]}$ and $K_1^{[18]} \sim K_1^{[23]}$ are effective key bits for the same equation. Similarly for equation 2.17 we find 13 effective key bits: $C_L^{[11]} \sim C_L^{[16]}$, $P_L^{[0]}$, $P_L^{[27]} \sim P_L^{[31]}$ and $C_H^{[7,18,24,29]} \oplus P_L^{[7,18,24,29]} \oplus P_H^{[15]}$ and 12 effective key bits: $K_1^{[42]} \sim K_1^{[47]}$ and $K_{12}^{[18]} \sim K_{12}^{[23]}$.

We can note that 13 text-bits can be used to derive 12 key-bits and the bit of the right side in each equation. We obtain hence a total of 26 (fortunately not duplicated) secret key-bits from the both equations, using 26 bits of text.

Breaking algorithm of 12-round DES

Algorithm 2.3: Here is the algorithm derived according to the above mathematical expressions to break the 12-round DES[5].

Prepare 2^{13} counters for each linear expression denoted thereafter $C_1^{(i)}$ and $C_2^{(i)}$ with $0 \le i \le 2^{13}$. Initialise them to 0.
/* Each indes I corresponds to the state of 13 effective text bits. */ FOR 2^{13} plaintext-ciphertext pairs (p, c) DO
Compute the values i_1 and i_2 using p and c for linear expression l_1 and l_2. Inerement by one $C_1^{(i1)}$ and $C_2^{(i2)}$.
END
Prepare 2^{12} counters for each linear expression denoted thereafter $K_1^{(k)}$ and $K_2^{(k)}$ with $0 \le k \le 2^{12}$.
/* Each counter corresponds to the state of 12 effective key bits. */ FOREACH k_1, k_2 DO

$$K_1^{(k1)} := \sum \{C_1^{(x)} \text{ such that } l_1(p_x, c_x, k_1)=0\}.$$
$$K_2^{(k2)} := \sum \{C_2^{(x)} \text{ such that } l_2(p_x, c_x, k_2)=0\}.$$

END
Sort K_1's and K_2's by decreasing magnitude $|K_x^{(y)} - 2^{12}|$ with $x \in \{1, 2\}$ and $0 \le y \le 2^{12}$. We get two lists of 2^{12} counters each that we denote $S_1^{(r)}$ and $S_2^{(r)}$ with $0 \le r \le 2^{12}$.
/* Guess of the last bit of information for each subkey candidate */ FOREACH $S_x^{(r)}$ DO
IF $|S_x^{(r)} - 2^{12}| \le 0$ THEN guess that right side of linear expression x ia 0. END
IF $|S_x^{(r)} - 2^{12}| \le 0$ THEN guess that right side of linear expression x ia 1. END
/* Combination of the two lists */
Let $f(r_1, r_2) := (r_1 + 1).(r_2+1)$, where r_x is the index corresponding to the sorted list x.
Build the final ranking of the subkey candidates by combining the two lists S_1 and S_2 by increasing f(,)-value. We denote this final list by F.
FOREACH subkey candidate in F DO
Search the remaining 30 bits.
IF good key found THEN EXIT END
END

We are now able to give the whole algorithm for breaking DES. According to the above algorithm, for each subkey candidate and for both linear approximations, we count the number of times that the left side of the linear expressions is equal to 0. Then, the resulting value of the counters must reflect the reliability of the corresponding subkey candidate.

In other words, we will get a list of subkey candidates for each linear expression and it is possible to sort these lists, the most likely 13-bits subkey candidate (i.e. the one which produces the biggest bias in the linear approximation) being in first position, the least likely one being at the end of a list.

We have then to combine these two lists, in order to provide a sorted list of 26-bits subkey candidates. The exhaustive search will then occur from the most likely candidate to the least likely one. This allows to reduce the number of known plaintext-ciphertext pairs from 2^{47} to 2^{43}. A formal description of this process is described in Algorithm 2.3. The algorithm is suitable for computer implementation so one can use the algorithm so as to break and analyze the 12-round DES algorithm. On the basis of Matsui's paper, we need 2^{32} pairs of plaintexts and ciphertexts to reach a correct answer with 94% success rate[4].

In this article we performed a theoretical analysis on both the algorithm of DES and linear cryptanalysis against DES in order to get a better insight and understanding in the real Data Encryption Standard (DES) algorithm and linear cryptanalysis technique. Based on Matsui's principle, we have analyzed the algorithm and we derived mathematical expressions. The analysis can be used as a first step to break any round DES algorithm based on the theoretical and mathematical expressions that we have analyzed.

We believe a lot of theoretical work and manipulations are still necessary in order to give a really accurate measure of the average complexity of the attack. Anyway, at this time we mathematically approved Matsui's principle as it is capable to break any round of DES algorithms.

REFERENCES

1. Yohannes, D. and Zheng-Quan, 2003. The current security awareness and reliability, Pakistan J. Applied Sci., Vol.3, 2003.
2. Stallings, W., 2000. Network security Essentials, Prentice-Hall.
3. National Bureau of Standards, 1997. Data Encryption Standard. U. S. Department of Commerce.
4. Matsui, M., 1994. Linear Cryptanalysis of DES Cipher (I), version 1.03.
5. Pascal Junod, Linear Cryptanalysis of DES, Diploma Thesis, Zurich.