Differential Cryptanalysis

of

DES-like Cryptosystems

Eli Biham Adi Shamir

The Weizmann Institute of Science
Department of Apllied Mathematics

July 19, 1990

Abstract

The Data Encryption Standard (DES) is the best known and most widely used cryptosystem for civilian applications. It was developed at IBM and adopted by the National Buraeu of Standards in the mid 70's, and has successfully withstood all the attacks published so far in the open literature. In this paper we develop a new type of cryptanalytic attack which can break the reduced variant of DES with eight rounds in a few minutes on a PC and can break any reduced variant of DES (with up to 15 rounds) in less than 2^{56} operations. The new attack can be applied to a variety of DES-like substitution/permutation cryptosystems, and demonstrates the crucial role of the (unpublished) design rules.

1 Introduction

Iterated cryptosystems are a family of cryptographically strong functions based on iterating a weaker function n times. Each iteration is called a *round* and the cryptosystem is called an *n-round cryptosystem.* The *round function* is a function of the output of the previous round and of a *subkey* which is a key dependent value calculated via a *key scheduling* algorithm. The round function is usually based on S boxes, bit permutations, arithmetic operations and the exclusive-or (denoted by $\oplus$ and XOR) operations. The *S boxes* are nonlinear translation tables mapping a small number of input bits to a small number of output bits. They are usually the only part of the cryptosystem that is not linear and thus the security of the cryptosystem crucially depends on their choice. The bit permutation is used to rearrange the output bits of the S boxes in order to make the input bits of each S box in the following round depend on the output of as many S boxes as possible. The XOR operation is often used to mix the subkey with the data. In most applications the encryption algorithm is assumed to be known and the secrecy of the data depends only on the secrecy of the randomly chosen key.

An early proposal for an iterated cryptosystems was Lucifer[7], which was designed at IBM to resolve the growing need for data security in its products. The round function of Lucifer has a combination of non linear S boxes and a bit permutation. The input bits are divided into groups of four consecutive bits. Each group is translated by a reversible S box giving a four bit result. The output bits of all the S boxes are permuted in order to mix them when they become the input to the following round. In Lucifer only two fixed S boxes (S_0 and S_1) were chosen. Each S box can be used at any S box location and the choice is key dependent. Decryption is accomplished by running the data backwards using the inverse of each S box.

The Data Encryption Standard (DES) [15] is an improved version of Lucifer. It was developed at IBM and adopted by the U.S. National Bureau of Standards (NBS) as the standard cryptosystem for sensitive but unclassified data (such as financial transactions and email messages). DES has become a well known and widely used cryptosystem. The key size of DES is 56 bits and the block size is 64 bits. This block is divided into two halves of 32 bits each. The main part of the round function is the F *function*, which works on the right half of the data using a subkey of 48 bits and eight (six-bit to four-bit) S boxes. The 32 output bits of the F function are XORed with the left half of the data and the two halves are exchanged. The complete specification of the DES algorithm appears in [15].

An extensive cryptanalytic literature on DES was published since its adoption in 1977. Yet, no short-cuts which can reduce the complexity of cryptanalysis to less than half of exhaustive search were ever reported in the open literature.

The 50% reduction[9] (under a chosen plaintext attack) is based on the following symmetry under complementation:

$$T = \mathrm{DES}(P, K)$$

implies that

$$\bar{T} = \mathrm{DES}(\bar{P}, \bar{K})$$

where $\bar{X}$ is the bit by bit complementation of X. Cryptanalysis can exploit this symmetry if two plaintext/ciphertext pairs (P_1, T_1) and (P_2, T_2) are available with $P_1 = \bar{P}_2$ (or similarly $T_1 = \bar{T}_2$). The attacker encrypts P_1 under all the 2^{55} keys K whose least significant bit is zero. If such a ciphertext T is equal to T_1 then the corresponding key K is likely to be the real key. If $T = \bar{T}_2$ then $\bar{K}$ is likely to be the real key. Otherwise neither K nor $\bar{K}$

can be the real key. Since testing whether $T = \bar{T}_2$ is much faster than an encryption, the computational saving is very close to 50%.

Diffie and Hellman[6] suggested exhaustive search of the entire key space on a parallel machine. They estimate that a VLSI chip may be built which can search one key every microsecond. By building a search machine with a million such chips, all searching in parallel, 10^{12} keys can be searched per second. The entire key space contains about $7 \cdot 10^{16}$ keys and it can be searched in 10^5 seconds which is about a day. They estimate the cost of this machine to be $20-million and the cost per solution to be $5000.

Hellman[8] presented a time memory tradeoff method for a chosen plaintext attack which takes mt words of memory and t^2 operations provided mt^2 equals the number of possible keys (2^{56} for DES). A special case ($m = t$) of this method takes about 2^{38} time and 2^{38} memory, with a 2^{56} preprocessing time. Hellman suggests a special purpose machine which produces 100 solutions per day with an average wait of one day. He estimates that the machine costs about $4-million and the cost per solution is about $1–100. The preprocessing is estimated to take 2.3 years on the same machine.

The *Method of Formal Coding* in which the formal expression of each bit in the ciphertext is found as a XOR sum of products of the bits of the plaintext and the key was suggested in [9]. The formal manipulations of these expressions may decrease the key search effort. Schaumuller-Bichl[16, 17] studied this method and concluded that it requires an enormous amount of computer memory which makes the whole approach impractical.

In 1987 Chaum and Evertse[2] showed that a meet in the middle attack can reduce the key search for DES reduced to a small number of rounds by the following factors:

Number of Rounds	Reduction Factor
4	2^{19}
5	2^{9}
6	2^{2}
7	–

They also showed that a slightly modified version of DES reduced to seven rounds can be solved with a reduction factor of 2. However, they proved that a meet in the middle attack of this kind is not applicable to DES reduced to eight or more rounds.

In their method they look for a set of data bits (J) in a middle round and a set of key bits (I) for which any change of the values of the I bits cannot change the value of the J bits in either directions. Knowing those fixed sets and given several plaintext/ciphertext pairs the following algorithm is used:

1. Try all the keys in which all the key bits in I are zero. Partially encrypt and decrypt a plaintext/ciphertext pair to get the data in the middle round.
2. Discard the keys for which the J bits are not the same under partial encryption/decryption.
3. For the remaining keys try all the possible values of the key bits in I.

This algorithm requires about $2^{56-|I|} + 2^{|I|}$ encryption/decryption attempts.

In the same year, Donald W. Davies[3] described a known plaintext cryptanalytic attack on DES. Given sufficient data, it could yield 16 linear relationships among key bits, thus reducing the size of a subsequent key search to 2^{40}. It exploited the correlation between the outputs of adjacent S boxes, due to their inputs being derived from, among other things, a pair of identical bits produced by the bit expansion operation. This correlation could reveal a linear relationship among the four bits of key used to modify these S box input bits. The two 32-bit halves of the DES result (ignoring IP) receive these outputs independently, so each pair of adjacent S boxes could be exploited twice, yielding 16 bits of key information.

The analysis does not require the plaintext P or ciphertext T but uses the quantity $P \oplus T$ and requires a huge number of random inputs. The S box pairs vary in the extent of correlation they produce so that, for example, the pair S7/S8 needs about 10^{17} samples but pair S2/S3 needs about 10^{21}. With about 10^{23} samples, all but the pair S3/S4 should give results (i.e., a total of 14 bits of key information). To exploit all pairs the cryptanalyst needs about 10^{26} samples. The S boxes do not appear to have been designed to minimize the correlation but they are somewhat better than a random choice in this respect. Since the number of samples is larger than the 2^{64} size of the sample space, this attack is purely theoretical, and cannot be carried out. However, for DES reduced to eight rounds the sample size of 10^{12} or 10^{13} (about 2^{40}) is on the verge of practicality. Therefore, Davies' analysis had penetrated more rounds than previously reported attacks.

During the last decade several cryptosystems which are variants of DES were suggested. Schaumuller-Bichl suggested three such cryptosystems [16, 18]. Two of them (called C80 and C82) are based on the DES structure with the replacement of the F function by nonreversible functions. The third one, called The Generalized DES Scheme (GDES), is an attempt to speed up DES. GDES has 16 rounds with the original DES F function but with a larger block size which is divided into more than two parts. She claims that GDES increases the encryption speed of DES without decreasing its security.

Another variant is the Fast Data Encryption Algorithm (Feal). Feal was designed to be efficiently implementable on an eight bit microprocessor. The first version of Feal[20], called Feal-4, has four rounds. Feal-4 was broken by Den-Boer[4] using a chosen plaintext attack with 100–10000 encryptions. The creators of Feal reacted by introducing a new version, called Feal-8, with eight rounds and additional XORs of the plaintext and the ciphertext with subkeys[19,14]. Both versions were described as cryptographically better than DES in several aspects.

In this paper we describe a new kind of attack that can be applied to many DES-like iterated cryptosystems. This is a chosen plaintext attack which uses only the resultant ciphertexts. The basic tool of the attack is the *ciphertext pair* which is a pair of ciphertexts whose plaintexts have particular differences. The two plaintexts can be chosen at random, as long as they satisfy the difference condition, and the cryptanalyst does not have to know their values. The attack is statistical in nature and can fail in rare instances.

The main results described in this paper are as follows (note that the complexities we quote are based on the number of encryptions needed to create all the necessary pairs on the target machine, while the attacking algorithm itself uses fewer and simpler operations). DES reduced to six rounds was broken in less than 0.3 seconds on a personal computer using 240

Rounds	Complexity
4	2^4
6	2^8
8	2^{16}
9	2^{26}
10	2^{35}
11	2^{36}
12	2^{43}
13	2^{44}
14	2^{51}
15	2^{52}
16	2^{58}

Table 1. Summary of the cryptanalysis of DES.

ciphertexts. DES reduced to eight rounds was broken in less than two minutes on a computer by analysing 15000 ciphertexts chosen from a pool of 50000 candidate ciphertexts. DES reduced to up to 15 rounds is breakable faster than exhaustive search, but DES with 16 rounds still requires 2^{58} steps (which is slightly higher than the complexity of exhaustive search). A summary of the cryptanalytic results on DES reduced to intermediate number of rounds appears in table 1.

Some researchers have proposed to strengthen DES by making all the subkeys Ki independent (or at least to derive them in a more complicated way from a longer actual key K). Our attack can be carried out even in this case. DES reduced to eight rounds with independent subkeys (i.e., with $8 \cdot 48 = 384$ independent key bits which are not compatible with the key scheduling algorithm) was broken in less than two minutes using the same ciphertexts as in the case of dependent subkeys. The full DES with independent subkeys (i.e., with $16 \cdot 48 = 768$ independent key bits) is breakable within 2^{61} steps. As a result, any modification of the key scheduling algorithm cannot make DES much stronger. The attacks on DES reduced to 9–16 rounds are not influenced by the P permutation and the replacement of the P permutation by any other permutation cannot make them less successful. On the other hand, the replacement of the order of the eight DES S boxes (without changing their values) can make DES much weaker: DES with 16 rounds with a particular replaced order is breakable in about 2^{46} steps. The replacement of the XOR operation by the more complex addition operation makes this cryptosystem much weaker. DES with random S boxes is shown to be very easy to break. Even a minimal change of one entry in one of the DES S boxes can make DES easier to break. GDES is shown to be trivially breakable with six encryptions in less than 0.2 seconds, while GDES with independent subkeys is breakable with 16 encryptions in less than 3 seconds.

This attack is applicable also to a wide variety of DES-like cryptosystems. In forthcoming papers we describe several extensions to our new attack. Lucifer reduced to eight rounds can be broken using less than 60 ciphertexts (30 pairs). The Feal-8 cryptosystem can be broken with less than 2000 ciphertexts (1000 pairs) and the Feal-4 cryptosystem can be broken with just eight ciphertexts and one of their plaintexts. As a reaction to our attack on Feal-8, its creators introduced Feal-N[11], with any even number of rounds N. They suggest the use of Feal-N with 16 and 32 rounds. Feal-NX[12] is similar to Feal-N with the extension of the key size to 128 bits. Nevertheless, Feal-N and Feal-NX can be broken for any N $\leq$ 31 rounds faster than exhaustive search.

Differential cryptanalytic techniques are applicable to hash functions, in addition to cryptosystems. For example, the following messages hash to the same value in Merkle's Snefru[10] function with two passes:

- `00000000 00000000 00000000 00000000 00000000 00000000`
  `00000000 00000000 00000000 00000000 00000000 00000000`
- `00000000 f1301600 13dfc53e 4cc3b093 37461661 ccd8b94d`
  `24d9d35f 71471fde 00000000 00000000 00000000 00000000`
- `00000000 1d197f00 2abd3f6f cf33f3d1 8674966a 816e5d51`
  `acd9a905 53c1d180 00000000 00000000 00000000 00000000`
- `00000000 e98c8300 1e777a47 b5271f34 a04974bb 44cc8b62`
  `be4b0efc 18131756 00000000 00000000 00000000 00000000`

and the following two messages hash to the same value in a variant of Miyaguchi's N-Hash[13] function with six rounds:

- `CAECE595 127ABF3C 1ADE09C8 1F9AD8C2`
- `4A8C6595 921A3F3C 1ADE09C8 1F9AD8C2`.

2 Introduction to differential cryptanalysis

Differential cryptanalysis is a method which analyses the effect of particular differences in plaintext pairs on the differences of the resultant ciphertext pairs. These differences can be used to assign probabilities to the possible keys and to locate the most probable key. This method usually works on many pairs of plaintexts with the same particular difference using only the resultant ciphertext pairs. For DES-like cryptosystems the difference is chosen as a fixed XORed value of the two plaintexts. In this introduction we show how these differences can be analyzed and exploited.

We now introduce the following notation:

n_x: An hexadecimal number is denoted by a subscript x (i.e., $10_x = 16$).

X^*, X': At any intermediate point during the encryption of pairs of messages, X and X^* are the corresponding intermediate values of the two executions of the algorithm, and X' is defined to be $X' = X \oplus X^*$.

$P(X)$: The P permutation is denoted by $P(X)$. Note that P as a variable denotes the plaintext.

$E(X)$: The E expansion is denoted by $E(X)$.

$IP(X)$: The initial permutation. In this paper the existence of IP and IP^{-1} are ignored, since they have no cryptanalytic significance in our attack.

P: The plaintext (after the known initial permutation IP) is denoted by P. P^* is the other plaintext in the pair and $P' = P \oplus P^*$ is the plaintexts XOR.

T: The ciphertexts of the corresponding plaintexts P, P^* (before the inverse initial permutation IP^{-1}) are denoted by T and T^*. $T' = T \oplus T^*$ is the ciphertexts XOR.

(L, R): The left and right halves of the plaintext P are denoted by L and R respectively.

(l, r): The left and right halves of the ciphertext T are denoted by l and r respectively.

$a, \ldots, j$: The 32 bit inputs of the F function in the various rounds. See figure 1. Note that $a = R$.

$A, \ldots, J$: The 32 bit outputs of the F function in the various rounds. See figure 1.

Si: The S boxes S1, S2, ..., S8.

Si_{EX}, Si_{KX}, Si_{IX}, Si_{OX}: The input of Si in round X is denoted by Si_{IX} for $X \in \{a, \ldots, j\}$. The output of Si in round X is denoted by Si_{OX}. The value of the six subkey bits entering the S box Si is denoted by Si_{KX} and the value of the six input bits of the expanded data (E(X)) which are XORed with Si_{KX} to form Si_{IX} is denoted by Si_{EX}. The S box number i and the round marker X are optional. For example $S1_{Ea}$ denotes the first six bits of $E(a)$. $S1_{Ka}$ denotes the first six bits of the subkey K1. $S1_{Ia}$ denotes the input of the S box S1 which is $S1_{Ia} = S1_{Ea} \oplus S1_{Ka}$. $S1_{Oa}$ denotes the output of S1 which is $S1_{Oa} = S1(S1_{Ia})$. See figure 2.

Definition 1 An *independent key* is a list of n subkeys which is not necessarily derived from the key scheduling algorithm.

Example 1 DES has $2^{16 \cdot 48} = 2^{768}$ possible independent keys, but only 2^{56} possible keys. Note that every key can be viewed as a special type of an independent key.

Remark To simplify the probabilistic analysis of our attack, we assume that all the subkeys are independent. Attacks on DES with dependent subkeys seem to be just as successful in practice, but their theoretical analysis is much harder.

Let us recall how the DES F function behaves in these terms. The F function takes a 32 bit input and a 48 bit key. The input is expanded (by

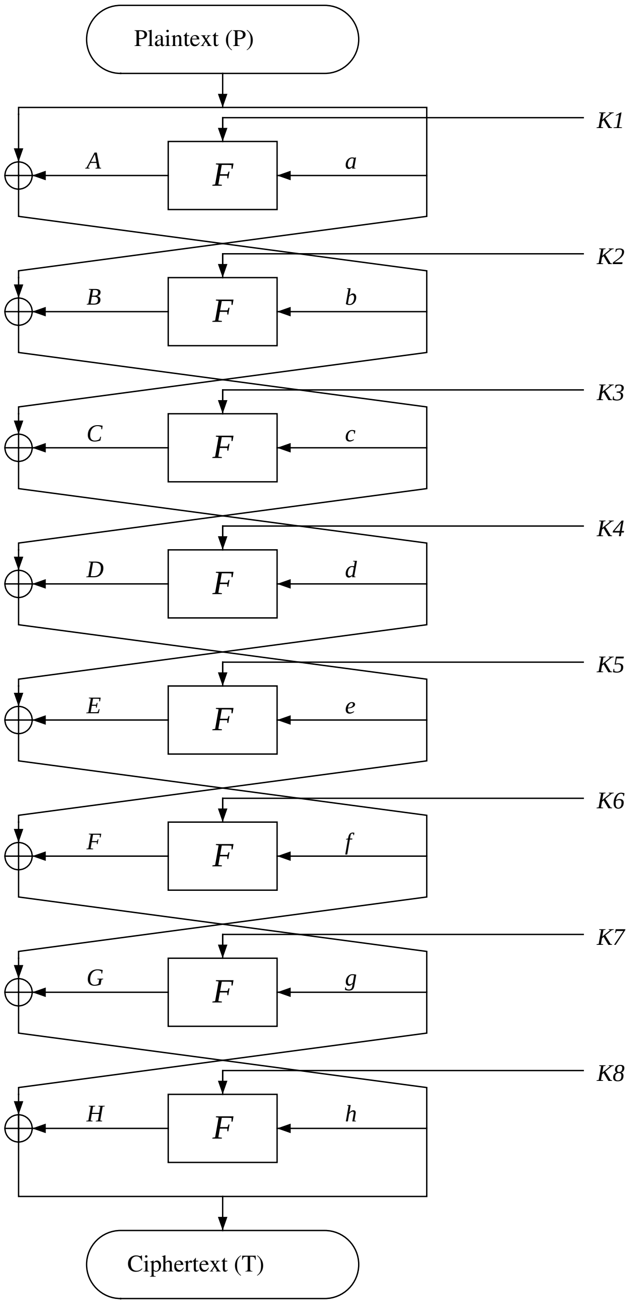


Figure 1. DES reduced to eight rounds.

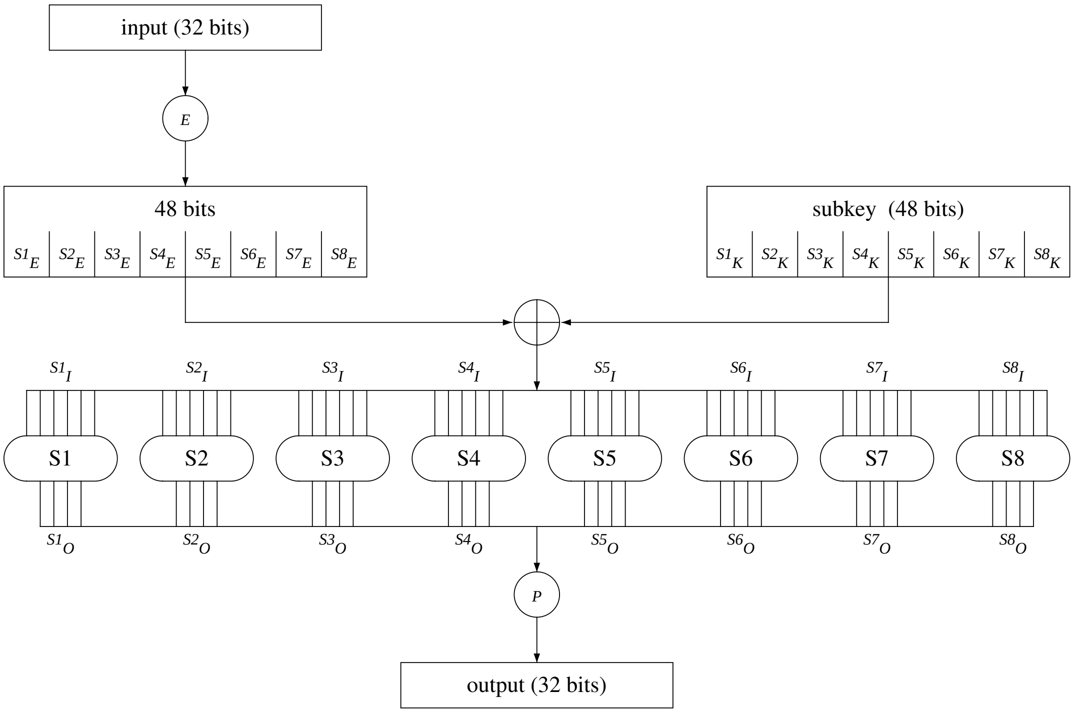


Figure 2. The F function of DES.

the E expansion) to 48 bits and XORed with the key. The result is fed into the S boxes and the resultant bits are permuted.

Given the XOR value of an input pair to the F function it is easy to determine its XOR value after the expansion by the formula:

$$E(X) \oplus E(X^*) = E(X \oplus X^*).$$

The XOR with the key does not change the XOR value in the pair, i.e., the expanded XOR stays valid even after the XOR with the key, by the formula:

$$(X \oplus K) \oplus (X^* \oplus K) = X \oplus X^*.$$

The output of the S boxes is mixed by the P permutation and thus the XOR of the pair after the P permutation is the permuted value of the S boxes output XOR, by the formula:

$$P(X) \oplus P(X^*) = P(X \oplus X^*).$$

The output XOR of the F function is linear in the XOR operation that connects the different rounds:

$$(X \oplus Y) \oplus (X^* \oplus Y^*) = (X \oplus X^*) \oplus (Y \oplus Y^*).$$

The XOR of pairs is thus invariant in the key and is linear in the E expansion, the P permutation and the XOR operation.

The S boxes are known to be non linear. Knowledge of the XOR of the input pairs cannot guarantee knowledge of the XOR of the output pairs. Usually several output XORs are possible. A special case arises when both inputs are equal, in which case both outputs must be equal too. However, a crucial observation is that for any particular input XOR not all the output XORs are possible, the possible ones do not appear uniformly, and some XORed values appear much more frequently than others.

Before we proceed we want to mention the known design principles of the S boxes[1]:

1. No S box is a linear or affine function of its input.
2. Changing one input bit to an S box results in changing at least two output bits.
3. $S(X)$ and $S(X \oplus 001100)$ must differ in at least two bits.
4. $S(X) \neq S(X \oplus 11ef00)$ for any choice of e and f.
5. The S boxes were chosen to minimize the differences between the number of 1's and 0's in any S box output when any single bit is held constant.

In DES any S box has $64 \cdot 64$ possible input pairs, and each one of them has an input XOR and an output XOR. There are only $64 \cdot 16$ possible tuples of input and output XORs. Therefore, each tuple results in average from four pairs. However, not all the tuples exist as a result of a pair, and the existing ones do not have a uniform distribution. Very important properties of the S boxes are derived from the analysis of the tables that summarize this distribution:

Definition 2 A table that shows the distribution of the input XORs and output XORs of all the possible pairs of an S box is called the *pairs XOR distribution table of the S box*. In this table each row corresponds to a particular input XOR, each column corresponds to a particular output XOR and the entries themselves count the number of possible pairs with such an input XOR and an output XOR.

Each line in a pairs XOR distribution table contains 64 possible pairs in 16 different entries. Thus in each line in the table the average of the entries is exactly four.

Example 2 Table 2 is a partial[1] pairs XOR distribution table of S1. S1 itself is described in table 3.

Example 3 The first line of table 2 shows that for the zero input XOR, the output XOR must be zero too, as we noticed above. Also, the different lines in the table have different output XOR distributions.

The following definition deals with pairs XOR distribution tables:

Definition 3 Let X be a six bit value and Y be a four bit value. We say that X *may cause* Y *by an S box* if there is a pair in which the input XOR of the S box equals X and the output XOR of the S box equals Y. If there is such a pair we write $X \rightarrow Y$, and if there is no such pair we say that X *may not cause* Y *by the S box* and write $X \not\rightarrow Y$.

Example 4 Consider the input XOR $S1'_I = 34_x$. It has only eight possible output XORs, while the other eight entries are impossible. The possible output XORs $S1'_O$ are 1_x, 2_x, 3_x, 4_x, 7_x, 8_x, D_x and F_x. Therefore, the input XOR $S1'_I = 34_x$ may cause output XOR $S1'_O = 1_x$ ($34_x \rightarrow 1_x$). Also $34_x \rightarrow 2_x$ and $34_x \rightarrow F_x$. On the other hand $34_x \not\rightarrow 0_x$ and $34_x \not\rightarrow 9_x$.

[1]The full pairs XOR distribution tables of all the S boxes appear in appendix B.

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

Table 2. Partial pairs XOR distribution table of S1.

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

Table 3. S1 table.

Examples 3 and 4 demonstrate that for a fixed input XOR, the possible output XORs do not have a uniform distribution. The following definition extends definition 3 with probabilities.

Definition 4 We say that X *may cause* Y *with probability* p *by an S box* if for a fraction p of the pairs in which the input XOR of the S box equals X, the output XOR equals Y.

Example 5 $34_x \rightarrow 2_x$ results from 16 out of the 64 pairs of S1, i.e., with

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

Table 4. Percentage of the possible entries in the various pairs XOR distribution tables.

probability $\frac{1}{4}$. $34_x \rightarrow 4_x$ results only from two out of the 64 pairs of S1, i.e., with probability $\frac{1}{32}$.

Different distributions appear in different lines of the table. In total between 70% and 80% of the entries are possible and between 20% and 30% are impossible. The exact percentage for each S box is shown in table 4. In various formulas in this paper we approximate the percentage of the possible entries by 80%.

The pairs XOR distribution tables let us find the possible input and output values of pairs given their input and output XORs. The following example shows a simple case:

Example 6 Consider the entry $34_x \rightarrow 4_x$ in the pairs XOR distribution table of S1. Since the entry $34_x \rightarrow 4_x$ has value 2, only two pairs satisfy these XORs. These pairs are duals. If the first pair is $S1_I$, $S1_I^*$ then the other pair is $S1_I^*$, $S1_I$. By looking at table 5 we see that these inputs must be 13_x and 27_x whose corresponding outputs are 6_x and 2_x respectively.

Next we show how to find the key bits using known input pairs and output XOR of an S box in the F function.

Example 7 Consider S1 and assume that the input pair is $S1_E = 1_x$, $S1_E^* = 35_x$ and that the value of the corresponding six key bits is $S1_K = 23_x$. Then

Output XOR ($S1'_O$)	Possible Inputs ($S1_I$)
1	03, $0F$, $1E$, $1F$, $2A$, $2B$, 37, $3B$
2	04, 05, $0E$, 11, 12, 14, $1A$, $1B$, 20, 25, 26, $2E$, $2F$, 30, 31, $3A$
3	01, 02, 15, 21, 35, 36
4	13, 27
7	00, 08, $0D$, 17, 18, $1D$, 23, 29, $2C$, 34, 39, $3C$
8	09, $0C$, 19, $2D$, 38, $3D$
D	06, 10, 16, $1C$, 22, 24, 28, 32
F	07, $0A$, $0B$, 33, $3E$, $3F$

Table 5. Possible input values for the input XOR $S1'_I = 34_x$ by the output XOR (in hexadecimal).

S box input		Possible Keys	
06,	32	07,	33
10,	24	11,	25
16,	22	17,	23
$1C$,	28	$1D$,	29

Table 6. Possible keys for $34_x \to D_x$ by S1 with input 1_x, 35_x (in hexadecimal).

the actual inputs of S1 (after XORing the input and key bits) are $S1_I = 22_x$, $S1^*_I = 16_x$ and the outputs are $S1_O = 1_x$, $S1^*_O = C_x$ respectively. The output XOR is $S1'_O = D_x$.

Assume we know that $S1_E = 1_x$, $S1^*_E = 35_x$ and $S1'_O = D_x$ and we want to find the key value $S1_K$. The input XOR is $S1'_E = S1'_I = 34_x$ regardless of the actual value of $S1_K$. By consulting table 2 we can see that the input to the S box has eight possibilities. These eight possibilities make eight possibilities for the key (by $S_K = S_E \oplus S_I$) as described in table 6. Each line in the table describes two pairs with the same two inputs but with the opposite order. Each pair leads to one key, so each line leads to two keys (which are $S_E \oplus S_I$ and $S_E \oplus S^*_I$). The right key value $S1_K$ must occur in this table.

S box input		Possible Keys	
01,	35	03,	37
02,	36	00,	34
15,	21	17,	23

Table 7. Possible keys for $34_x \rightarrow 3_x$ by S1 with input 21_x, 15_x (in hexadecimal).

Using additional pairs we can get additional candidates for $S1_K$. Lets look at the input pair $S1_E = 21_x$, $S1^*_E = 15_x$ (with the same $S1_K = 23_x$). The inputs to the S box are $S1_I = 2_x$, $S1^*_I = 36_x$ and the outputs are $S1_O = 4_x$, $S1^*_O = 7_x$. The output XOR is $S1'_O = 3_x$. The possible inputs to the S box where $34_x \rightarrow 3_x$ and the corresponding possible keys are described in table 7. The right key must occur in both tables. The only common key values in tables 6 and 7 are 17_x and 23_x. These two values are indistinguishable with this input XOR since $17_x \oplus 23_x = 34_x = S1'_E$, but may become distinguishable by using a pair with a different input XOR value ($S1'_E \neq 34_x$).

The following example is an extension of example 7 to a three-round cryptosystem.

Example 8 Assume we have a ciphertext pair whose plaintext XOR is known and the values of the six bits 33, ..., 38 of the plaintext XOR are zero. The input XOR of the first round is zero in all the bits entering S1 ($S1'_{Ea} = S1'_{Ia} = 0$) and thus the output XOR of S1 in the first round must be zero ($S1'_{Oa} = 0$). The left half of the ciphertext is calculated as the XOR value of the left half of the plaintext, the output of the first round and the output of the third round ($l = L \oplus A \oplus C$). Since the plaintext XOR and the ciphertext XOR are known and the output XOR of S1 in the first round is known as well, the output XOR of S1 in the third round can be calculated. The input pair $S1_{Ec}$, $S1^*_{Ec}$ in the third round is easily extractable from the ciphertext pair.

If the input pair of S1 in the third round is $S1_{Ec} = 1_x$, $S1^*_{Ec} = 35_x$ and the output XOR is $S1'_{Oc} = D_x$ then the value of $S1_{Kc}$ can be found as in example 7 and it must appear in table 6. Using additional pairs we can discard some of the possible values till we get a unique value of $S1_{Kc}$. Since

$S1'_{Ec}$ is not constant, there should not be any indistinguishable values of the subkey.

The following definition extends definitions 3 and 4 for use with the F function:

Definition 5 Let X and Y be 32 bit values. We say that X *may cause* Y *with probability* p *by the* F *function* if for a fraction p of all the possible input pairs encrypted by all the possible subkey values in which the input XOR of the F function equals X, the output XOR equals Y. If $p > 0$ we denote this possibility by $X \rightarrow Y$.

Lemma 1 In DES, if $X \rightarrow Y$ with probability p by the F function then every fixed input pair Z, Z^* with $Z' = Z \oplus Z^* = X$ causes the F function output XOR to be Y by the same fraction p of the possible subkey values.

Proof To prove the lemma it suffices to show the property for each of the S boxes. For each input XOR of the data S'_E there is $S'_I = S'_E$ regardless of S_K. If there are k possible input pairs to the S box with this input XOR that may cause a given output XOR, we can choose precisely k key values $S_K = S_E \oplus S_I$, each taking the fixed input pair S_E, S_E^* to one of the possible input pairs S_I, S_I^* of the S box and thus causing the given output XOR. Thus, the fraction p is held constant for all the input pairs, and therefore equals the average over all the input pairs. ∎

In other iterated cryptosystems this lemma does not necessarily hold. However, we assume that the fraction is very close to p, which is usually the case.

Corollary 1 The probability p of $X \rightarrow Y$ by the F function is the product of p_i in which $X_i \rightarrow Y_i$ by the S boxes S_i ($i \in \{1, \ldots, 8\}$) where $X_1X_2X_3X_4X_5X_6X_7X_8 = E(X)$ and $Y_1Y_2Y_3Y_4Y_5Y_6Y_7Y_8 = P^{-1}(Y)$.

The above discussion about finding the key bits entering S boxes can be extended to find the subkeys entering the F function. The method is as follows:

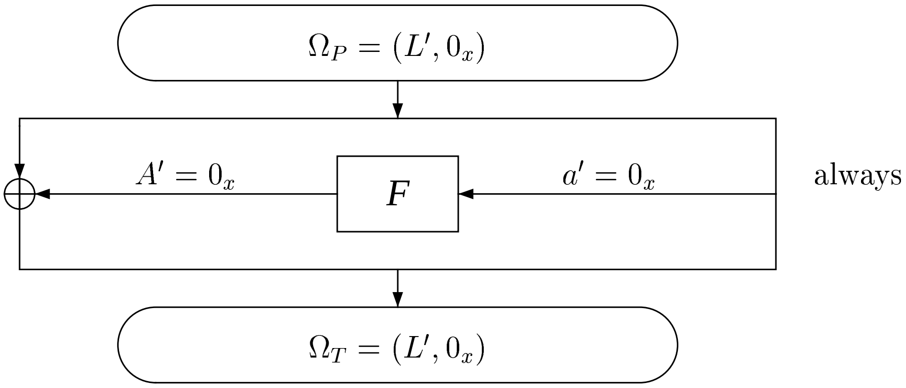
1. Choose an appropriate plaintext XOR.
2. Create an appropriate number of plaintext pairs with the chosen plaintext XOR, encrypt them and keep only the resultant ciphertext pairs.
3. For each pair derive the expected output XOR of as many S boxes in the last round as possible from the plaintext XOR and the ciphertext pair. (Note that the input pair of the last round is known since it appears as part of the ciphertext pair).
4. For each possible key value, count the number of pairs that result with the expected output XOR using this key value in the last round.
5. The right key value is the (hopefully unique) key value suggested by all the pairs.

We are left with the problem of pushing the knowledge of the XORs of the plaintext pairs as many rounds as possible (in step 3) without making them all zeroes. When the XORs of the pairs are zero, i.e., both texts are equal, the outputs are equal too, which makes all the keys equally likely. The pushing mechanism is a statistical characteristic of the cryptosystem which is an extension of the single round analysis. Before we define it formally we give an informal definition and three examples.

Definition 6 (informal) Associated with any pair of encryptions are the XOR value of its two plaintexts, the XOR of its ciphertexts, the XORs of the inputs of each round in the two executions and the XORs of the outputs of each round in the two executions. These XOR values form an *n-round characteristic*. A characteristic has a probability, which is the probability that a random pair with the chosen plaintext XOR has the round and ciphertext XORs specified in the characteristic. We denote the plaintexts XOR of a characteristic by Ω_P and its ciphertexts XOR by Ω_T.

The following example describes a one-round characteristic with probability 1. This is the only one-round characteristic with probability greater than $\frac{1}{4}$. This characteristic is very useful and is applicable in any DES-like cryptosystem.

Example 9 A one-round characteristic with probability 1 is (for any L'):

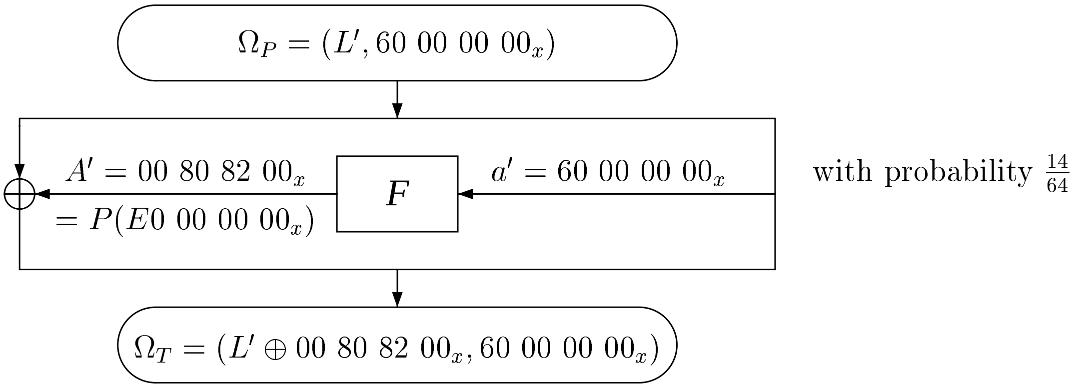


The following example describes a simple one-round characteristic with probability $\frac{14}{64}$.

Example 10 In this one-round characteristic all the S box input XORs except one are zero. One S box input XOR is not zero, and is chosen to maximize the probability that the input XOR may cause the output XOR. Since there are several input bits that are going into two neighboring S boxes by the E expansion we have to ensure that the XORs of these bits are zero. There are only two private bits entering each S box. These bits can have non zero XOR values. The best such probability for S1 is $\frac{14}{64}$ (i.e., there is an entry that contains 14 pairs that does not cause the input of the neighboring S2 or S8 to be non zero). Thus, it is easy to get a one-round characteristic with probability $\frac{14}{64}$ which is:

$$\begin{array}{rll} S1: & 0C_x \rightarrow E_x & \text{with probability } \frac{14}{64} \\ S2, \ldots, S8: & 00_x \rightarrow 0_x & \text{always.} \end{array}$$

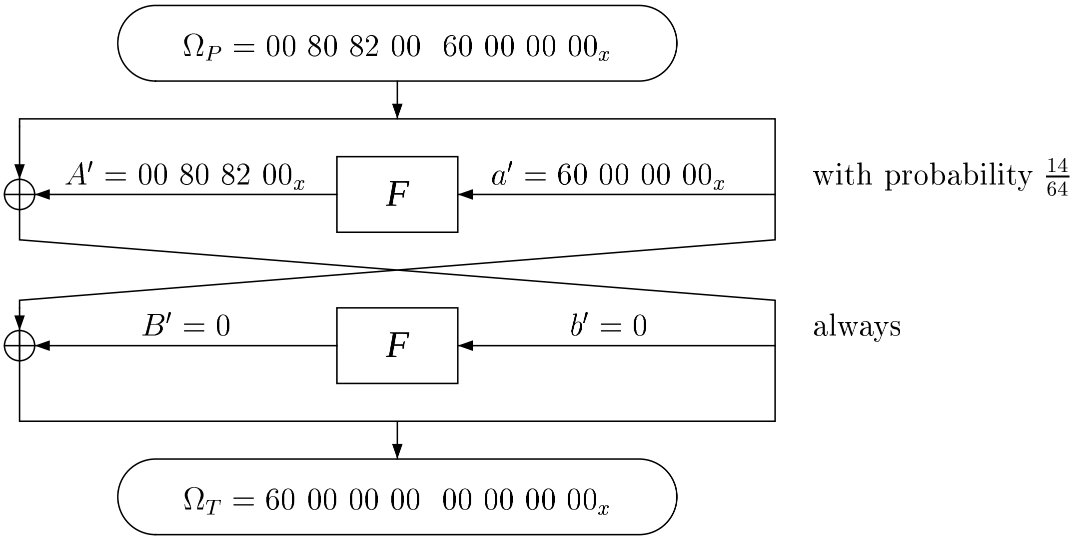
This characteristic can also be written (for any L') as:



One-round characteristics with probability $\frac{1}{4}$ are possible using non zero input XOR in S2 or S6.

The following example describes a two-round characteristic which is easily obtained by concatenating the two one-round characteristics that are described in examples 10 and 9:

Example 11 A two-round characteristic with probability $\frac{14}{64}$:



We can now formulate the exact definition of a characteristic:

Definition 7 An *n-round characteristic* is a tuple $\Omega = (\Omega_P, \Omega_\Lambda, \Omega_T)$ where Ω_P and Ω_T are m bit numbers and Ω_Λ is a list of n elements $\Omega_\Lambda = (\Lambda_1, \Lambda_2, \ldots, \Lambda_n)$, each of which is a pair of the form $\Lambda_i = (\lambda_I^i, \lambda_O^i)$ where λ_I^i and λ_O^i are $m/2$ bit numbers and m is the block size of the cryptosystem. A characteristic satisfies the following requirements:

$$\begin{aligned}
\lambda_I^1 &= \text{the right half of } \Omega_P \\
\lambda_I^2 &= \text{the left half of } \Omega_P \oplus \lambda_O^1 \\
\lambda_I^n &= \text{the right half of } \Omega_T \\
\lambda_I^{n-1} &= \text{the left half of } \Omega_T \oplus \lambda_O^n
\end{aligned}$$

and for every i s.t. $2 \le i \le n-1$:

$$\lambda_O^i = \lambda_I^{i-1} \oplus \lambda_I^{i+1}.$$

Definition 8 A *right pair with respect to an n-round characteristic* $\Omega = (\Omega_P, \Omega_\Lambda, \Omega_T)$ *and an independent key* K is a pair for which $P' = \Omega_P$ and for the first n rounds of the encryption of the pair using the independent key K the input XOR of the i^{th} round equals λ_I^i and the output XOR of the F function equals λ_O^i. Every pair which is not a right pair with respect to the characteristic and the independent key is called a *wrong pair with respect to the characteristic and the independent key*. Throughout this paper we refer them shortly by *right pair* and *wrong pair*.

Definition 9 The *concatenation* of an n-round characteristic $\Omega^1 = (\Omega_P^1, \Omega_\Lambda^1, \Omega_T^1)$ with an m-round characteristic $\Omega^2 = (\Omega_P^2, \Omega_\Lambda^2, \Omega_T^2)$ where Ω_T^1 equals the swapped value of the two halves of Ω_P^2, is the characteristic $\Omega = (\Omega_P^1, \Omega_\Lambda, \Omega_T^2)$ where Ω_Λ is the concatenation of the lists Ω_Λ^1 and Ω_Λ^2.

The following definitions, lemma, conclusion and theorem deal with the probability of characteristics:

Definition 10 *Round* i *of a characteristic* Ω *has probability* p_i^Ω if $\lambda_I^i \to \lambda_O^i$ with probability p_i^Ω by the F function.

Definition 11 *An n-round characteristic Ω has probability p^Ω* if p^Ω is the product of the probabilities of its n rounds:

$$p^\Omega = \prod_{i=1}^{n} p_i^\Omega.$$

Note that by definitions 9 and 11 the probability of a characteristic Ω which is the concatenation of the characteristic Ω^1 with the characteristic Ω^2 is the product of their probabilities: $p^\Omega = p^{\Omega^1} \cdot p^{\Omega^2}$. As a result, every n-round characteristic can be described as the concatenation of n one-round characteristics with probability which is the product of the one-round characteristics probabilities.

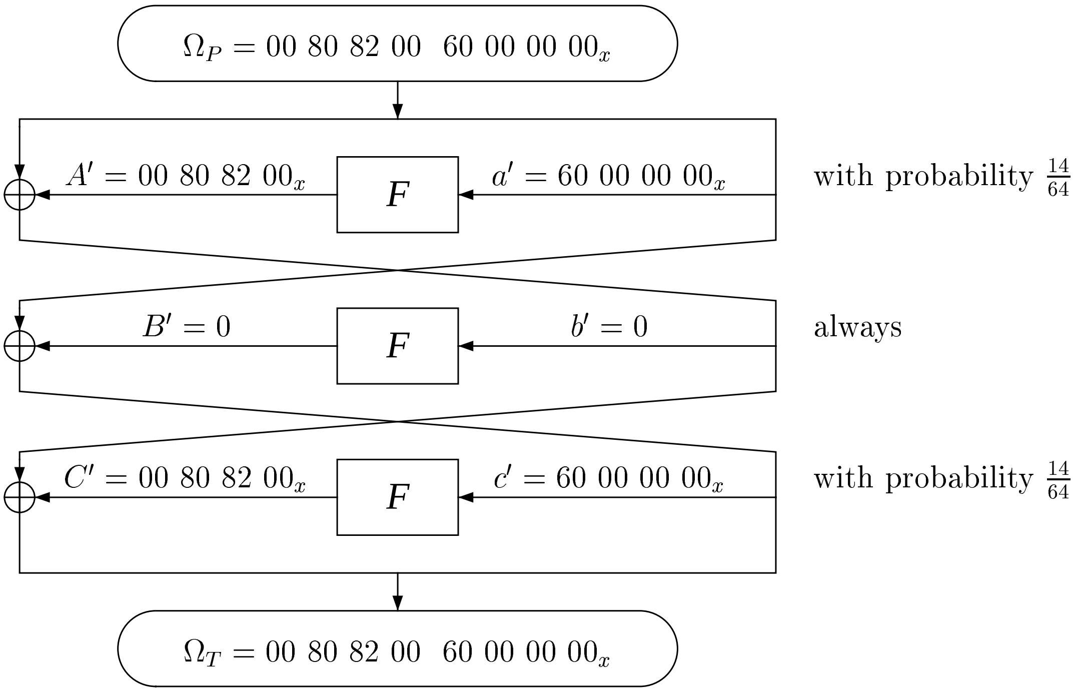
Theorem 1 The formally defined probability of a characteristic $\Omega = (\Omega_P, \Omega_\Lambda, \Omega_T)$ is the actual probability that any fixed plaintext pair satisfying $P' = \Omega_P$ is a right pair when random independent keys are used.

Proof The probability of any fixed plaintext pair satisfying $P' = \Omega_P$ to be a right pair is the probability that at all the rounds i: $\lambda_I^i \rightarrow \lambda_O^i$. The probability at each round is independent of its exact input (as proved in lemma 1) and independent of the action of the previous rounds (since the independent keys completely randomize the inputs to each S box, leaving only the XOR value fixed). Therefore, the probability of a pair to be a right pair is the product of the probabilities of $\lambda_I^i \rightarrow \lambda_O^i$, which was defined above as the probability of the characteristic. ■

For practical purposes the significant probability with respect to a characteristic is the probability that a pair whose plaintext XOR equals the characteristic's plaintext XOR is a right pair using a fixed key (the one we try to find). This probability is not constant for all the keys (as we show later in this paper in a special case). However, we assume that the characteristic's probability is a very good approximation of it, which is usually the case.

After this formal discussion we show a three-round characteristic:

Example 12 An extension to three rounds of the characteristic described in example 11 can be achieved by concatenating it with the characteristic of example 10. Thus a three-round characteristic with probability $\left(\frac{14}{64}\right)^2 \approx 0.05$ is:

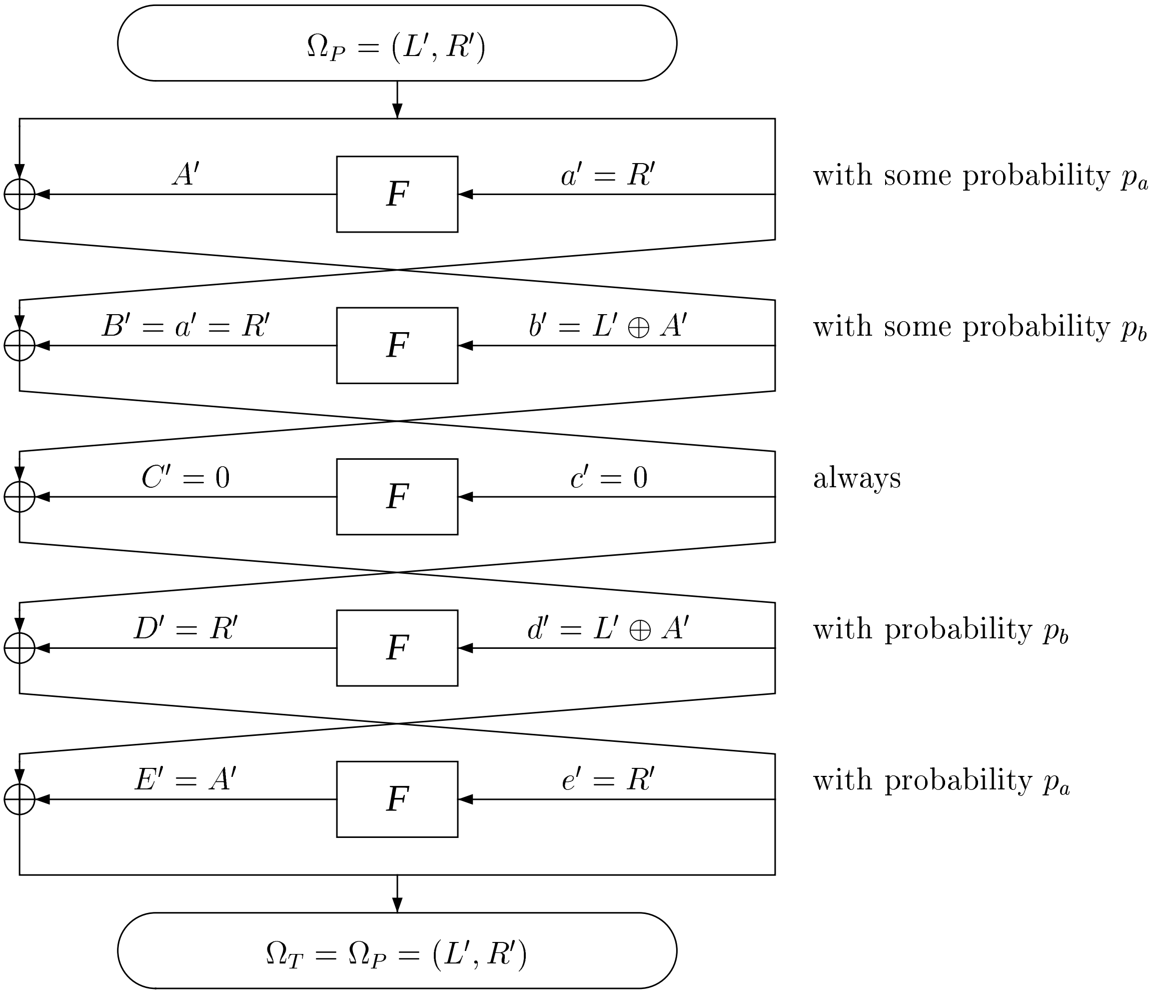


where in the fourth round $d' = b' \oplus C' = C' = A'$. We see that when the plaintexts differ in the five specified bit locations, with probability about 0.05 there is a difference of only three bits at the input of the fourth round. After the bit expansion, five S boxes have non zero input XOR and three have zero input XORs and thus zero output XORs. In this case it is possible to deduce 12 bits of e' by $e' = c' \oplus D'$.

This structure of three rounds with a zero input XOR in the middle round is very useful and forms the best possible probability for three-round characteristics[2]. A similar structure can be used in five-round characteristics. The middle round has zero input and output XORs and there is a symmetry

[2]Since less than two differing S boxes are impossible and there are characteristics of this structure with two differing S boxes, each with the best possible probability $(\frac{1}{4})$.

around it, i.e.:



Where in the sixth round $f' = d' \oplus E' = b' \oplus A' = L'$. The existence of a string $b' \to a' \to A'$ ensures the existence of such a five-round characteristic. The characteristic's probability is quite low since three S box inputs must differ in both rounds $b' \to a'$ and $a' \to A'$, and six in the whole five-round characteristic. The best probability for an S box is $\frac{16}{64} = \frac{1}{4}$. This limits the five-round characteristic's probability to be lower than or equal to $\left(\frac{1}{4}\right)^6 = \frac{1}{4096}$. In fact, the best known five-round characteristic has probability about $\frac{1}{10486}$.

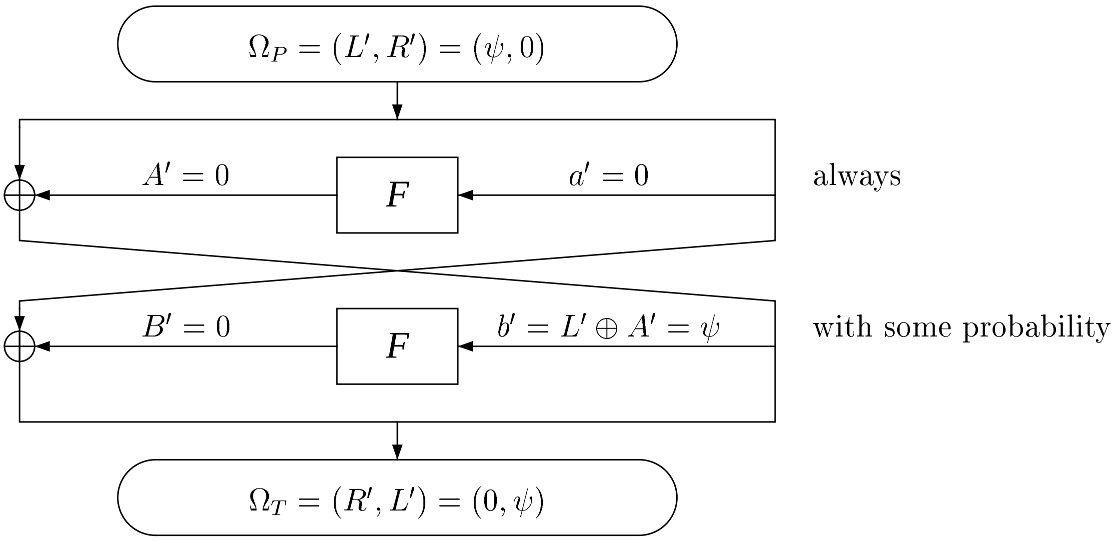
Among the most useful characteristics are those that can be iterated.

Definition 12 A characteristic $\Omega = (\Omega_P, \Omega_\Lambda, \Omega_T)$ is called an *iterative characteristic* if the swapped value of the two halves of Ω_P equals Ω_T.

We can concatenate an iterative characteristic to itself any number of times and can get characteristics with an arbitrary number of rounds. The advantage of iterative characteristics is that we can build an n-round characteristic for any large n with a fixed reduction rate of the probability for each additional round, while in non iterative characteristics the reduction rate of the probability usually increases due to the avalanche effect.

There are several kinds of iterative characteristics but the simplest ones are the most useful. These characteristics are based on a non zero input XOR to the F function that may cause a zero output XOR (i.e., two different inputs yield the same output). This is possible in DES if at least three neighboring S boxes differ in the pair (this phenomena is also described in [5,1]). The structure of these characteristics is described in the following example.

Example 13 If the input XOR of the F function is marked by ψ, s.t. $\psi \to 0$ then we have the following iterative characteristic:



The best such characteristic has probability about $\frac{1}{234}$. A five-round characteristic based on this iterative characteristic has probability about $\frac{1}{55000}$.

The statistical behavior of most characteristics does not allow us to look for the intersection of all the keys suggested by the various pairs as we did in example 7, since the intersection is usually empty: the wrong pairs do not necessarily list the right key as a possible value. However, we know that the right key value should result from all the right pairs which occur (approximately) with the characteristic's probability. All the other possible key values are fairly randomly distributed: the expected XOR value (which is usually not the real value in the pair) with the known ciphertext pair can cause any key value to be possible, and even the wrong key values suggested by the right pairs are quite random. Consequently, the right key appears with the characteristic's probability (from right pairs) plus other random occurrences (from wrong pairs). To find the key we just have to count the number of occurrences of each of the suggested keys. The right key is likely to be the one that occurs most often.

Each characteristic lets us look for a particular number of bits in the subkey of the last round (all the bits that enter some particular S boxes). The most useful characteristics are those which have a maximal probability and a maximal number of subkey bits whose occurrences can be counted. Yet, it is not necessary to count on all the possible subkey bits. The advantages of counting on all the possible subkey bits are the good identification of the right key value and the small amount of data needed. However, counting the number of occurrences of all the possible values of a large number of bits usually demands huge memory which can make the attack impractical. We can count on a smaller number of subkey bits entering a smaller number of S boxes, and use all the other S boxes only to identify and discard those wrong pairs in which the input XORs in such S boxes cannot cause the expected output XORs. Since about 20% of the entries in the pairs XOR distribution tables of the S boxes are impossible, about 20% of the wrong pairs can be discarded by each S box before they are actually counted.

The following definition and lemma give us a tool to evaluate the usability of a counting scheme based on a characteristic:

Definition 13 The ratio between the number of right pairs and the average count in a counting scheme is called the *signal to noise ratio of the counting scheme* and is denoted by S/N.

To find the right key in a counting scheme we need a high probability characteristic and enough ciphertext pairs to guarantee the existence of several right pairs. This means that for a characteristic with probability $\frac{1}{10000}$ we need several tens of thousands of pairs. How many pairs we need depends on the probability of the characteristic, the number of key bits that we count on and the level of identification of wrong pairs that can be discarded before the counting. If we are looking for k key bits then we count the number of occurrences of 2^k possible key values in 2^k counters. The counters contain an average count of $\frac{m \cdot \alpha \cdot \beta}{2^k}$ counts where m is the number of pairs, α is the average count per counted pair and β is the ratio of the counted to all pairs (i.e., counted and discarded). The right key value is counted about $m \cdot p$ times using the right pairs where p is the characteristic's probability, plus the random counts estimated above for all the possible keys. The signal to noise ratio of a counting scheme is therefore:

$$S/N = \frac{m \cdot p}{m \cdot \alpha \cdot \beta / 2^k} = \frac{2^k \cdot p}{\alpha \cdot \beta}.$$

A simple corollary of this formula is that the signal to noise ratio of a counting scheme is independent of the amount of pairs used in the scheme. Another corollary is that different counting schemes based on the same characteristic but with a different number of subkey bits have different S/N.

Usually we relate the number of pairs needed by a counting scheme to the number of the right pairs needed. The number of right pairs needed is mainly a function of the signal to noise ratio. When the S/N is high enough, only a few occurrences of right pairs are needed to uniquely identify the right value of the subkey bits. We observed experimentally that when the S/N is about 1–2, about 20–40 occurrences of right pairs are sufficient. When the S/N is much higher even 3–4 right pairs are usually enough. When the S/N is much smaller the identification of the right value of the subkey bits requires an unreasonably large number of pairs.

In many attacks we use several simultaneous characteristics. In order to minimize the number of ciphertexts needed, we can pack them into more economical structures.

Definition 14 A *quartet* is a structure of four ciphertexts that simultaneously contains two ciphertext pairs of one characteristic and two ciphertext pairs of a second characteristic. An *octet* is a structure of eight ciphertexts that simultaneously contains four ciphertext pairs of each of three characteristics.

Example 14 The following four plaintexts form a quartet (where ψ_1 and ψ_2 are the plaintext XORs of the characteristics):

1. A random plaintext P.
2. $P \oplus \psi_1$.
3. $P \oplus \psi_2$.
4. $P \oplus \psi_1 \oplus \psi_2$.

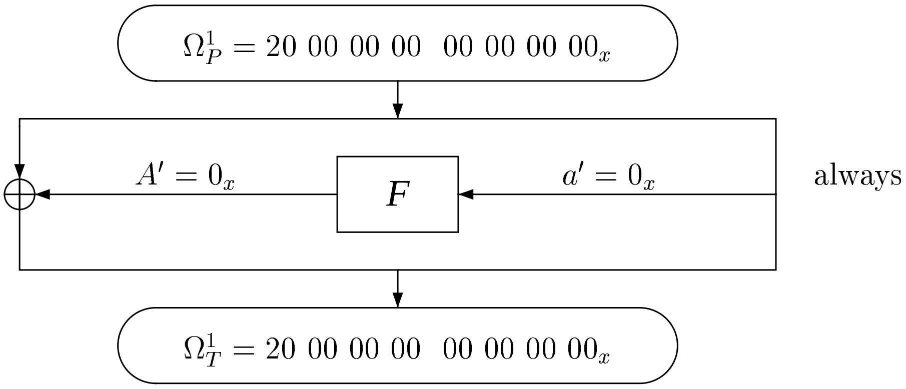
The two pairs of the first characteristic are the pairs labeled (1, 2) and (3, 4) and the two pairs of the second characteristic are the pairs labeled (1, 3) and (2, 4).

The use of these structures can be done in two ways. When an attack uses n pairs of each one of two characteristics we can use $n/2$ quartets which contain the same information as each of the n pairs of each characteristic. Thus, we save half the data. Using three characteristics we can save 2/3 of the data. The other approach is used when an attack can simultaneously use two characteristics while counting the same bits. Then we can divide the data so that half of the pairs are based on the first characteristic and the other half on the second. When quartets can be used we can save half the data, and when octets can be used we can save 2/3 of the data.

3 DES reduced to four rounds

In section 2 we defined the notions of pairs and characteristics. In this section we describe how it can be used to cryptanalyze DES reduced to four rounds. This cryptanalysis is quite simple since it uses a characteristic with probability 1, but it serves as a good introductory example to the method of differential cryptanalysis.

In this attack we use the following one-round characteristic Ω^1 with probability 1 which is an instance of the characteristic described in example 9:



where in the second round $b' = L' \oplus A' = 20\ 00\ 00\ 00_x$.

In the first round the characteristic has $a' = 0 \rightarrow A' = 0$ with probability 1. The single bit difference between the two plaintexts starts to play a role in the second round in S1. Since the inputs to S1 differ only in one bit, at least two output bits must differ. Typically such two bits enter three S boxes in the third round ($c' = a' \oplus B' = B'$), where there is a difference of one bit in each S box input. Thus, about six output bits differ at the third round. These bits are XORed with the known difference of the input of S1 in the second round ($d' = b' \oplus C'$), making a difference of about seven bits in the input of the fourth round and about 11 bits in the entries of the S boxes (due to the E expansion). Such an avalanche makes it very likely that the input of all the S boxes differ at the fourth round. Even if an input of an S box does not differ in one pair it can differ in another pair and the exact value of d' is usually different for every pair.

The 28 output XOR bits of S2, ..., S8 in B' must be equal to zero since their input XORs are zero. Since $a' \oplus B' = c' = D' \oplus l'$ (see figure 3) then

$$D' = a' \oplus l' \oplus B'. \tag{1}$$

When the ciphertext pair values T and T^* are known then d and d^* are known to be their right halves (by $d = r$). Since a', l' and the 28 bits of B' are known, the corresponding 28 bits of D' are known as well by equation 1.

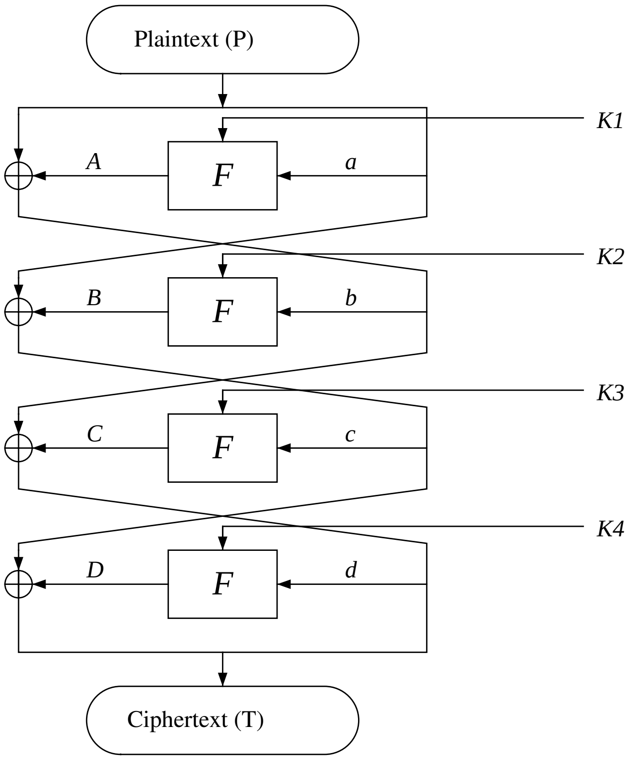


Figure 3. DES reduced to four rounds.

These 28 bits are the output XORs of S boxes S2, ..., S8. Thus, we know the values S_{Ed}, S^*_{Ed} and S'_{Od} of seven S boxes in the fourth round.

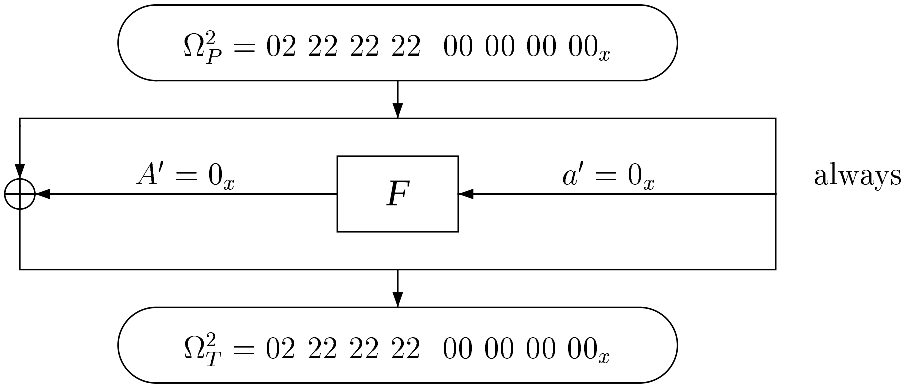
Given the encrypted pairs we use a separate counting procedure for each one of the seven S boxes in the fourth round. We try all the 64 possible values of S_{Kd} and check if $S_{Id} = S_{Ed} \oplus S_{Kd}$ and $S^*_{Id} = S^*_{Ed} \oplus S_{Kd}$ yield

$$S_{Od} \oplus S^*_{Od} = S'_{Od}.$$

For each key we count the number of pairs for which the test succeeds. The right key value is suggested by all the pairs since we use a characteristic with probability 1 which causes all the pairs to be right pairs. The other 63 key values may occur in some of the pairs. It is unlikely that a value occurs in all the pairs for which S'_I are different and S'_O are different. In rare cases when more than one key value is suggested by all the pairs a few additional pairs can be tried, or the analysis of the other key bits can be done in parallel for all the surviving candidates.

So far we have found $7 \cdot 6 = 42$ bits of the subkey of the last round (K4). If the subkeys are calculated via the DES key scheduling algorithm these are 42 actual key bits out of the DES 56 key bits, and 14 key bits are still missing. One can now try all the 2^{14} possibilities of the missing bits and decrypt the given ciphertexts using the resulting keys. The right key should satisfy the known plaintext XOR value for all the pairs, but the other $2^{14}-1$ values have only probability 2^{-64} to satisfy this condition.

Some researchers have proposed to strengthen DES by making all the subkeys Ki independent (or at least to derive them in a more complicated way from a longer actual key K). Our attack can be carried out even in this case. To find the six missing bits of K4 and to find K3 we use another plaintext XOR value with the following characteristic Ω^2:



Where in the second round $b' = L' \oplus A' = 02\ 22\ 22\ 22_x$.

The value of $S1'_{Eb}$ is zero. Thus, $S1'_{Ob} = 0$. As above we find $S1'_{Od}$ using equation 1 and similarly we can find the corresponding six key bits $S1_{Kd}$.

Now we know the complete fourth round subkey K4. Using K4 we partially decrypt all the given ciphertexts by "peeling off" the effect of the last round. As a result we remain with a three-round cryptosystem. In this cryptosystem the second P' value lets us calculate the third round subkey K3. The inputs to the third round c and c^* are known as halves of the ciphertexts of the three-round cryptosystem. The input XOR c' is easily calculated. The output XOR C' is $C' = b' \oplus d'$ where b' and d' are known. The counting method is used again to count the number of occurrences of the possible keys of all the eight S boxes at the third round. The values that are counted for all the pairs are likely to the right key values. As a result the complete K3 is found with high probability.

The P' values used above are insufficient to find a unique K2 since the S'_{Eb} are constant for all the pairs, and thus the right key values are indistinguishable from the alternative key values obtained by XORing them with S'_{Eb}. Although we can find these two possibilities for each S box, i.e., 2^8 possibilities for K2, we cannot use the above XOR values to find K1 since in both XOR values there is $R' = 0$ and thus $a' = 0$ and $A' = 0$. Note that $a' = 0 \rightarrow A' = 0$ happens regardless of the key and thus all the possible values of K1 are equally likely using these XOR values. To solve this problem we have to use an additional characteristic which has a non zero input XOR for all the S boxes of the first round. In addition we want to be able to distinguish the key values of all the S boxes so we choose two characteristics Ω^3 and Ω^4. These characteristics can be chosen arbitrarily under the following two conditions:

- $S'_{Ea} \neq 0$ for all the S boxes using either Ω^3_P or Ω^4_P.
- For every particular S box S'_{Ea} of the characteristic Ω^3_P is different from S'_{Ea} of Ω^4_P.

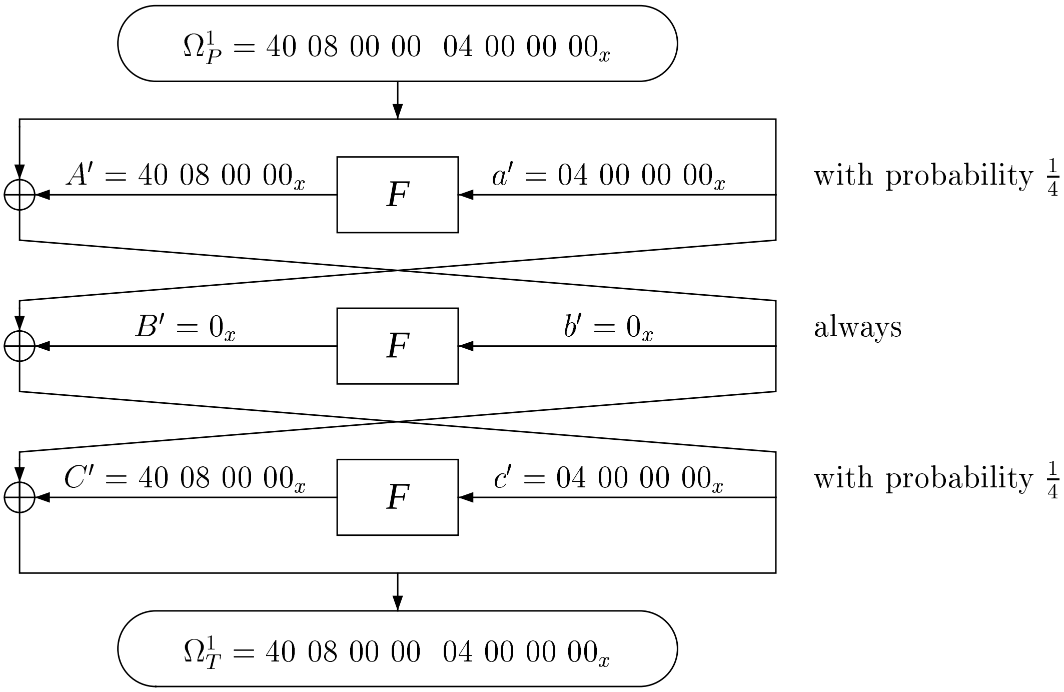
Then b and b^* are known by decryption of the third round and B' is known by $B' = a' \oplus c' = R' \oplus c'$. The counting method is used to find K2. This time it has to use the appropriate R' value for each pair. Now a, a^* and a' are known by decryption of the second round and A' is known by $A' = L' \oplus b'$. The counting method finds K1. Using K1, K2, K3 and K4 we can decrypt the original ciphertexts to get the corresponding plaintexts and then verify their plaintext XOR values. If we find only one possibility for all the subkeys the verification must succeed. If several possibilities are found then only one of them is likely to be verified successfully, and thus the right key can be identified.

Typically, 16 encryptions are sufficient for this attack. These 16 encryptions contain eight pairs of the characteristic Ω^1, eight pairs of Ω^2, four pairs of Ω^3 and four pairs of Ω^4. In order not to increase the amount of data needed we use two octets that occupy four pairs of each of three plaintext XOR.

4 DES reduced to six rounds

The cryptanalysis of DES reduced to six rounds is more complex than the cryptanalysis of the four round version. We use two statistical characteristics with probability $\frac{1}{16}$, and choose the key value that is counted most often. Each one of the two characteristics lets us find the 30 key bits of K6 which are used at the input of five S boxes in the sixth round, but three of the S boxes are common so the total number of key bits found by the two characteristics is 42. The other 14 key bits can be found later by means of exhaustive search or by a more careful counting on the key bits entering the eighth S box in the sixth round.

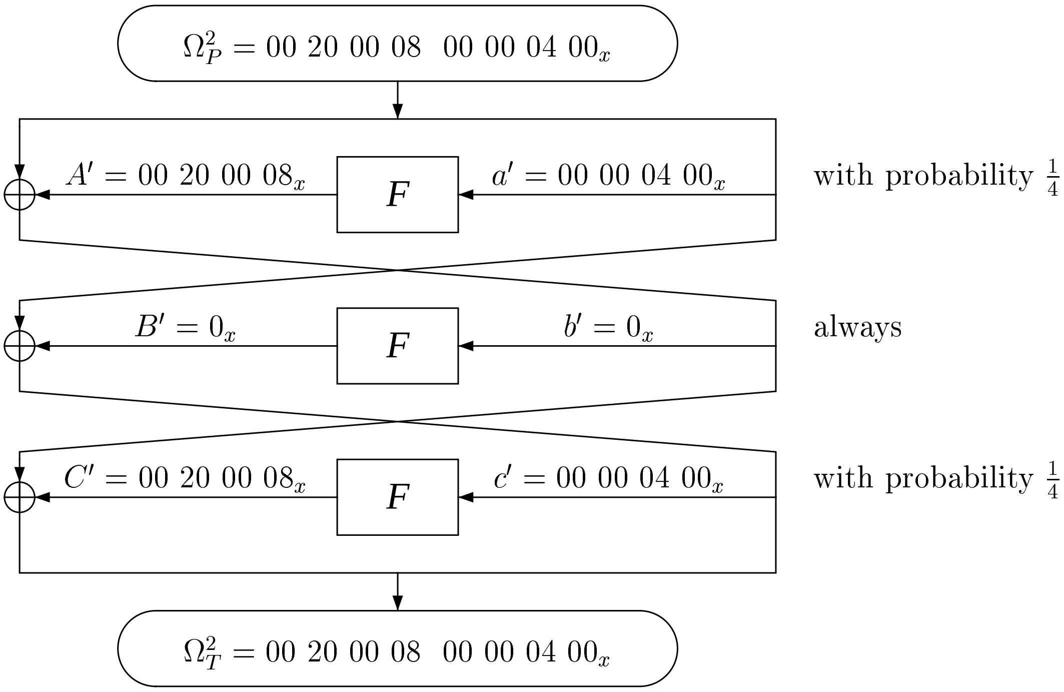
The first characteristic Ω^1 is:



Where in the fourth round $d' = b' \oplus C' = 40\ 08\ 00\ 00_x$.

Five S boxes in the fourth round (S2, S5, ..., S8) have zero input XORs ($S'_{Ed} = 0$) and thus their output XORs are zero ($S'_{Od} = 0$). The corresponding output XORs in the sixth round can be found by $F' = c' \oplus D' \oplus l'$. Since the right key value is not suggested by all the pairs (due to the probabilistic nature of the characteristic), we cannot use a separate counting procedure for the subkey bits entering each S box. In order to increase the S/N we should simultaneously count on subkey bits entering several S boxes. The best approach is to count on all the 30 countable subkey bits together, which maximizes the probability that the right key value is the one counted most often. A straightforward implementation of this method requires 2^{30} counters, which is impractical on most computers. However, the improved counting procedure described at the end of this section achieves exactly the same result with much smaller memory.

The same efficient algorithm is used to find the 30 key bits of S1, S2, S4, S5 and S6 using the second characteristic Ω^2 which is:



Where in the fourth round $d' = b' \oplus C' = 00\ 20\ 00\ 08_x$.

Again, five S boxes in the fourth round (S1, S2, S4, S5 and S6) have zero input XORs. The computed key values of the common S boxes S2, S5 and S6 should be the same in both calculations (otherwise we should analyze more pairs or consider additional candidate keys with almost maximal counts). If this test is successful, we have probably found 42 bits of K6.

DES has 56 key bits. 14 of them are still missing. The simplest way to find them is to search all the 2^{14} possibilities for the expected plaintext XOR value of the decrypted ciphertexts. A faster way is to start looking for the six missing bits of K6 which enter S3 (the other eight key bits occur only in other subkeys). At first we use our partial knowledge of the key to filter the given pairs. For each pair we check if at the five S boxes having $S'_{Ed} = 0$ by the characteristic, the value of S'_{Of} obtained by f and f^* and the known key bits form the expected value from $F' = c' \oplus D' \oplus l'$. If not then this cannot be a right pair. Otherwise it is almost certainly a right pair (since

Into S box #	e bits S_{Ee}	Key bits S_{Ke}
S1	`++++++`	`3+..++`
S2	`++3+++`	`+3+333`
S3	`++++++`	`++++++`
S4	`++++3+`	`++..++`
S5	`3+++++`	`+++.++`
S6	`++++3+`	`+.+.++`
S7	`3+++++`	`+++.++`
S8	`++3+++`	`++++++`

Table 8. Known bits at the fifth round.

the condition can be satisfied accidentally only with probability 2^{-20}). For the remainder of the cryptanalysis we use only the (roughly) $\frac{1}{16}$ of the pairs which are believed to be the right pairs. This filtration greatly improves the signal to noise ratio of the following scheme, which otherwise would be impractical.

Table 8 describes the known bits of the key and the input to the F function at the fifth round assuming we know the 42 key bits. The digit '`3`' means that the bit depends on the exact value of the missing key bits that enter S3 in the sixth round. '`+`' means that it depends only on known key bits. Eight key bits are not used at all in the subkey K6, and are marked by '`.`'. This table shows that by guessing the six missing bits of K6 we can verify its correctness by calculating e and e^* for each right pair by a single round decryption with K6 and by verifying that the values of $S2'_{Oe}$, $S3'_{Oe}$ and $S8'_{Oe}$ (for which all the input and key bits are known) are as expected by $E' = d' \oplus f'$. Furthermore, we can verify that there are values of the missing key bits (for each S box separately) such that the other S boxes output XORs are as expected. The verification of most of the 64 possibilities of the six missing bits of K6 should fail, and with high probability only one possibility survives. This value completes K6. Only eight key bits are missing now. They can be found by trying all the 256 possibilities, or by applying a similar analysis to key bits that enter S boxes in the fifth round.

How much data is needed? The signal to noise ratio of the first part of the algorithm (which finds 30 key bits) is

$$S/N = \frac{2^{30} \cdot \frac{1}{16}}{4^5} = 2^{30-4-10} = 2^{16}.$$

The S/N is high and thus only 7–8 right pairs of each characteristic are needed. Since the characteristics' probability is $\frac{1}{16}$, we need about 120 pairs of each characteristic for the analysis. The S/N of the later part is

$$S/N = \frac{2^6 \cdot 1}{4} = 16.$$

This is lower, but we do not care since we can almost certainly identify and use only the 7–8 right pairs from the first part (while eliminating most of the noise) and intersect the sets of possible key values. To reduce the number of ciphertexts needed we use quartets which combine the two characteristics. As a result only 240 ciphertexts (representing 120 pairs of each characteristic) are needed for the complete cryptanalysis.

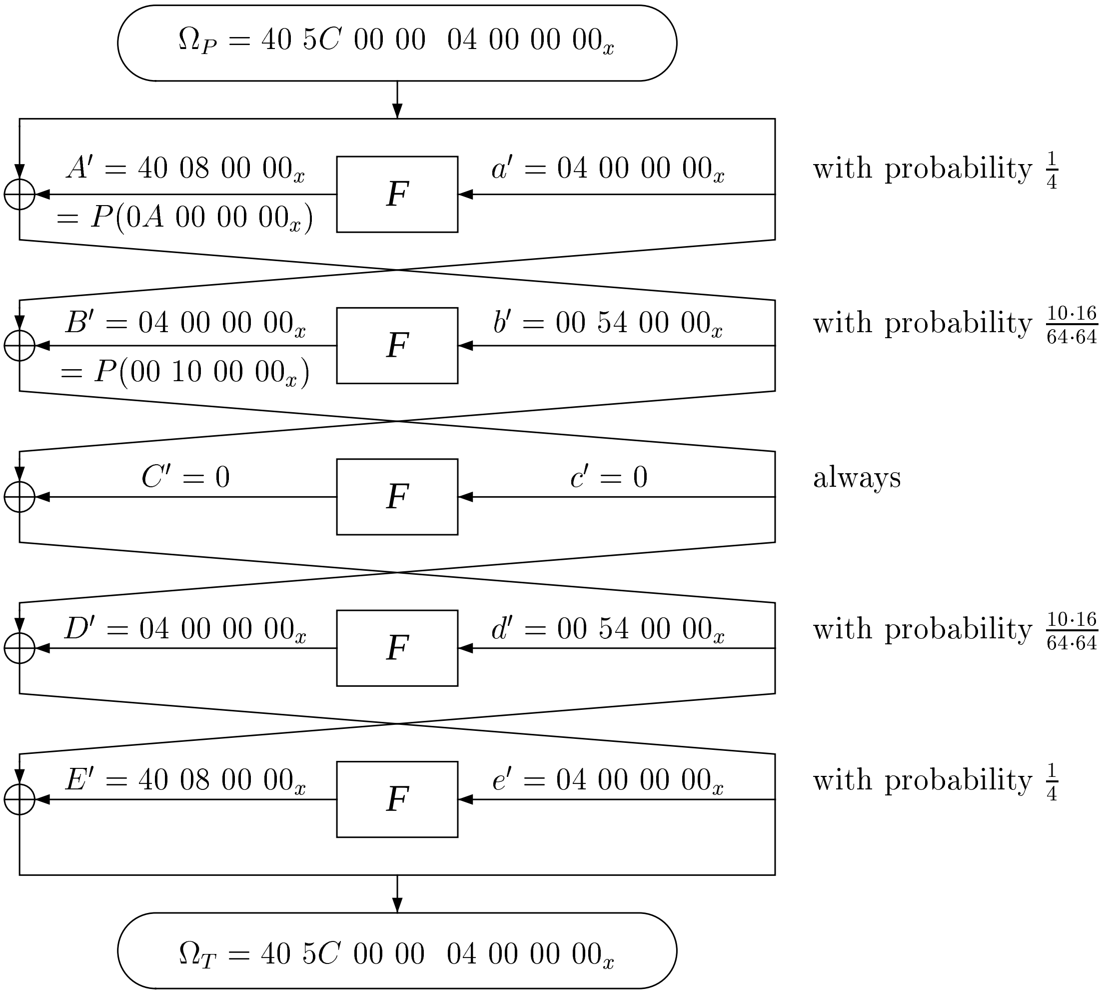
In order to decrease the amount of memory needed in the first part of this attack we devised an equivalent but faster counting algorithm that uses negligible memory and can count on all the countable subkey bits simultaneously. This algorithm can be used in any counting scheme that needs a huge memory but analyses a relatively small number of pairs (after filtering out all the identifiable wrong pairs). The idea behind this algorithm is to describe the pairs and the possible key values by a graph. In this graph each pair is a vertex and every two pairs which suggest a common key value have a connecting edge labeled by this value. Thus, each key value forms a clique which contains all its suggesting pairs. The largest clique corresponds to the key value which is counted by the largest number of pairs. In our implementation, for each of the five S boxes we keep a bit mask of 64 bits, one bit for each possible key. Given the values of S_E, S_E^* and S'_O we set the bits of the key masks that correspond to possible keys. Each pair has five such key masks, one for every S box. A clique is defined as a set of pairs for which for each of the five key masks there is a common bit set in all the pairs in the set (i.e., the binary "and" operation is non zero for all the five key masks). Finding the largest clique can be done in the following way: first compare the key masks of every pair with all the following pairs in the pairs list. At each comparison there is usually at least one key mask without any common bit set. For the remaining possibilities we try to "and" the result with third pairs, fourth pairs and so on until no more pairs can be added to the clique. Given the largest clique we can easily compute the corresponding key bits by looking at each key mask for the key value it represents.

Using the clique algorithm with 240 ciphertexts it takes about 0.3 seconds on a COMPAQ personal computer to find the key in 95% of the tests conducted on DES reduced to six rounds. When 320 ciphertexts are used the program succeeds in almost all the cases. The program uses about 100K bytes of memory, most of which is devoted to various preprocessed tables used to speed up the algorithm.

5 DES reduced to eight rounds

DES reduced to eight rounds can be broken using about 25000 ciphertext pairs for which the plaintext XOR is $P' = 40\ 5C\ 00\ 00\ 04\ 00\ 00\ 00_x$. The method finds 30 bits of K8. 18 additional key bits can be found using similar manipulations on the pairs. The remaining eight key bits can be found using exhaustive search.

The following characteristic is used in this analysis:



This characteristic has probability $\frac{1}{10485.76}$. The input XOR in the sixth round of a right pair is

$$f' = d' \oplus E' = b' \oplus A' = L' = 40\ 5C\ 00\ 00_x.$$

Consequently, for five S boxes $S'_{Ef} = S'_{If} = 0$ and $S'_{Of} = 0$.

Note There is an additional five-round characteristic with probability about

$\frac{1}{33000}$. Its plaintext XOR is

$$\Omega_P = 04\ 04\ 07\ 80\ \ 00\ 20\ 20\ 00_x.$$

In this characteristic only four S boxes in the sixth round satisfy $S'_{Ef} = 0$. There are other characteristics for which either the probability or the number of unchanged S boxes in the sixth round are smaller, and thus their use is less advantageous.

In right pairs the five S boxes S2, S5, ..., S8 satisfy $S'_{Ef} = S'_{If} = 0$ and $S'_{Of} = 0$. By $H' = l' \oplus g' = l' \oplus e' \oplus F'$ we can find the output XORs of the corresponding S boxes in the eighth round. The input data of the eighth round is known from the ciphertexts. Therefore, we can use the counting method to find the 30 subkey bits entering the five S boxes at the eighth round. The signal to noise ratio of this counting scheme is $S/N = \frac{2^{30}}{4^5 \cdot 10485.76} = 100$.

Counting on 30 subkey bits demands a huge memory of 2^{30} counters. In this case the clique method is not recommended since its computation time grows very fast (more than quadratically) with the number of pairs, while the computation time of the counting method is linear in the number of pairs. Nevertheless, we can reduce the amount of memory by counting on fewer subkey bits entering fewer S boxes. The remaining S boxes can be used for identification of some of the wrong pairs (in which $S'_{Eh} \not\to S'_{Oh}$). About 20% of the entries in the pairs XOR distribution tables are impossible and thus each remaining S box discards 20% of the wrong pairs. Counting on 24 key bits thus has $S/N = \frac{2^{24}}{4^4 \cdot 0.8 \cdot 10485.76} \approx 7.8$ and counting on 18 key bits has $S/N = \frac{2^{18}}{4^3 \cdot 0.8^2 \cdot 10485.76} \approx 0.6$.

In counting schemes that count on a reduced number of bits we can choose the reduced set of countable S boxes arbitrarily. In this particular case we can choose the reduced set with the advantage of increasing the characteristic's probability and the signal to noise ratio by using a slightly modified characteristic which ignores output bits that are not counted anyway. The slightly modified characteristic is similar to the original one except that in the fifth round only one bit of $S2'_{Oe}$ is fixed and all the combinations of the other three are allowed:

$$e' = 04\ 00\ 00\ 00_x \to E' = P(0W\ 00\ 00\ 00_x) = X0\ 0Y\ Z0\ 00_x,$$

where $W \in \{0,1,2,3,8,9,A,B\}$, $X \in \{0,4\}$, $Y \in \{0,8\}$ and $Z \in \{0,4\}$. Therefore at the sixth round

$$f' = X0\ 5V\ Z0\ 00_x$$

where $V = Y \oplus 4$. The only possible combination in which $Z = 0$ is $04\ 00\ 00\ 00_x \rightarrow 40\ 08\ 00\ 00_x$ which has probability $\frac{16}{64}$. All the other combinations (in which $Z = 4$) have an overall probability $\frac{20}{64}$. We cannot count on the subkey bits $S5_{Kh}$ but it is still advisable to check the possibility of $S5'_{Eh} \rightarrow S5'_{Oh}$ which is satisfied by 80% of the pairs. Therefore, the probability of $e' \rightarrow E'$ is $\frac{16}{64} + 0.8\frac{20}{64} = \frac{32}{64} = \frac{1}{2}$. The probability of the five-round modified characteristic is $\frac{16 \cdot 10 \cdot 16}{64^3} \cdot \frac{16 \cdot 10 \cdot 32}{64^3} \approx \frac{1}{5243}$. The signal to noise ratio of a counting scheme which count on the 24 subkey bits entering S2, S6, S7 and S8 is $S/N = \frac{2^{24}}{4^4 \cdot 0.8 \cdot 5243} \approx 15.6$. This signal to noise ratio allows to use only about five right pairs. Therefore, it uses a total amount of about 25000 pairs. The signal to noise ratio of a counting scheme which counts on 18 subkey bits entering three S boxes out of S2, S6, S7 and S8 is $S/N = \frac{2^{18}}{4^3 \cdot 0.8^2 \cdot 5243} \approx 1.2$. This counting scheme which counts on 18 bits needs 150000 pairs and has an average of about 24 counts for any wrong key value and about 53 counts for the right key value ($53 = 24 + \frac{150000}{5243} = 24 + 29$).

A summary of this cryptanalytic method using 2^{18} memory cells is as follows:

1. Set up an array of 2^{18} counters which is initialized by zeroes. The array corresponds to the 2^{18} values of the 18 key bits of K8 entering S6, S7 and S8.

2. Preprocess the possible values of S_I that satisfy each $S'_I \rightarrow S'_O$ for the eight S boxes into a table. This table is used to speed up the program.

3. For each ciphertext pair do:

   (a) Assume $h' = r'$, $H' = l'$ and $h = r$. Calculate $S'_{Eh} = S'_{Ih}$ and $S_{Oh'}$ for S2, S5, ..., S8 by h' and H'. Calculate S_{Eh} for S6, S7 and S8 by h.

   (b) For each one of the S boxes S2, S5, S6, S7 and S8 check if $S'_{Ih} \not\rightarrow S'_{Oh}$. If $S'_{Ih} \not\rightarrow S'_{Oh}$ for one of the S boxes then discard the pair as a wrong pair.

(c) For each one of the S boxes S6, S7 and S8: fetch from the preprocessed table all the values of S_{Ih} which are possible for $S'_{Ih} \rightarrow S'_{Oh}$. For each possible value calculate $S_{Kh} = S_{Ih} \oplus S_{Eh}$. Increment by one all the counters corresponding to combinations of the possible values of $S6_{Kh}$, $S7_{Kh}$ and $S8_{Kh}$.

4. Find the entry in the array that contains the maximal count. The entry index is most likely to be the real value of $S6_{Kh}$, $S7_{Kh}$ and $S8_{Kh}$ which is the value of the 18 bits $31, \ldots, 48$ of K8.

To find the other bits, we filter all the pairs and leave just the pairs with the expected S'_O value using the known values of h and the known bits of K8 entering S6, S7 and S8. The expected number of the remaining pairs is 53.

The next bits we are looking for are the twelve bits of K8 that correspond to S2 and S5. We use a similar counting method (exploiting the enhanced S/N created by the higher concentration of right pairs) and then filter more pairs. A wrong pair is not discarded by either this filter or its predecessor with probability 2^{-20} and thus almost all the remaining pairs are right pairs.

Using the known subkey bits of K8 we can calculate the values of 20 bits of each of H and H^* for each pair and thus 20 bits of each of g and g^* (by $g = l' \oplus H'$). Table 9 shows the dependence of the g bits and the subkey bits of K7 at the seventh round on the known and unknown subkey bits of K8 at the eighth round. The digits `1`, `3` and `4` mean that they depend on the value of the unknown key bits entering the corresponding S box in the eighth round. '`+`' means that it depend only on the known bits of K8. Eight key bits are not used at all in K8 and are marked by '`.`'.

The expected value of G' is known by the formula $G' = f' \oplus h'$. We can now look for the 18 missing bits of K8 by exhaustive search of 2^{18} possibilities for every pair. Thus we know H, H^* and g, g^* and 40 bits of K7. For each pair we check that the expected value of G' holds. For the right value of those 18 key bits the expected G' holds for almost all the filtered pairs. All the other possible values satisfy the expected G' value only for a few pairs (usually 2–3 pairs while the right value holds for 15 pairs). To save computer time we search primarily for the 12 key kits entering S1 and S4 in the eighth round. They suffice to compute $S3'_{Og}$ as seen in table 9. By similar methods

Into S box #	g bits S_{Eg}	Key bits S_{Kg}
S1	`+4++++`	`3+..4+`
S2	`++3++1`	`134333`
S3	`+14+++`	`+1+41+`
S4	`++++31`	`11..1+`
S5	`31++4+`	`+++.++`
S6	`4++13+`	`+.+.++`
S7	`3+4+++`	`+++.++`
S8	`++31+4`	`++++++`

Table 9. Known bits at the seventh round.

we find these 12 bits and then find the other eight bits. This completes the calculation of the 48 bits of K8. Only eight key bits are still missing and they can be found by exhaustive search of 256 cases, using one pair of ciphertexts, and verifying that the plaintext XOR is as expected.

To save disk space we can filter the pairs as soon as they are created and discard all the identifiable wrong pairs (leaving $0.8^5 \approx \frac{1}{3}$ of all the pairs). Therefore, in the case of counting on 24 bits, the 25000 pairs are reduced to about 7500 pairs. For the case of counting on 18 bits we devised another criterion which discards most of the wrong pairs while leaving almost all the right pairs. This criterion is based on a carefully chosen weighting function and discards any pair whose weight is lower than a particular threshold. This criterion is the extension of the filtering of the identifiable wrong pairs (where the threshold is actually zero) and is based on the idea that a right pair typically suggests more possible key values than a wrong pair. The weighting function is the product of the number of possible keys of each of the five countable S boxes (i.e., the number in the corresponding entry in the pairs XOR distribution tables). The threshold is chosen to maximize the amount of discarded pairs, while leaving as many right pairs as possible. The best threshold value was experimentally found to be 8192 which discards about 97% of the wrong pairs and leaves almost all the right pairs. This reduces the number of pairs we actually analyze from 150000 to about 7500, with a corresponding reduction in the running time of the attack.

$S2_I$	$S2^*_I$	$S2_O$	$S2^*_O$
123456	123456	1234	1234
000010	001010	0001	1011
000110	001110	1110	0100
010001	011001	1100	0110
010101	011101	0001	1011
100000	101000	0000	1010
100010	101010	1110	0100
100100	101100	0111	1101
100110	101110	1011	0001

Table 10. The possible instances of $08_x \to A_x$ by S2 (in binary).

The attacking program finds the key in less than two minutes on a COMPAQ personal computer with 95% success rate (using 150000 pairs). Using 250000 pairs the success rate is increased to almost 100%. The program uses 460K bytes of memory, most of it for the counting array (one byte suffices for each counter since the maximum count is about 53, and thus the total array size is 2^{18} bytes), and the preprocessed speed up tables. The program which counts using 2^{24} memory cells finds the key using only 25000 pairs.

5.1 Enhanced characteristic's probability

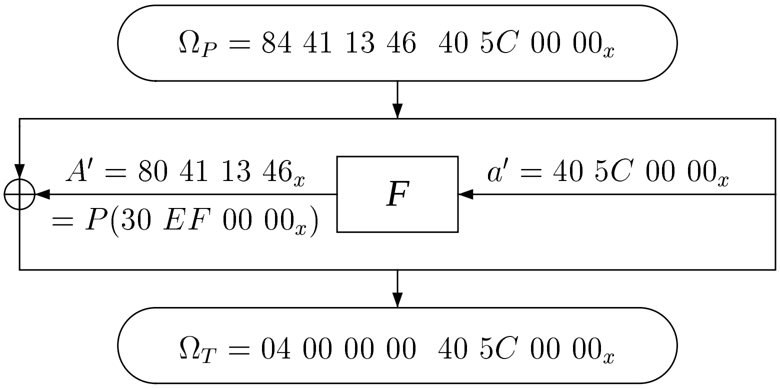
In addition to the statistical behavior of the characteristic we can use the possible values of individual input and output bits of the S boxes. Lets look at the first round of the characteristic. We have $08_x \to A_x$ by S2 with probability $\frac{16}{64}$. Table 10 describes the possible input and output values.

We can see that the input bits number 2 and 6 are always equal. In addition for $\frac{12}{16}$ of the input values they are both 0 and for $\frac{4}{16}$ of them they are both 1. If we know the XOR of the key bits entering these two bits of S2 in the first round (i.e., bits 57 and 42 of the key) we can use only plaintexts whose corresponding bits (i.e., bits 5 and 9) have the same XOR value (causing bits number 2 and 6 to be equal). Other pairs of plaintexts cannot satisfy the characteristic. The statistics and the S/N ratio are then twice as good, and let us use less than half the number of pairs.

If we know the values of both bits in a key we can choose the two bits in the plaintexts s.t. the bit values entering S2 are both zero. In this case the statistics for S2 becomes $\frac{12}{16}$ instead of $\frac{16}{64}$. Thus we get a factor of three in the statistics and the S/N. The higher S/N lets us use less than $\frac{1}{3}$ of the pairs needed originally. A factor of four can be easily obtained by a characteristic that holds for all the inputs in which bit number 1 has value 1 and both bits number 2 and 6 have value 0.

5.2 Extension to nine rounds

The five-round characteristic can be extended to six rounds by concatenating it to the following characteristic:



with probability about $\frac{1}{100}$

This characteristic has probability $\frac{12\cdot 14\cdot 16}{64^3} \approx \frac{1}{100}$ and thus the probability of the concatenated six-round characteristic is about $\frac{1}{1000000}$.

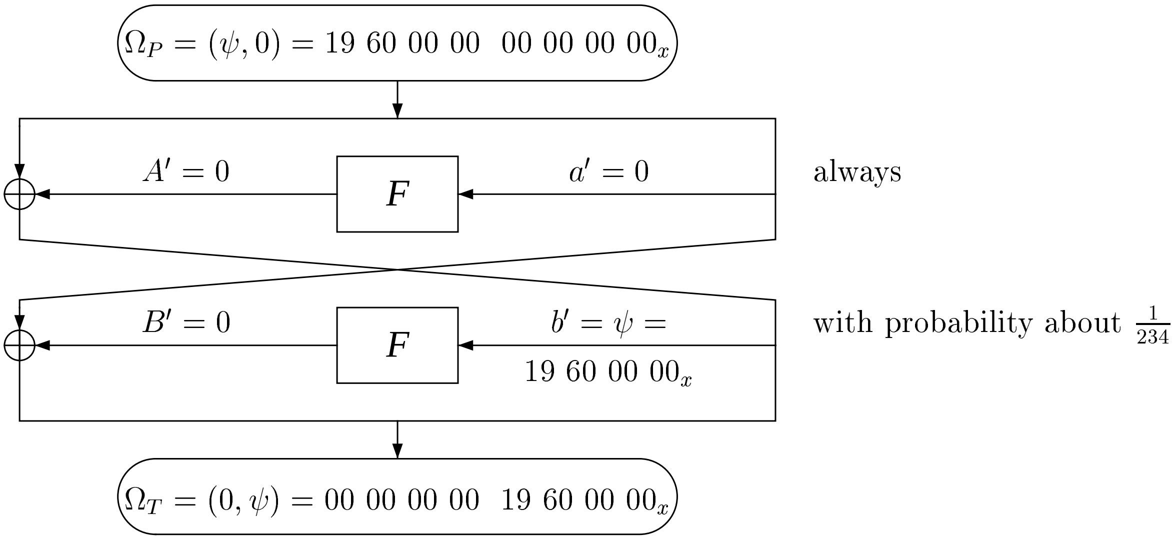
DES reduced to nine rounds can be broken using 30-million pairs by a method based on this six-round characteristic and using an array of size 2^{30} with $S/N = \frac{2^{30}}{4^5\cdot 1000000} \approx 1$. The first part of the algorithm that finds the first 30 key bits is almost the same as in the eight rounds algorithm except that it counts on all the 30 bits at once. The second part of the algorithm that uses table 9 is slightly different since the key scheduling at the ninth round is based on a shift of one bit instead of two bits. The input part stays the same.

6 DES with an arbitrary number of rounds

The following iterative characteristic can be used to cryptanalyze (at least in principle) variants of DES with an arbitrary number of rounds.

Notation The value 19 60 00 00_x is denoted by ψ.

The iterative characteristic is:



Due to the importance of this iterative characteristic, throughout this paper we refer it as *the iterative characteristic.*

Lemma 2 The iterative characteristic has probability $\frac{14 \cdot 8 \cdot 10}{64^3} \approx \frac{1}{234}$.

Proof $S'_{Eb} \neq 0$ only at three S boxes: S1, S2 and S3, for which:

$$\begin{array}{llll} S1'_{Eb} = S1'_{Ib} = 03_x & \rightarrow & S1'_{Ob} = 0 & \text{with probability } \frac{14}{64} \\ S2'_{Eb} = S2'_{Ib} = 32_x & \rightarrow & S2'_{Ob} = 0 & \text{with probability } \frac{8}{64} \\ S3'_{Eb} = S3'_{Ib} = 2C_x & \rightarrow & S3'_{Ob} = 0 & \text{with probability } \frac{10}{64} \end{array}$$

and for the other S boxes (S4, ..., S8):

Number of rounds	Probability
3	$\frac{1}{234}$
5	$\frac{1}{55000}$
7	$\approx 2^{-24}$
9	$\approx 2^{-32}$
11	$\approx 2^{-40}$
13	$\approx 2^{-48}$
15	$\approx 2^{-56}$

Table 11. The probability of the iterative characteristic versus number of rounds.

$$S'_{Eb} = S'_{Ib} = 0 \quad \rightarrow \quad S'_{Ob} = 0 \qquad\qquad \text{always.}$$

Thus $B' = 0$ with probability $\frac{14\cdot 8\cdot 10}{64^3} \approx \frac{1}{234}$. ■

Theorem 2 By an iterative concatenation of the iterative characteristic with itself and with the one-round characteristic with probability 1 (described in example 9) we get characteristics with probabilities as summarized in table 11. In addition the plaintext XORs and the ciphertext XORs of these characteristics are equal:

$$\Omega_P = \Omega_T = 19\ 60\ 00\ 00\ \ 00\ 00\ 00\ 00_x = (\psi, 0)$$

and for the next round (w.l.g. a five-round characteristic)

$$f' = \psi$$

and five of its S boxes satisfy $S'_{Ef} = 0$.

Proof The results of this theorem are derived from definition 11 and lemma 2. The XOR data during the intermediate rounds looks like:

$$\begin{array}{rcll} \Omega_P & = & (\psi, 0) & \\ a' = 0 & \rightarrow & A' = 0 & \qquad \text{always} \end{array}$$

$$
\begin{array}{rcll}
b' = \psi & \rightarrow & B' = 0 & \text{with probability about } \frac{1}{234} \\
c' = a' \oplus B' = 0 & \rightarrow & C' = 0 & \text{always} \\
d' = \psi & \rightarrow & D' = 0 & \text{with probability about } \frac{1}{234} \\
e' = c' \oplus D' = 0 & \rightarrow & E' = 0 & \text{always} \\
 & \vdots & &
\end{array}
$$

and so forth for any number of rounds. ■

Note There is another value for which lemma 2 and theorem 2 hold with the same probabilities. This value is $\psi^{\dagger} = 1B\ 60\ 00\ 00_x$. There are several additional values for which the probabilities are smaller. The best of them is $\psi^{\ddagger} = 00\ 19\ 60\ 00_x$ for which the probability is exactly $\frac{1}{256}$. The extension of this iterative characteristic to 15 rounds has probability 2^{-56}.

There are several possible types of attacks, depending on the number of additional rounds in the cryptosystem that are not covered by the characteristic itself. The attack on DES reduced to eight rounds in section 5 uses a five-round characteristic and there were three additional rounds. This kind of attack is called a 3R-attack. The other kinds of attacks are a 2R-attack with two additional rounds and a 1R-attack with one additional round (where the characteristic causes r' to be fixed). A 0R-attack is also possible but it can be reduced to a 1R-attack with a better statistics and the same S/N. A 0R-attack has the advantage that the right pairs can be recognized almost without mistakes (the probability of a wrong pair to survive is 2^{-64}) and thus the memory requirements can become negligible using the clique method. For a fixed cryptosystem it is advisable to use the shortest possible characteristic due to its better statistics. Thus, a 3R-attack is advisable over a 2R-attack and both are advisable over a 1R-attack.

In the following sections the actual attacks on DES reduced to 8–16 rounds are described. All these attacks find some bits of the subkey of the last round. The other bits of the subkey of the last round can be calculated using these known bits and a reduction of the cryptosystem to a smaller number of rounds can be done. Only eight bits do not appear in the subkey of the last round and they can be found by trying all the 256 possible keys.

6.1 3R-attacks

In 3R-attacks counting can be done on all the bits of the subkey of the last round entering the S boxes that have zero input XORs at the round that follows the last round of the characteristic. The four, six, eight and nine-round attacks described in the previous sections are of this type.

In DES reduced to eight rounds the first 30 subkey bits can be found using the iterative characteristic with five rounds (whose probability is about $\frac{1}{55000}$) by an attack which is similar to the one described in section 5. Using an array of size 2^{24} we have $S/N = \frac{2^{24}}{4^4 \cdot 0.8 \cdot 55000} = 1.5$. We need about 2^{20} pairs. Using an array of size 2^{30} we have $S/N = \frac{2^{30}}{4^5 \cdot 55000} \approx 19$. About 67% $(1 - 0.8^5)$ of the pairs can be identified in advance as wrong pairs.

6.2 2R-attacks

In 2R-attacks, counting can be done on all the bits of the subkey of the last round. Possibility checks can be done for all the previous round S boxes. An S box whose input XOR is zero should also have an output XOR of zero, i.e., the success rate of this check is $\frac{1}{16}$. For the other S boxes the success rate is about 0.8.

In DES reduced to nine rounds the 48 bits of K9 can be found using 2^{26} pairs using the seven-round characteristic. We know that:

$$
\begin{array}{llll}
\Omega_P = (\psi, 0) & & & \\
a' = 0 & \rightarrow & A' = 0 & \text{always} \\
b' = \psi & \rightarrow & B' = 0 & \text{with probability about } \frac{1}{234} \\
c' = 0 & \rightarrow & C' = 0 & \text{always} \\
 & \vdots & & \\
g' = 0 & \rightarrow & G' = 0 & \text{always} \\
h' = \psi & \rightarrow & H' = i' \oplus g' = r' & \\
i' = r' & \rightarrow & I' = h' \oplus l' = l' \oplus \psi & \\
\Omega_T = (l', r') & & &
\end{array}
$$

We can check that $h' \rightarrow H'$ and $i' \rightarrow I'$ and count the possible occurrences of the key bits. At $h' \rightarrow H'$ five S boxes satisfy $S'_{Eh} = S'_{Ih} = 0$ and thus S'_{Oh} must be zero (which happens for wrong pairs with probability $\frac{1}{16}$), while the other three S boxes satisfy $S'_{Ih} \rightarrow S'_{Oh}$ (which happens for wrong pairs with probability 0.8). Therefore the counting on all the 48 bits of K9 has $S/N = \frac{2^{48} \cdot 2^{-24}}{4^8 \cdot 0.8^3 \cdot (\frac{1}{16})^5} \approx 2^{29}$ and counting on 18 bits has $S/N = \frac{2^{18} \cdot 2^{-24}}{4^3 \cdot 0.8^5 \cdot 0.8^3 \cdot (\frac{1}{16})^5} \approx 2^{11}$. Even a separate counting on the six key bits entering each S box is possible with $S/N = \frac{2^6 \cdot 2^{-24}}{4 \cdot 0.8^7 \cdot 0.8^3 \cdot (\frac{1}{16})^5} \approx 10$. The identification of the wrong pairs leaves only $0.8^3 \cdot \left(\frac{1}{16}\right)^5 \cdot 0.8^8 \approx 2^{-24}$ of the wrong pairs and thus only about one wrong pair is left per each right pair. The characteristic's probability is 2^{-24} and thus we need about 2^{26} pairs for the cryptanalysis. This attack needs more data than the previous 3R-attack on DES reduced to nine rounds but needs much less memory. Due to the very good identification of wrong pairs (only about eight pairs are not discarded, four right pairs and four wrong pairs) it is possible to use the clique method on all the 48 bits.

Eleven rounds can be broken by using the nine-round characteristic with an array of size 2^{18} and $S/N = \frac{2^{18} \cdot 2^{-32}}{4^3 \cdot 0.8^5 \cdot 0.8^3 \cdot (\frac{1}{16})^5} \approx 6$ using 2^{35} pairs. The clique method can still be used on 48 subkey bits with $S/N = \frac{2^{48} \cdot 2^{-32}}{4^8 \cdot 0.8^3 \cdot (\frac{1}{16})^5} \approx 2^{21}$ with an identification that leaves $2^{32} \cdot 2^{-24} = 2^8$ wrong pairs per each right pair.

13 rounds can be broken using the eleven-round characteristic with an array of size 2^{30} and $S/N = \frac{2^{30} \cdot 2^{-40}}{4^5 \cdot 0.8^3 \cdot 0.8^3 \cdot (\frac{1}{16})^5} \approx 4$ using 2^{43} pairs. The clique method is not possible since $2^{43} \cdot 2^{-24} = 2^{19}$ pairs are not discarded. Counting schemes on 18 and 24 bits are not advisable due to the low S/N.

15 rounds can be broken using the 13-round characteristic with an array of size 2^{42} and $S/N = \frac{2^{42} \cdot 2^{-48}}{4^7 \cdot 0.8 \cdot 0.8^3 \cdot (\frac{1}{16})^5} \approx 2.5$ using 2^{51} pairs. This is still faster than exhaustive search, but requires unrealistic amounts of space and ciphertexts.

6.3 1R-attacks

In 1R-attacks counting can be done on all the bits of the subkey of the last round entering the S boxes with non zero input XORs. Verification of the values of r' itself and possibility checks on all the other S boxes in the last round can be done. For those S boxes with a zero input XOR the output XOR should be zero too, i.e., the check success rate is $\frac{1}{16}$. Since the input XOR is constant we cannot distinguish between several subkey values. However, the number of such values is small (eight in all the 1R-attacks described here) and each can be checked later in parallel by the next part of the algorithm (either via exhaustive search or by a differential cryptanalysis attack).

Ten rounds can be broken using the nine-round characteristic where

$$\begin{array}{rcll} h' = \psi & \rightarrow & H' = 0 & \text{with probability } \frac{1}{234} \\ i' = 0 & \rightarrow & I' = 0 & \text{always} \\ j' = \psi = r' & \rightarrow & J' = l' \oplus i' = l'. & \end{array}$$

We can identify the right pairs easily. Those pairs satisfy $r' = \psi$ and the 20 bits in l' going out of S4, ..., S8 are zero. This also holds for 2^{-52} of the wrong pairs. For the other three S boxes we count the possible values of their 18 key bits with $S/N = \frac{2^{18} \cdot 2^{-32}}{4^3 \cdot 2^{-52}} = 2^{32}$. Thus we need 2^{34} pairs.

Twelve rounds can be broken using the eleven-round characteristic with $S/N = \frac{2^{18} \cdot 2^{-40}}{4^3 \cdot 2^{-52}} = 2^{24}$ and with 2^{42} pairs.

Fourteen rounds can be broken using the 13-round characteristic with $S/N = \frac{2^{18} \cdot 2^{-48}}{4^3 \cdot 2^{-52}} = 2^{16}$ and with 2^{50} pairs.

For sixteen rounds we get $S/N = \frac{2^{18} \cdot 2^{-56}}{4^3 \cdot 2^{-52}} = 2^8$ using the 15-round characteristic. This can be broken using 2^{57} pairs. Note that the creation of 2^{57} pairs is more time consuming than exhaustive search for the 2^{56} possible keys.

6.4 Summary of the cryptanalysis

A summary of the cryptanalytic results appears in table 12. The description

# of rnds	#pairs needed	#pairs used	#bits found	Characteristics		S/N		Comments
4	2^3	2^3	42	1	1	16	[6]	
6	2^7	2^7	30	3	$\frac{1}{16}$	2^{16}	$*$	
8	2^{15}	2^{13}	30	5	$\frac{1}{10486}$	15.6	[24]	
8	2^{17}	2^{13}	30	5	$\frac{1}{10486}$	1.2	[18]	
8	2^{20}	2^{19}	30	5	$\frac{1}{55000}$	1.5	[24]	The iterative characteristic.
9	2^{25}	2^{24}	30	6	$\frac{1}{1000000}$	1.0	[30]	Extension to six rounds.
9	2^{26}	8	48	7	2^{-24}	2^{29}	$*$	
10	2^{34}	4	18	9	2^{-32}	2^{32}	$*$	
11	2^{35}	2^{11}	48	9	2^{-32}	2^{21}	$*$	
12	2^{42}	4	18	11	2^{-40}	2^{24}	$*$	
13	2^{43}	2^{19}	48	11	2^{-40}	4	[30]	
14	2^{50}	4	18	13	2^{-48}	2^{16}	$*$	
15	2^{51}	2^{27}	48	13	2^{-48}	2.5	[42]	Needs a huge memory. With less memory needs 2^{57} pairs.
16	2^{57}	2^5	18	15	2^{-56}	2^8	$*$	Slower than exhaustive search.

Table 12. Summary of the cryptanalysis of DES.

of each field is as follows:

# of rnds: The number of rounds in the cryptosystem.

#pairs needed: The number of pairs needed to cryptanalyze the cryptosystem. *The number of ciphertexts needed is twice the number of pairs.*

#pairs used: The number of pairs that are actually used in the attack, excluding the identifiable wrong pairs that can be easily discarded during the collection phase.

$S2_I$ 123456	$S2_I^*$ 123456	$S2_O = S2_O^*$ 1234
000111	110101	0111
001111	111101	1110
010101	100111	0001
010111	100101	1010

Table 13. Possible inputs and outputs for $32_x \to 0$ by S2 (in binary).

#bits found: The number of key bits found in the initial attack (using a single characteristic). The other key bits can be found by auxiliary techniques.

Characteristic: The number of rounds and the probability of the characteristic used in the attack.

S/N**:** The signal to noise ratio of the attack. The number in brackets (if any) denotes the number of initial bits found with that S/N. An asterisk denotes that the clique method is preferable over the counting method and then the S/N is on the number of bits found. The other key bits are found either in parallel or at a second pass.

Comments: Real comments.

6.5 Enhanced characteristic's probability

In addition to the statistical behavior of the iterative characteristic we can use the individual values of the input and output bits of the S boxes.

In the iterative characteristic we have the following behavior. When $32_x \to 0$ by S2 the values of the input bits number 4 and 6 are always both 1 (see table 13). It does not happen in the first round and thus it cannot be used as in section 5.1. Also we have $2C_x \to 0$ by S3 where in $\frac{8}{10}$ of the cases bit number 2 equals 0 and in $\frac{2}{10}$ of the cases bit number 2 equals 1 (see table 14).

$S3_I$ 123456	$S3^*_I$ 123456	$S3_O = S3^*_O$ 1234
000010	101110	0000
000011	101111	0111
000111	101011	1001
001111	100011	1010
010001	111101	0010

Table 14. Possible inputs and outputs for $2C_x \rightarrow 0$ by S3 (in binary).

The XOR value bit 6 of $S2_I$ and bit 2 of $S3_I$ equals the XOR value of the corresponding key bits in $S2_K$ and $S3_K$ since the corresponding bits in $S2_E$ and $S3_E$ are the same bit due to the bit expansion. If their XOR value is known to be 1 then the probability of the iterative characteristic becomes $\frac{14 \cdot 8 \cdot 8}{64^2 \cdot 32} = \frac{7}{2^{10}} \approx \frac{1}{146}$. If their XOR value is known to be 0 then the probability becomes $\frac{14 \cdot 8 \cdot 2}{64^2 \cdot 32} = \frac{7}{2^{12}} \approx \frac{1}{595}$.

The other characteristic described with the same probability has the opposite direction. When $36_x \rightarrow 0$ by S2 the value of bit number 6 is always 0 and thus the probabilities are exchanged. If the XOR of the key bits is 0 then the probability is $\frac{1}{146}$ and if 1 it is $\frac{1}{595}$.

The attack on DES with 16 rounds is now as follows. There are seven rounds in which the input XOR is assumed to be ψ. Suppose that, out of these seven rounds, we have n rounds $(0 \leq n \leq 7)$ whose key bit number 6 of $S2_K$ equals key bit number 2 of $S3_K$. In this case, the probability of the 15-round characteristic is

$$\left(\frac{7}{2^{12}}\right)^n \left(\frac{7}{2^{10}}\right)^{7-n} = 4^n \left(\frac{7}{2^{12}}\right)^7 \approx 1.6\frac{4^n}{2^{65}}.$$

For the other characteristic it is $1.6\frac{4^{7-n}}{2^{65}}$. Table 15 describes the probabilities for each number of equalities among the key bits and the relative frequency of such keys.

To increase the probability (especially in the worse cases) we use quartets based on both characteristics. Since both characteristics allow counting on

# of equals	keys ratio	probability first char	probability other char	sum of probabilities	# needed ciphertexts
0	$\frac{1}{128}$	$1.6 \cdot 2^{-65}$	$1.6 \cdot 2^{-51}$	$1.6 \cdot 2^{-51}$	$1.25 \cdot 2^{52}$
1	$\frac{7}{128}$	$1.6 \cdot 2^{-63}$	$1.6 \cdot 2^{-53}$	$1.6 \cdot 2^{-53}$	$1.25 \cdot 2^{54}$
2	$\frac{21}{128}$	$1.6 \cdot 2^{-61}$	$1.6 \cdot 2^{-55}$	$1.625 \cdot 2^{-55}$	$1.23 \cdot 2^{56}$
3	$\frac{35}{128}$	$1.6 \cdot 2^{-59}$	$1.6 \cdot 2^{-57}$	2^{-56}	2^{58}
4	$\frac{35}{128}$	$1.6 \cdot 2^{-57}$	$1.6 \cdot 2^{-59}$	2^{-56}	2^{58}
5	$\frac{21}{128}$	$1.6 \cdot 2^{-55}$	$1.6 \cdot 2^{-61}$	$1.625 \cdot 2^{-55}$	$1.23 \cdot 2^{56}$
6	$\frac{7}{128}$	$1.6 \cdot 2^{-53}$	$1.6 \cdot 2^{-63}$	$1.6 \cdot 2^{-53}$	$1.25 \cdot 2^{54}$
7	$\frac{1}{128}$	$1.6 \cdot 2^{-51}$	$1.6 \cdot 2^{-65}$	$1.6 \cdot 2^{-51}$	$1.25 \cdot 2^{52}$

Table 15. Probabilities by number of key bits equalities.

the same S boxes we can use them simultaneously. We can see from the table that even though we can now break 16 rounds with less than 2^{56} encryptions, it does not work for all the keys but only for a small fraction of them. For this fraction exhaustive search is still faster. Table 15 shows that although the knowledge of the specific bit values during the rounds of the characteristics enhances the attack and decreases the number of pairs needed, the improvement is relatively small and does not affect the overall complexity.

7 Variants of DES

This section describes several variants of DES and how the attack works on them.

7.1 Modifying the P permutation

All the attacks based on the iterative characteristic are independent of the choice of the P permutation. Thus any modification of the P permutation by any other permutation cannot make the attack less successful.

7.2 Modifying the order of the S boxes

The DES cryptosystem specifies a certain order of the eight S boxes. A modification of the order of the S boxes can make the cryptosystem much weaker. Consider for example the case in which $S1$, $S7$ and $S4$ are brought together in this order (w.l.g. in the first three places) and the other S boxes are set in any order. Then there is a similar iterative characteristic. This characteristic is denoted by $\psi^{\bullet} = 1D\ 40\ 00\ 00_x$ where

$$\begin{array}{ll} \text{S1: } 03_x \rightarrow 0 & \text{with probability } \frac{14}{64} \\ \text{S7: } 3A_x \rightarrow 0 & \text{with probability } \frac{16}{64} \\ \text{S4: } 28_x \rightarrow 0 & \text{with probability } \frac{16}{64} \end{array}$$

and $\psi^{\bullet} \rightarrow 0$ with probability $\frac{14 \cdot 16 \cdot 16}{64^3} \approx \frac{1}{73}$.

The 15-round characteristic has probability $\frac{1}{73^7} \approx 2^{-43}$ and thus the 16-round cryptosystem can be attacked using 2^{45} pairs with $S/N = \frac{2^{18} \cdot 2^{-43}}{4^3 \cdot 2^{-52}} = 2^{21}$.

The 17-round characteristic has probability $\frac{1}{73^8} \approx 2^{-50}$ and thus the 18-round cryptosystem can be attacked using 2^{52} pairs with $S/N = \frac{2^{18} \cdot 2^{-50}}{4^3 \cdot 2^{-52}} = 2^{14}$.

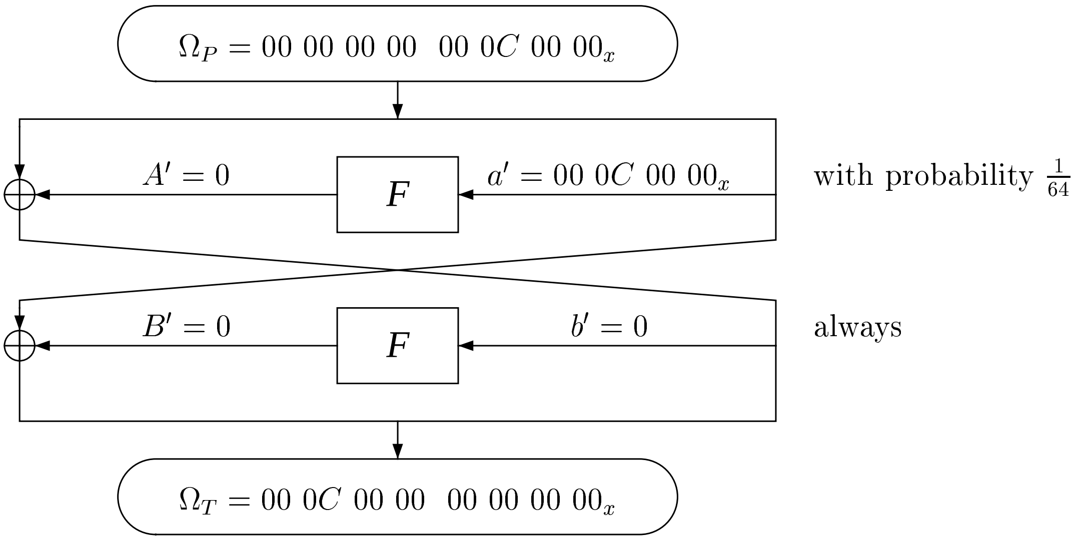
In these attacks the clique method can be used due to the excellent identification of wrong pairs (only 2^{-53} of them remain). As in the attack based on the iterative characteristic this attack is independent of the choice of the P permutation.

7.3 Modifying XORs by additions

In DES there are two XOR operations in each round. The first XORs the expanded input with the subkey within the F function while the other XORs the output of the F function with the other half of the input data. The following subsections describe three possible modifications which replace some of the XOR operations by addition operations. The same analysis holds for modification by subtraction operations.

7.3.1 Modifying the XORs within the F function

If we replace the occurrences of the XORs within the F function by addition operations we get a much weaker cryptosystem. The attack uses the following iterative characteristic:



The $00\ 0C\ 00\ 00_x \rightarrow 0$ should be explained: $00\ 0C\ 00\ 00_x$ is the input XOR of the F function. The expansion to 48 bits is 000058000000_x. The addition of the key causes the input XOR to become 000028000000_x with probability $\frac{1}{16}$. Thus the input XORs of all the S boxes except S4 is zero, while $S4'_I = 28_x$. However, $28_x \rightarrow 0$ by S4 with probability $\frac{1}{4}$.

The 15-round characteristic has probability $(\frac{1}{64})^7 = 2^{-42}$. The 1R-attack counting scheme which finds the six subkey bits entering S4 in the sixteenth round has $S/N = \frac{2^6}{2^{42} \cdot 2^{-32} \cdot 2^{-24} \cdot 4} = 2^{18}$. Thus the attack needs about 2^{44} pairs of encryptions. The six key bits entering S3 can then be found using the same encryptions with even higher signal to noise ratio. Exhaustive search of the 2^{44} possible keys (with 12 fixed bits) recovers the right key. The total complexity of this attack is thus 2^{45}.

7.3.2 Modifying all the XORs

Modifying all the XORs by additions change the probability of this characteristic from 2^{-6} to 2^{-8}. This happens because the additional addition operation (for example $c = a + B$) does not change the input XOR ($c' = a'$ for $B' = 0$) with probability $\frac{1}{4}$. Thus the 16-round characteristic has probability 2^{-64}, the 15-round characteristic has probability 2^{-58}, the 14-round characteristic has probability 2^{-56} and the 13-round characteristic has probability 2^{-50}.

The analysis of this attack shows that 2^{52} pairs are needed to cryptanalize the 14-round cryptosystem. The attacks on the 15-round and 16-round cryptosystems are slower than exhaustive search.

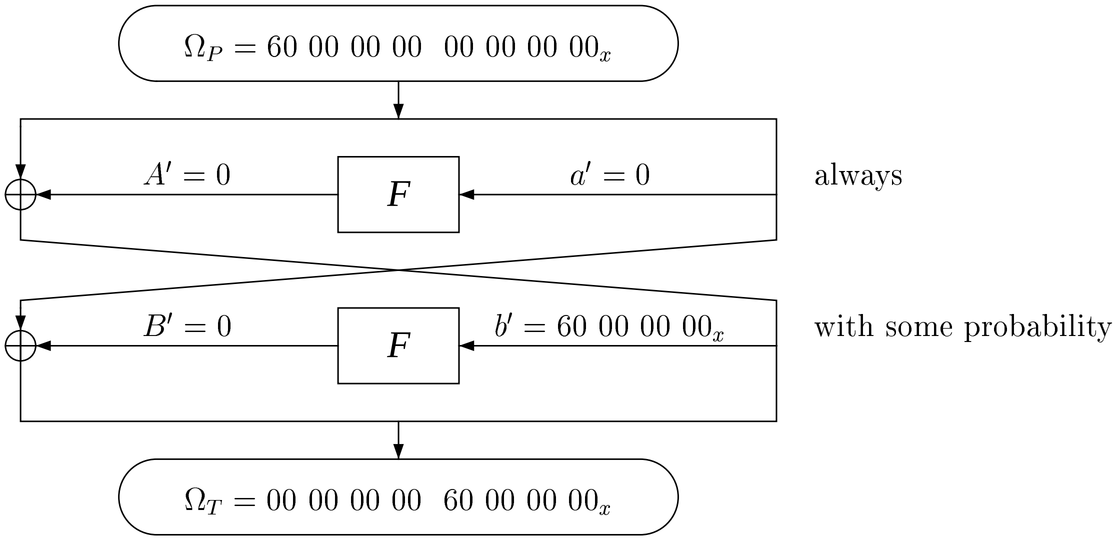
7.3.3 Modifying all the XORs in an equivalent DES description

DES has an equivalent description in which the expansion is moved to the end of the F function and all the calculations are done using 48 bits instead of 32. The cryptosystem which is the result of modifying all the XORs in this description by additions is not equivalent to the modified standard cryptosystem as described in the previous subsection. In this subsection we show that this cryptosystem is much weaker than the modified standard cryptosystem. We can save the repeated cancellation of non zero input XORs entering S3 in the previous characteristic by doing it in the first addition, since during the various rounds the data bits entering each S box are kept expanded. We get a two-round iterated characteristic with probability $\frac{1}{16}$ which is concatenated to a single occurrence of a one-round characteristic with probability $\frac{1}{16}$ at the first round. Thus an n-round characteristic with an odd n has probability $\frac{1}{16} \cdot (\frac{1}{16})^{\frac{n-1}{2}} = 2^{-2-2n}$.

The 15-round characteristic has probability 2^{-32}. A 1R-attack on the 16-round cryptosystem while counting the six key bits entering S4 in the last round has $S/N = \frac{2^6}{2^{32} \cdot 2^{-48} \cdot 2^{-42} \cdot 1} = 2^{64}$. Thus only about 2^{34} pairs are needed. The other key bits entering the last round can be found using similar characteristics. The best three characteristics have probabilities between 2^{-32} and 2^{-35}, and the attacks based on them can find 18 key bits. Therefore, 2^{37} pairs are needed to find the first 18 key bits. The remaining 2^{38} key bits can be found by exhaustive search. The total complexity of this attack is thus 2^{39}.

7.4 Random and modified S boxes

In a random S box there is a very high probability (about 0.998) that there are two different inputs that differ in the two middle input bits of an S box (which do not affect the neighboring S boxes) which have the same output. In this case there is an iterative characteristic which is (w.l.g. the S box is S1 and $S1'_I = C_x$):

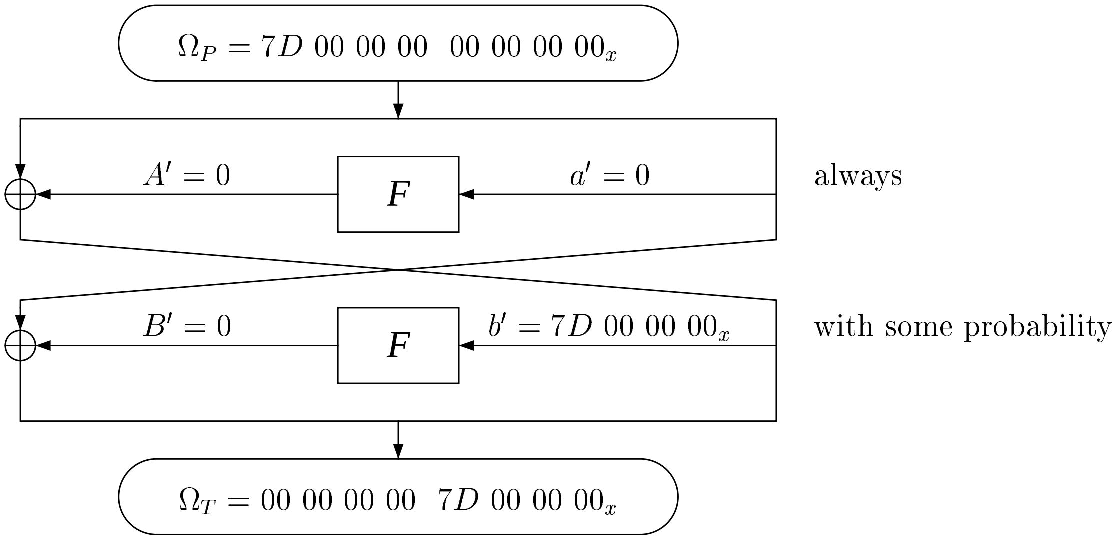


Char Prob	Prob 8 S boxes	13 rnds char prob	13 rnds S/N	Needed Pairs
$\frac{1}{32}$	1.00000	2^{-30}	2^{-2}	
$\frac{2}{32}$	1.00000	2^{-24}	2^{4}	2^{27}
$\frac{3}{32}$	0.99991	$2^{-20.5}$	$2^{7.5}$	2^{23}
$\frac{4}{32}$	0.97079	2^{-18}	2^{10}	2^{20}
$\frac{5}{32}$	0.68375	$2^{-16.1}$	$2^{11.9}$	2^{18}
$\frac{6}{32}$	0.27330	$2^{-14.5}$	$2^{13.5}$	2^{17}
$\frac{7}{32}$	0.07240	$2^{-13.2}$	$2^{14.8}$	2^{15}
$\frac{8}{32}$	0.01499	2^{-12}	2^{16}	2^{14}
$\frac{9}{32}$	0.00260	$2^{-11.0}$	$2^{17.0}$	2^{13}
$\frac{10}{32}$	0.00039	$2^{-10.1}$	$2^{17.9}$	2^{12}

Table 16. Characteristic probabilities with random S boxes.

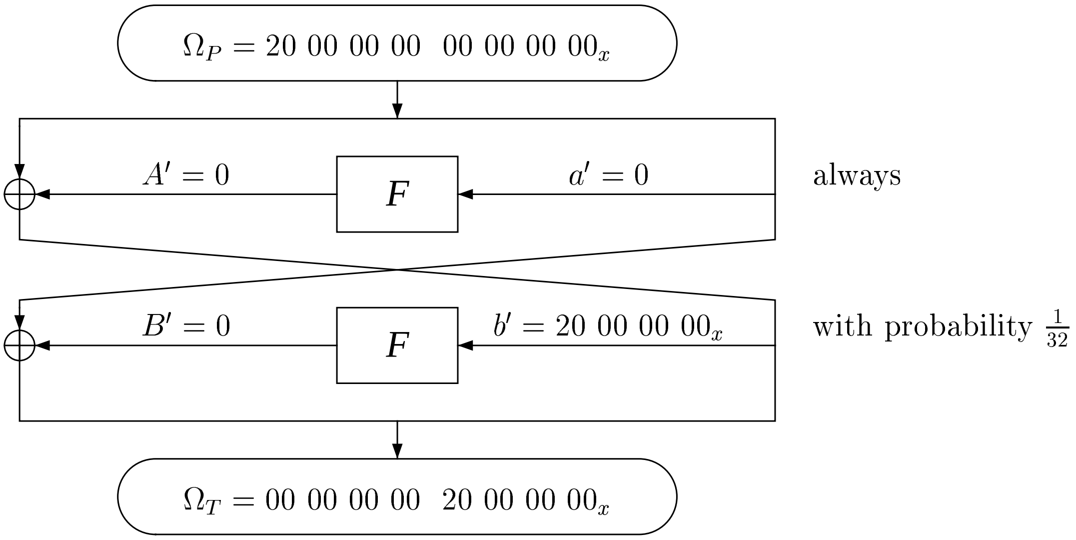
97% of the sets of eight S boxes have such iterative characteristic with probability $\frac{1}{8}$ or more. The corresponding 13-round characteristics have probability 2^{-18} for which the 3R-attack on 42 subkey bits needs 2^{20} pairs with $S/N = 2^{10}$. Table 16 describes the relationship between the probability of the characteristics, the number of pairs needed, and the probability that a set of random S boxes has such a characteristic.

In S boxes chosen as four random permutations (as in the original DES S boxes) two different inputs that differ in the private bits of one S box must have different outputs. But there is a high probability that there are two different inputs differing in the input bits of two S boxes which have the same output. In this case there is an iterative characteristic which is (w.l.g. the difference is in S1 and S2 and the differing bits of the data are by bit mask $7D\ 00\ 00\ 00_x$):



In random tests we found several attacks that use 2^{43} to 2^{47} pairs. We estimate that attacks that use this number of pairs can be found for more than 90% of the 16-round cryptosystems which use S boxes chosen as four random permutations.

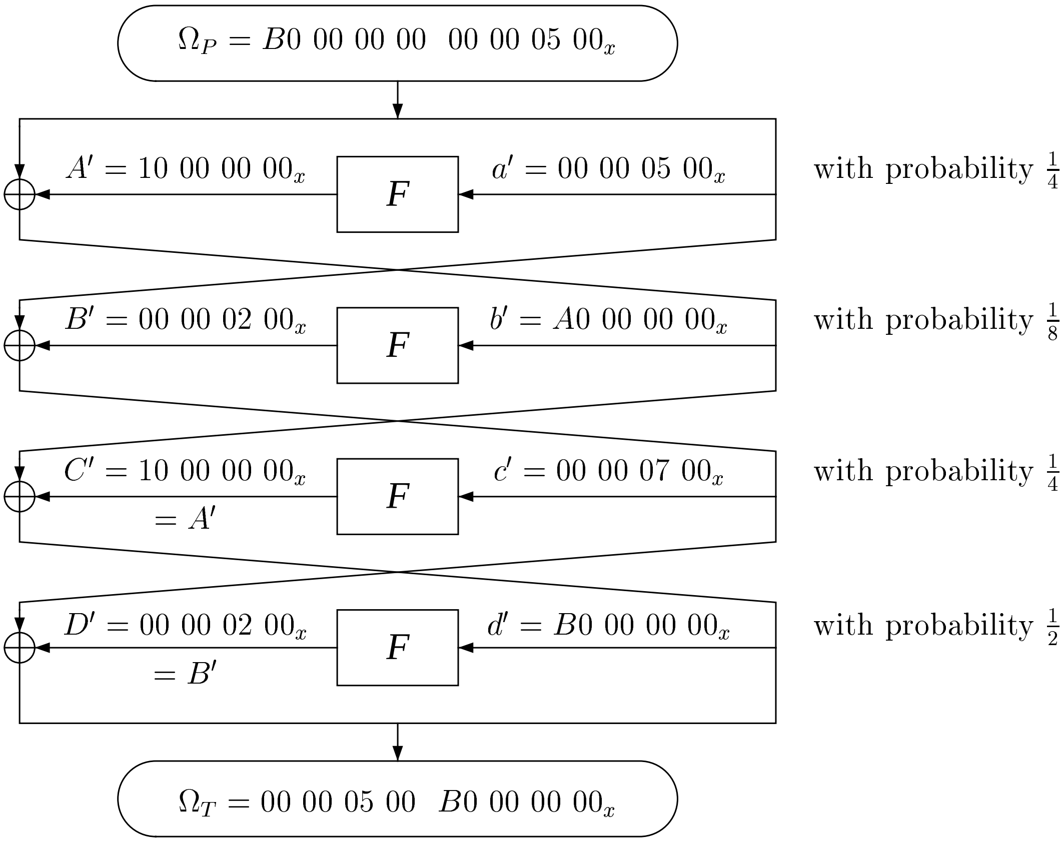
With a single modification in one entry of one of the original DES S boxes we can force this S box to have two different inputs with the same output. For example, such a modification may set the value of $S(4)$ to be equal to $S(0)$ (i.e., the third value in the first line to be equal to the first value in the first line). Therefore there are two different inputs (0 and 4) with the same output (the input XOR is 4 and the output XOR is 0). The probability of $4 \rightarrow 0$ by this S box is $\frac{1}{32}$. An iterative characteristic based on this property has probability $\frac{1}{32}$ and is (w.l.g. the difference is in S1):



Therefore the probability of the 15-round characteristic is $\frac{1}{32^7} = 2^{-35}$. Using a 1R-attack 2^{37} pairs are needed to attack the 16-round modified DES with $S/N = \frac{2^6 \cdot 2^{-35}}{4 \cdot 2^{-60}} = 2^{29}$ in order to find two indistinguishable values of the first six key bits.

7.5 Four bits to four bits S boxes

A cryptosystem similar to DES in which the E expansion is eliminated and the S boxes map four bits to four bits is quite weak. Even the cryptosystems that use permutations derived from the original S boxes are easily attacked. For example, using the first lines of the original S boxes we can find the following four-round iterative characteristic with probability $\frac{1}{256}$:



Using a 2R-attack only 2^{28} pairs are needed to break the 16-round cryptosystem. There are several additional characteristics that can be used to attack the cryptosystem with a similar amount of pairs.

8 DES with independent keys

In this section we describe an attack on DES reduced to eight rounds with independent keys and its application to DES with 16 rounds with independent keys.

8.1 Eight rounds

The attack on DES reduced to eight rounds with independent keys is basically similar to the attack on DES reduced to eight rounds described in section 5. We start by using the same algorithm to find the first 30 bits of K8 and then proceed to find the remaining bits of K8 and the bits of all the other subkeys by variants of this algorithm. The attack uses the same characteristic as in the attack described in section 5 plus 100 pairs with additional two characteristics.

After finding the first 30 bits of K8, we filter the pairs, identify the right pairs and discard all the wrong pairs (with relatively few errors). The other 18 bits of K8 cannot be found yet since we cannot assume that the subkeys are related to each other by the key scheduling algorithm. To avoid this problem we first look for bits of K7. Table 9 shows the bits in g that can be calculated for any given ciphertext (the known key bits there are irrelevant to our case). For each of the eight S boxes of the seventh round and for each of its 64 possible key values we count the number of pairs for which this key value is possible. A key value is possible for an S box in a pair if there is an input pair to the S box whose computable bits have the calculated value, the other bits have any value and the output XOR is as expected by the characteristic and the ciphertexts (by $G' = f' \oplus h' = f' \oplus r'$). The most frequent key value is likely to be the right key value. Since there is not enough data to make this key value unique we look for the set of key values with maximal counts and choose the bits that have the same value in all the set. Those bits are likely to have the right values. The other bits stay unknown. Experience has shown that the known bits of $S1_{Kh}$, $S3_{Kh}$ and $S4_{Kh}$ are at the locations denoted by '1' bits in $2F_x$, 27_x and $3C_x$ respectively. If some of these bits are unknown it is almost certainly due to a mistaken value of the known bits of K8.

By the knowledge of the subkey bits of the eighth round we can calculate several input bits of the seventh round for any ciphertext. The input to the seventh round g has missing bits that enter all the S boxes. There is one S box whose input depends just on one missing bit while the inputs of all the other S boxes depend on two missing bits at least. This S box is S1 whose input bit could be calculated if an output of S4 of the eighth round was known. To find the key bits of $S4_{Kh}$ we try all the 64 possibilities of its value for each pair, and find the key bits value by the counting method. Now each of the inputs of $S3_{Eg}$ and $S4_{Eg}$ have one missing bit: $S3_{Eg}$ could be calculated if $S1_{Oh}$ was known and $S4_{Eg}$ could be calculated if $S3_{Oh}$ was known. To find these subkey bits we try all the 128 possibilities of $S1_{Kh}$ and the missing bit of $S3_{Kg}$ and then the 128 possibilities of $S3_{Kh}$ and the missing bit of $S4_{Kg}$. Now K8 is completely known. To find K7 we repeat the algorithm of finding K7 described above with the difference that now we know all K8. Only one bit of K7 remains indistinguishable. This bit is bit number 2 of $S1_{Kg}$.

So far we used the filtered pairs. These pairs are assumed to be right pairs whose f' is as expected. They cannot help finding K6 since the input XORs of five of the S boxes are zero so this part of K6 can not be found at all. The other three S boxes have constant input XORs so there are two indistinguishable values for the subkey bits entering each S box. In order to find K6 we have to use wrong pairs for which the characteristic holds in the first three of the five rounds. From now on we use all the pairs and filter them by a different criterion in each phase of the cryptanalysis.

K6: To find K6 we decrypt two rounds of the ciphertexts and get the values of f and f^*. We assume that the first three rounds of the characteristic hold in the chosen pairs so d' is as expected with zero input XORs entering six S boxes. Thus we can calculate the output XORs of these S boxes in the sixth round by $F' = c' \oplus D' \oplus g'$. Since $c' = 0$ and S'_{Ed} is zero in the six S boxes, we get that $F' = g'$ in the output bits of these S boxes. The filtering chooses all the pairs for which f' and F' satisfy $S'_{Ef} \rightarrow S'_{Of}$ for S1, S2, S5, ..., S8. Using the resultant pairs we count on the 12 subkey bits entering S1 and S2 and the missing bit of K7 (needed for the decryption of the seventh round).

To find the other bits of K6 we filter the pairs again by using the known bits of K6 to check the output XOR of S1 and S2, and count on $S5_{Kf}$, ..., $S8_{Kf}$, a separate counting for each S box (we have a very good filtering so the S/N is high enough). In parallel we count on $S3_{Kf}$ and on $S4_{Kf}$ using the assumption that e' is as expected by the characteristic (four rounds hold) and the filter that discards any pair for which $S'_{Oe} \neq 0$ for S1, S3, ..., S8 (since only $S2'_{Ee} \neq 0$). Several possibilities are found for some of the S boxes' key bits, and the following phases are run on each one of them in parallel.

K5: We assume $c' = 0$ and $d' = b'$. Then $D' = e'$ where e and e^* are calculated by a partial decryption. S'_{Od} must be zero in the six S boxes in which $S'_{Ed} = 0$. We filter the pairs and leave only those that have $S'_{Od} = 0$. Then we count on each of the eight S boxes of the fifth round. Several possibilities can be found for some of the S_{Ke}'s. A list of all the possibilities of K5 is created and used to try each one of them in parallel in the following phases.

K4: At the second round there must be $S2'_{Eb} = S6'_{Eb} = 0$ for any pair (these S box inputs do not depend on the differing bits of the plaintexts). d and d^* are found by a partial decryption. In addition $D' = a' \oplus B' \oplus e'$ so $S2'_{Od}$ and $S6'_{Od}$ are known and there must be $S2'_{Ed} \rightarrow S2'_{Od}$ and $S6'_{Ed} \rightarrow S6'_{Od}$. If it does not hold for even one pair it is not a filtering problem. It must be a wrong value of the subkeys K5, ..., K8. A separate counting is done for each of the six S boxes S1, S2, S5, ..., S8. The counting on the other S boxes S3 and S4 is done only for pairs whose d' is as expected by the characteristic since otherwise we cannot know the value of $S3'_{Od}$ and $S4'_{Od}$ because $S3'_{Ob}$ and $S4'_{Ob}$ are unknown. Since $S3'_{Ed}$ and $S4'_{Ed}$ are constants there are two indistinguishable values for each of their keys. As usual we create a list of the possible K4 values and try them in parallel.

K3: c and c^* can be found by a partial decryption of the following rounds using K4, ..., K8. $S'_{Ea} = 0$ in all the S boxes except S2. Thus S'_{Oc} can be found for S1, S3, ..., S8 by $C' = L' \oplus A' \oplus d'$. For every pair there must be $S'_{Ec} \rightarrow S'_{Oc}$. Therefore, even if only one S box (S1 or S3, ..., S8) of one pair does not match $S'_{Ec} \rightarrow S'_{Oc}$ it must be that the values of K4, ..., K8 are wrong. If this does not happen, the counting is done in parallel for all the S boxes except S2 using all the pairs. $S2'_{Ea} \neq 0$, thus the calculation of $S2'_{Oc}$ is impossible without further assumptions. Therefore we assume that the values of A' and b' are as expected by the characteristic. The filtering discards any pair that does not have $S'_{Ob} = 0$ for S1, S2 and S5, ..., S8 using $B' = a' \oplus c' = R' \oplus c'$ (since we assume $S'_{Eb} = 0$ in these S boxes). The counting of $S2_{Kc}$ is done using the filtered pairs.

K2 and K1: The plaintext XOR used above is useless to find K2 and

K1 since all the pairs have $S2'_{Eb} = S6'_{Eb} = 0$ and for all the S boxes of the first round except S2 there is $S'_{Ea} = 0$. The key bits cannot be found at all for these S boxes. For K1 and K2 we must use another plaintext XOR. We need only 100 such pairs, which can be obtained without adding new ciphertexts by arranging some of the original ciphertexts in quartets. This plaintext XOR and the algorithm of finding K1 and K2 are very similar to the case of K1 and K2 in the four round version. See at the end of section 3 for more details.

This attack was implemented in C on a COMPAQ personal computer. It finds the key in less than two minutes with 95% success rate using 150000 pairs. Using 250000 pairs the success rate is almost 100%. The program uses 460K bytes of memory, most of it for the counting array (of size 2^{18} bytes) and the preprocessed optimization tables. The program which counts using 2^{24} memory cells finds the key using only 25000 pairs. As demonstrated by these figures, DES reduced to eight rounds with independent subkeys is almost as easy to solve as the case of dependent subkeys.

8.2 sixteen rounds

DES with independent keys with any number of rounds is vulnerable to similar attacks. Lets concentrate on DES with 16 rounds with independent keys. As we noticed in section 6 we can find eight possibilities for 18 bits of K16 using 2^{57} pairs. Three characteristics can be used to cover the subkey bits entering all the S boxes in the 16^{th} round. The three characteristics are the iterative characteristic itself, a similar iterative characteristic which is non zero in the input XORs of S3, S4 and S5 whose 15 round probability is 2^{-56} and a similar characteristic with non zero input XORs to S6, S7 and S8 whose 15 round probability is about 2^{-57}. Altogether, about 2^{59} pairs are needed to find two possibilities for the six bits entering each of the S boxes, except S2 whose bits are completely determined by two characteristics. Therefore 2^7 possibilities for K16 are found. We try in parallel all the 128 possibilities of the value of K16 and reduce the cryptanalytic problem to a DES reduced to 15 rounds. Since we know how to attack DES reduced to 15 rounds with less data in a complexity that is smaller by a factor of 2^6 then trying the 128 possibilities takes up to twice the time of finding the possibilities of K16.

Most of the possibilities are discarded during this reduction and reductions to fewer rounds are possible with even smaller complexity. Therefore the crypt-analysis of all the DES with 16 rounds with independent keys takes about 2^{61} steps and use 2^{59} pairs. Even though this is an impractical complexity bound, it is much faster than the 2^{768} complexity of exhaustive search.

9 The Generalized DES Scheme (GDES)

The Generalized DES Scheme (GDES) is an attempt to speed up DES which was suggested by Schaumuller-Bichl[16,18]. The speed up is obtained by increasing the ratio between the block size and the number of calculations of the F function.

The GDES blocks are divided into q parts of 32 bits each. The F function is calculated once per round on the rightmost part, and the result is XORed into all the other parts, which are then cyclically rotated to the right. After the last round the order of the parts is exchanged to make the encryption and decryption differ only in the order of the subkeys. The scheme is shown in figure 4, where n is the number of rounds of the GDES cryptosystem,

$$\begin{array}{rcll} B_i^{(j)} & = & B_{i-1}^{(j-1)} \oplus F(B_{i-1}^{(q)}, K_i) & \qquad j \in \{2, \ldots, q\}, i \in \{1, \ldots, n\} \\ B_i^{(1)} & = & B_{i-1}^{(q)} & \qquad i \in \{1, \ldots, n\}, \end{array}$$

$B_0 = (B_0^{(1)}, \ldots, B_0^{(q)})$ is the plaintext and $B_n^t = (B_n^{(q)}, \ldots, B_n^{(1)})$ is the cipher-text.

9.1 GDES properties

This section describes several properties of GDES.

1. In GDES with $n < q$,

$$B_0^{(i)} \oplus \varphi = B_n^{(n+i)} \qquad \forall i \in \{1, \ldots, q-n\}$$

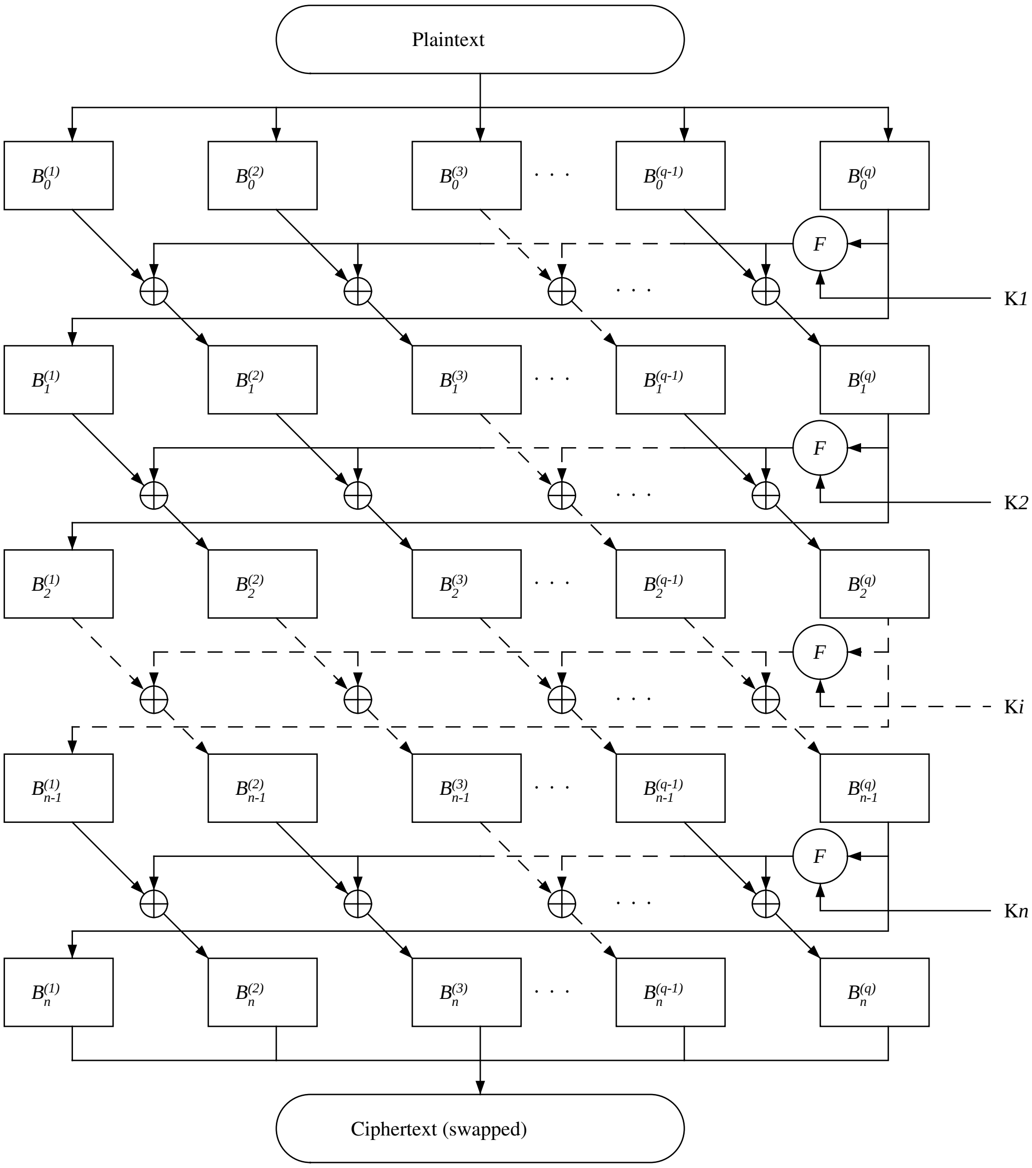


Figure 4. The Generalized DES Scheme.

where $\varphi = \bigoplus_{j=1}^{n} F(B_{j-1}^{(q)}, K_j)$.

Thus, the following formulae are satisfied for any $i, j \in \{1, \ldots, q-n\}$:

$$B_0^{(i)} \oplus B_0^{(j)} = B_n^{(n+i)} \oplus B_n^{(n+j)}$$

$$B_0^{(i)} = B_0^{(j)} \iff B_n^{(n+i)} = B_n^{(n+j)}$$

and for pairs of plaintexts for which $B_0^{(q-n+1)}, \ldots, B_0^{(q)}$ are kept constant (i.e., $B_0'^{(q-n+1)} = \ldots = B_0'^{(q)} = 0$):

$$B_0'^{(i)} = B_m'^{(m+i)} = B_n'^{(n+i)} \qquad \forall i \in \{1, \ldots, q-n\}, \forall m \in \{0, \ldots, n\}.$$

2. In GDES with $n \leq q$, any pair of encryptions in which $B_0^{(q-n+2)}, \ldots, B_0^{(q)}$ are kept constant satisfies:

$$B_0'^{(q-n+1)} = B_{n-1}'^{(q)} = B_n'^{(1)}.$$

3. For any odd q and any n the following equation is satisfied:

$$\bigoplus_{j=1}^{q} B_0^{(j)} = \bigoplus_{j=1}^{q} B_m^{(j)} = \bigoplus_{j=1}^{q} B_n^{(j)} \qquad \forall m \in \{0, \ldots, n\}.$$

4. In GDES with $n = q-1$,

$$B_0'^{(j)} = 0 \qquad \forall j \in \{2, \ldots, q\}$$

implies that

$$B_n'^{(j)} = 0 \qquad \forall j \in \{1, \ldots, q-1\}$$

and

$$B_n'^{(q)} = B_0'^{(1)}.$$

5. In GDES with $n = 2q-2$,

$$B_0'^{(1)} = \eta_1$$

$$B_0'^{(2)} = \eta_2$$

$$B_0'^{(j)} = 0 \qquad \forall j \in \{3, \ldots, q\}$$

where $\eta_1 = 44\ 08\ 00\ 00_x$ and $\eta_2 = 04\ 00\ 00\ 00_x$ or $\eta_1 = 00\ 20\ 04\ 08_x$ and $\eta_2 = 00\ 00\ 04\ 00_x$ implies that

$$B_n'^{(j)} = 0 \qquad \forall j \in \{1, \ldots, q-2\}$$

$$B_n'^{(q-1)} = \eta_2$$

$$B_n'^{(q)} = \eta_1$$

with probability $\frac{1}{16}$ since $\eta_2 \rightarrow \eta_1 \oplus \eta_2$ with probability $\frac{1}{4}$. There are additional values for η_1 and η_2 with smaller probabilities.

6. In GDES with $n = 2q - 1$,

$$B_0'^{(1)} = \psi$$

and

$$B_0'^{(j)} = 0 \qquad \forall j \in \{2, \ldots, q\}$$

(where ψ is the value used in section 6: $\psi = 19\ 60\ 00\ 00_x$) implies that

$$B_n'^{(j)} = 0 \qquad \forall j \in \{1, \ldots, q-1\}$$

and

$$B_n'^{(q)} = \psi$$

with probability about $\frac{1}{234}$. GDES with $n = lq - 1$ satisfies it for any $l \geq 2$ with probability about $\left(\frac{1}{234}\right)^{l-1}$.

9.2 Cryptanalysis of GDES

This section describes how to cryptanalyze GDES for various values of n and q. We assume that q is even (as suggested in [16,18]), but note that odd q can be attacked by variants of our technique. All the attacks find the subkeys and are independent of the key scheduling algorithm. The special case of $q = 8$ and $n = 16$ which is suggested in [16,18] as a faster and more secure alternative to DES is breakable with just six ciphertexts in a fraction of a second on a personal computer.

9.2.1 A known plaintext attack for $n = q$

Using a known plaintext attack we are given several plaintexts (each one of the form $B_0 = (B_0^{(1)},\ldots,B_0^{(q)}))$ and the corresponding ciphertexts (each one of the form $B_n^t = (B_n^{(q)},\ldots,B_n^{(1)}))$. Then

$$\bigoplus_{j=1}^{n} F(B_{j-1}^{(q)}, K_j) = \bigoplus_{j=1}^{q} \left(B_0^{(j)} \oplus B_n^{(j)}\right)$$

and for any $i \in \{1, \ldots, n\}$

$$\bigoplus_{\substack{j=1\\ j\neq i}}^{n} F(B_{j-1}^{(q)}, K_j) = B_0^{(q+1-i)} \oplus B_n^{(q+1-i)}.$$

Thus the output of the F functions is

$$F(B_{i-1}^{(q)}, K_i) = B_0^{(q+1-i)} \oplus B_n^{(q+1-i)} \oplus \bigoplus_{j=1}^{q} \left(B_0^{(j)} \oplus B_n^{(j)}\right)$$

and the input of the F functions is

$$B_{i-1}^{(q)} = B_0^{(q+1-i)} \oplus \bigoplus_{j=1}^{i-1} F(B_{j-1}^{(q)}, K_j).$$

We thus have S_E and S_O of each one of the $8n$ S boxes. As a result we get only four choices for the six subkey bits of each S box. Using two or three encryptions the choices can be filtered by leaving only the ones that appear in all the encryptions, and thus all the subkey bits can be found.

9.2.2 A chosen plaintext attack for $n = q$

Using a chosen plaintext attack with pairs whose plaintext XORs are known we can compute the input and output XORs of the F functions by the same method used in the known plaintext attack. We can thus find all the subkeys (starting with the subkey of the last round and working backwards towards the first round) using three pairs of ciphertexts with different plaintext XORs.

9.2.3 A chosen plaintext attack for $n = 2q - 1$

Using a chosen plaintext attack with pairs satisfying

$$B_0'^{(j)} = 0 \qquad \forall j \in \{2, \ldots, q\}$$

and any $B_0'^{(1)} \neq 0$, we get

$$B_{q-1}'^{(j)} = 0 \qquad \forall j \in \{1, \ldots, q-1\}$$

and

$$B_{q-1}'^{(q)} = B_0'^{(1)}.$$

The rest of the encryption is based on q rounds and thus a chosen plaintext attack similar to the one for $n = q$ can be used to find q subkeys by analyzing three ciphertext pairs.

The other $q - 1$ subkeys can be found using the above chosen plaintext attack with two additional ciphertexts.

9.2.4 A chosen plaintext attack for $n = 3q - 2$

This attack is similar to the previous one, and uses ciphertext pairs satisfying:

$$\begin{aligned} B_0'^{(1)} &= \eta_1 \\ B_0'^{(2)} &= \eta_2 \\ B_0'^{(j)} &= 0 \qquad \forall j \in \{3, \ldots, q\}. \end{aligned}$$

where η_1 and η_2 are defined in section 9.1. The right pairs are about $\frac{1}{16}$ of all the pairs. We can identify most of the wrong pairs by checking that the input XOR cannot cause the output XOR. This happens with probability about 0.8 for each S box. Thus only $0.8^{8q} = 0.16^q$ of the wrong pairs remain. When $q \geq 3$ this is less than $0.8^{8 \cdot 3} = \frac{1}{250}$ of the pairs. This excellent identification makes it possible to consider only 48 pairs, and identify the three expected occurrences of right pairs among them. We can further decrease this amount to 24 pairs by using quartets of two XOR values.

9.2.5 A chosen plaintext attack for $n = lq - 1$

This attack works for $n = lq-1$ rounds for $l \geq 3$. It is similar to the previous ones using

$$B_0'^{(1)} = \psi$$
$$B_0'^{(j)} = 0 \qquad \forall j \in \{2, \ldots, q\}.$$

The characteristic holds with probability $\left(\frac{1}{234}\right)^{l-2}$. The identification leaves $0.8^{8q-5} \cdot \left(\frac{1}{16}\right)^5$ of the wrong pairs. Thus if $0.8^{8q-5} \cdot 2^{-20} \ll \left(\frac{1}{234}\right)^{l-2}$ (i.e., $l < \frac{q}{3} + 4.7$) then the identification is excellent and only three right pairs are needed (among the $3 \cdot 234^{l-2}$ pairs considered) for counting the occurrences for each S box separately. Otherwise we can count on several S boxes simultaneously using more memory and a better S/N. Counting on the 48 bits of the subkey of the last round has

$$S/N = \frac{2^{48} \cdot 2^{-8l}}{4^8 \cdot 0.8^{8q-13} \cdot 2^{-20}} \approx 2^{48-8l+2.5q}.$$

This attack shows that any GDES which is faster than DES is also less secure than DES. GDES with $n = 8q$ rounds is just as fast as DES. Consider GDES with $n = 8q - 1$ which is slightly faster than DES. Then the usable characteristic has $7q - 1$ rounds and six repetitions of the iterative characteristic. Thus its probability is about $\left(\frac{1}{234}\right)^6 \approx 2^{-48}$. Counting on all the 48 bits of the subkey of the last round has

$$S/N = \frac{2^{48} \cdot 2^{-48}}{4^8 \cdot 0.8^{8q-13} \cdot 2^{-20}} \approx 2^{2.5q}.$$

Thus about 4–8 right pairs are needed, giving a total of $8 \cdot 2^{48} = 2^{51}$ pairs. This complexity decreases rapidly when we try to make GDES even faster by making n substantially smaller than $8q$.

9.2.6 The actual breaking algorithm for $n = 2q$

The breaking algorithm for the recommended case of $n = 2q$ needs six ciphertexts with particular plaintext XOR values. In this section we describe an attack on the extension of GDES which uses independent subkeys, which needs 16 encryptions. Note that for $n = 16$ and $q = 8$ the extended attack uses only $16 \cdot 64 = 1024$ ciphertext bits in order to find the $16 \cdot 48 = 768$ key bits, giving an almost optimal ratio of 1.33 ciphertext bits per key bit.

The attacker chooses a random plaintext P, encrypts the following 16 plaintexts, and uses only the resultant ciphertexts.

- The plaintext P itself.
- The nine plaintexts obtained from P by XORing $66\ 00\ 00\ 00_x$, $60\ 60\ 00\ 00_x$, $60\ 00\ 60\ 00_x$, $60\ 00\ 00\ 60_x$, $60\ 00\ 00\ 06_x$, $9E\ 5F\ AC\ 7D_x$, $F7\ A5\ 35\ C7_x$, $7A\ FA\ 78\ D5_x$ and $21\ 22\ E3\ 2C_x$ into $B_0^{(1)}$ (the first 32 bits of P).
- The six plaintexts obtained from P by XORing $A6\ BD\ EF\ B7_x$, $F4\ F3\ 82\ 3C_x$, $4F\ 5C\ 37\ 51_x$, $2B\ 76\ 7A\ DB_x$, $5A\ 19\ F9\ 68_x$ and $33\ EE\ DD\ FF_x$ into all the $B_0^{(i)}$ blocks.

These XOR values are chosen by the following criteria:

1. The first plaintext is the randomly chosen basis for the differential attack.
2. Five plaintexts have the maximal number of unchanged inputs to S boxes in the q^{th} round compared to P and to each other. For the values chosen at least five of the inputs to each S box are unchanged, which makes it possible to find the subkey of the last round.
3. Four other plaintexts have a maximal difference in the S boxes of the q^{th} round. This is used to find the subkeys of the $q + 1^{\text{th}}$ and all the subsequent rounds (There is not enough variability in the previous values to find all those subkeys).
4. Six plaintexts have a maximal difference in the S boxes of the first q rounds. This makes it possible to find the first q subkeys.

The cryptanalytic algorithm is as follows. At first the attacker tries to find the subkey of the last round. Each one of the 15 pairs consisting of P and the first five plaintexts has a different set of six S boxes whose input XORs in $B_0^{(1)}$ are zero. All the other $B_0^{(i)}, i \in \{2, \ldots, q\}$ have input XORs which are trivially zero. Thus each one of the first $q-1$ F functions have the same input and output values in all the pairs. In each pair the q^{th} F function has zero input and output XORs in six of the eight S boxes. Using this knowledge we get the output XOR of these six S boxes in the last ($2q^{\text{th}}$) round by the formula:

$$F'(B_{n-1}^{(q)}, K_n) = \bigoplus_{j=2}^{q} B_n'^{(j)}.$$

The input XOR is easily computed as $B_{n-1}'^{(q)} = B_n'^{(1)}$ and the input itself is $B_n^{(1)}$. Now we try all the possible key bits for each S box separately and check that for the given input XOR we get the given output XOR value. For each S box there are at least five pairs which can distinguish values of the key bits. The (almost certainly unique) value suggested by all the pairs is the key of the corresponding S box. Therefore, the whole subkey of the last round is found. Now a decryption of the last round can be done reducing the cryptosystem to $2q-1$ rounds.

Note that if the subkeys are derived by the DES key scheduling algorithm then 48 bits out of the 56 key bits are known at this point. The others can be easily found by trying all the 256 possibilities of the missing eight key bits. We thus proceed to analyze the case of independent subkeys.

In the following $q-1$ rounds we get the input and the input XOR of the F function from the (partially decrypted) ciphertexts. The input XOR is calculated by the formula:

$$F'(B_{r-1}^{(q)}, K_r) = B_0'^{(1)} \oplus \bigoplus_{j=2}^{q} B_r'^{(j)}$$

where r is the round number ($r \in \{q+1, \ldots, 2q-1\}$). In this case the first ten ciphertexts are used. The additional four ciphertexts are needed primarily to find $\mathrm{K}(q+1)$ since in the first six encryptions there are too many zero XOR bits and more variety is needed. These added ciphertexts do not help in the n^{th} round since there we want the output XORs of the S boxes in the q^{th} round to be zero.

In the remaining q rounds we use all the 16 ciphertexts. The additional ciphertexts have non zero differences in all the S boxes in all the rounds, whereas the first ten had a constant value during the first $q-1$ rounds. The input XOR is calculated by the formula:

$$F'(B_{r-1}^{(q)}, K_r) = \varphi \oplus \bigoplus_{j=2}^{q} B_r'^{(j)}$$

where r is the round number ($r \in \{1, \ldots, q\}$) and φ is

$$\varphi = \begin{cases} B_0'^{(1)}, & \text{if } r < q; \\ B_0'^{(2)}, & \text{if } r = q. \end{cases}$$

9.2.7 Conclusions

GDES with $n = q = 8$ is breakable using a known plaintext attack with three ciphertexts. With a key scheduling similar to DES, GDES is vulnerable to a known plaintext attack when $n = q + 1$ as well.

GDES with $q = 8$ and $n = 16$ was suggested in [16,18]. The 15-round variant is easily breakable using the $n = 2q-1$ attack with three ciphertexts. The 16-round version is breakable using the extension to $n = 2q$ with six ciphertexts in 0.2 seconds on a COMPAQ personal computer. If independent keys are used then it is breakable with 16 ciphertexts in three seconds on the same computer.

GDES with $q = 8$ and $n = 22$ is breakable using the $n = 3q - 2$ attack with 48 ciphertexts (24 pairs). GDES with $q = 8$ and $n = 31$ is breakable using the $n = 4q-1$ attack with 250000 pairs and $S/N = \frac{2^{18}}{234^2 \cdot 0.8^{13}} \approx 2^7$ with memory of size 2^{18}. Even GDES with $q = 8$ and $n = 63$ is weaker than DES and is breakable using 2^{52} ciphertexts. In general, any GDES which is faster than DES is also less secure than DES.

10 Non-differential attacks on DES reduced to few rounds

In this section we describe several novel attacks on DES reduced to 3–6 rounds which are not based on the ciphertext pair paradigm. These attacks are of three kinds: ciphertext only attacks, known plaintext attacks and statistical known plaintext attacks.

10.1 Ciphertext only attacks

10.1.1 A three-round attack

This attack assumes that the eight plaintext bytes are ASCII characters whose most significant bits are zeroes. The Initial Permutation (IP) moves the most significant bits of all these bytes into a single byte. This byte is the fifth byte of the permuted plaintext which is the first byte of the right half. Given a ciphertext $T = (l, r)$ we can easily calculate eight bits of the output of the second round by $B = a \oplus c = R \oplus r$. From table 26 we see that these eight bits are the output of seven S boxes in the second round (where two of them come from S5). The attack is as follows:

1. We try all the possibilities of the key bits entering S5 in the second round and all the key bits entering the six S boxes S1, S2, S3, S4, S6 and S8 in the third round. Their output bits are XORed with the data bits entering S5 in the second round. Three bits are counted in both rounds and thus 39 bits are exhaustively tried.
2. Using the tried key bits and any ciphertext we find the output of the six S boxes in the third round and the input and output of S5 in the second round.
3. We compare the two computed output bits of S5 in the second round to their expected value. If they are different then the 39 key bits are wrong. A quarter of the tried keys have the expected value. By trying

additional ciphertexts we can discard more key values. We stop when one candidate remains.

Since we start with 2^{39} possible keys and only $\frac{1}{4}$ of them survive each test, we need about $\log_4 2^{39} = 19.5$ ciphertexts. When the correct 39 key bits are determined, we can exhaustively try all the possible values of the remaining 17 bits by checking whether the decoded plaintexts are ASCII characters. The attack thus needs a total of 2^{39} steps and 20 ciphertexts to break DES reduced to three rounds.

10.1.2 Another three-round attack

In this attack we assume that the plaintext bytes belong to a smaller set in which the three most significant bits are constant. Such sets are the ASCII capital letters, the ASCII lower case letters and the ASCII digits. The three most significant bits of all the eight plaintext bytes are moved to three bytes by the initial permutation. These three bytes are the first byte of the left half and the first and second bytes of the right half. Since the first and second bytes of the right half are constant in all the plaintext blocks, the inputs of S2 and S3 in the first round are constant and thus their outputs are constant as well. We can calculate the output bits of the third round by the equation:

$$C = L \oplus A \oplus l. \tag{2}$$

Two bits of the eight constant bits in L have corresponding constant bits in A: one of them is an output of S2 and the other is an output of S3 (see table 26). Since l is known, the two bits in C are known up to a XOR with a constant. These bits are outputs of S2 and S3. Trying all the 64 possibilities of the key bits entering S2 in the third round, we can check that in any pair of ciphertexts the output bit of S2 satisfies $C_1 \oplus l_1 = C_2 \oplus l_2$. Since half the keys satisfy this condition, we need about $1 + \log_2 64 = 7$ ciphertexts to find the six key bits entering S2 in the third round. The same ciphertexts can be used to find the six key bits entering S3 in the third round. This leaves 44 unknown key bits, which can be found in 2^{44} steps with seven ciphertexts.

10.1.3 A four-round attack

This attack is an extension of the previous three-round attack and assumes (as there) that the three most significant bits of each plaintext byte are constant. In this attack two bits of C are found by the equation:

$$C = L \oplus A \oplus d = L \oplus A \oplus r$$

which is similar to equation 2. Then two output bits (one in S2 and one in S3 in the third round) are known up to a constant. We try all the possible key values of the six key bits of S2 (or similarly S3) in the third round and all the possible key values of the six S boxes in the fourth round whose output bits are XORed with the data bits entering S2 (or S3) in the third round. We try a total of 36 key bits entering the fourth round and six key bits entering the third round, but five bits are common (six when using S3) and thus we have to try 2^{37} possible key values. We need about $1 + \log_2 2^{37} = 1 + 37 = 38$ ciphertexts to make the computed key unique.

10.2 Known plaintext attacks

10.2.1 A three-round attack

The DES key scheduling algorithm divides the 56 key bits into two halves. Each half has 28 bits, and supplies the key bits to four S boxes.

Consider DES reduced to three rounds with a single known plaintext/ciphertext pair. The exclusive-or value of the output of the first round and the third round is known by the equation:

$$A \oplus C = L \oplus l.$$

We first try all the 2^{28} possibilities of one half of the key. Each candidate makes it possible to compute the output of four S boxes in the first round and the output of the same S boxes in the third round. We know their expected exclusive-or value. Since the value has 16 bits, only about 2^{-16} of the candidates survive this test. Thus we get about 2^{12} possibilities for the first 28 bits of the key. In a similar way we get about 2^{12} possibilities for the other 28 bits of the key. Therefore we find about $2^{12} \cdot 2^{12} = 2^{24}$ possibilities for the full key, which can be exhaustively searched. The complexity of this algorithm is about 2^{29}, and can be reduced to about 2^{21} by choosing the key bits entering each S box sequentially rather than in parallel, and discarding partial keys as soon as they lead to a contradiction.

10.3 Statistical known plaintext attacks

10.3.1 A three-round attack

In this attack we use the fact that in a pairs XOR distribution table, if we know that the output XOR is zero then the input XOR is zero with probability $\frac{1}{4}$. Given the plaintext and the ciphertext of an encryption we can easily calculate $A \oplus C = L \oplus l$. Then the following algorithm is used for each S box. Choose only the encryptions whose output XOR from this S box is zero ($\frac{1}{16}$ of the encryptions): $S_{Oa} \oplus S_{Oc} = 0$. If $S_{Ia} \oplus S_{Ic} = 0$ then the corresponding bits of $a \oplus c = R \oplus r$ equal $S_{Ka} \oplus S_{Kc}$. We count the number of occurrences of each such XOR value. The right value is suggested by about $\frac{1}{4}$ of the encryptions. Each other value is suggested by about $\frac{3}{4} \cdot \frac{1}{63}$ of the encryptions. The value that appears most frequently is likely to be the value of $S_{Ka} \oplus S_{Kc}$. This algorithm is used for each S box and thus we find $8 \cdot 6 = 48$ bits that are XORs of the actual key bits. Then trying 2^8 possibilities we can find the full 56 bit key. We need about four occurrences of the right value of the key XOR for each S box, i.e., total of about $4 \cdot 4 \cdot 16 = 256$ plaintext/ciphertext pairs.

10.3.2 A four-round attack

In this attack we use the fact that for all the S boxes there is a weak correlation between the value of the XOR of the four output bits and the value of bit number 2 of the input. In particular, for every two inputs of an S box, if the XOR of the four output bits of the first input equals the corresponding value of the second input then both bits 2 of the input are equal with a certain probability. This probability is different for each S box and varies between 0.56 and 0.70.

Given the plaintext and the ciphertext of an encryption we can easily calculate $S_{Oa} \oplus S_{Oc}$ by

$$A \oplus C = L \oplus l.$$

Then the following algorithm is used separately for each S box. For every encryption calculate the (single bit) XOR of the four output bits of the first round and the four output bits of the third round by the above equation. This value is likely to be equal to the XOR of bit number 2 of the inputs of the S box in these two rounds. S_{Ia} is known up to a XOR with the key (by the plaintext) and thus bit number 2 of the input in the third round is known up to a XOR with a constant with a high probability. This constant is the XOR of the corresponding bit number 2 in $S_{Ka} \oplus S_{Kc}$. Thus by $D = l \oplus c$ we find the corresponding output bit in the fourth round up to that constant with a high probability. We try all the 64 possibilities of the key bits entering the corresponding S box in the fourth round and the two possibilities of the constant and verify that the specific output bit of the S box equals its expected value. The right key value is counted in about 56%–70% of the encryptions, depending on the exact S box. Any wrong key value is counted in about half of the encryptions. The key value which is counted most frequently is likely to be the right value. This attack finds a total of seven bits: six of them are actual key bits and the seventh is a XOR of two key bits.

The attack obtains the best results when the probability is as high as possible. To increase the probability we use only encryptions with specific values of $S_{Oa} \oplus S_{Oc}$ which maximize this probability. For instance, when $S5_{Oa} \oplus S5_{Oc} = 0$ this probability is about 0.81. There is a tradeoff between the number of allowed values and the corresponding probability. As the number of allowed values increases, the probability decreases so we need more data to carry out the attack. However, as the number of allowed values decreases we need more data to make the occurrence of these values sufficiently probable. Table 17 describes the best tradeoff achievable by this attack. To make the best use of this attack it is advisable to use about 200 plaintext/ciphertext pairs, for which we can find almost 28 key bit, and search exhaustively for the (about 2^{28}) remaining possibilities of the key. Us-

By	Finding	Average	Best tradeoff	
S box	Bits of	Probability	Values	Encryptions
S1	S4	66%	16	75
S2	S8	57%	8	195
S3	S1	58%	7	240
S4	S2	56%	9	370
S5	S1	70%	16	50
S6	S8	61%	8	135
S7	S5	60%	14	210
S8	S6	63%	12	120

Table 17. Number of encryptions needed to find S_{Kd} for each S box.

ing about 370 plaintext/ciphertext pairs we can find almost 42 key bits and search exhaustively for the (about 2^{14}) remaining possibilities of the key.

10.3.3 A five-round attack

This five-round attack is similar to the previous algorithm. We can calculate $B \oplus D = R \oplus r$. Then an input XOR bit of the S box in the second and fourth round is known with probability between 0.56 and 0.70. As a result, an output bit of $A \oplus E$ is known up to a XOR with a constant by $L \oplus A = b$ and $d \oplus E = l$ and thus

$$A \oplus E = b \oplus d \oplus L \oplus l.$$

Using a counting method that counts on the key bits entering the same S box in the first round, the key bits entering the corresponding S box in the fifth round, and the constant, we can find 13 bits of the key: six of them are actual key bits from the first round, six are actual key bits from the fifth round, and the thirteenth is a XOR of two key bits. The amount of data needed to find these 13 key bits is about the same as in the previous attack.

10.3.4 A six-round attack

This attack is again similar to the attack on five rounds, but we also have to count all the possibilities of the 36 subkey bits of the sixth round which enter S boxes whose output bits enter the counted S box in the fifth round by the P permutation. In total we count on 49 bits. The total complexity of this attack is about 2^{55}–2^{56} but the basic operation (which is similar to a single application of the F function) is much simpler than an encryption, and thus the time needed is marginally faster than exhaustive search.

A DES tables

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

Table 18. S1 table.

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

Table 19. S2 table.

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

Table 20. S3 table.

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

Table 21. S4 table.

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

Table 22. S5 table.

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

Table 23. S6 table.

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

Table 24. S7 table.

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

Table 25. S8 table.

From				To			
Bit no.	S box & bit		Bit Mask (hex)	Bit no.	S box & bit	Bit Mask (hex)	Missing S box
1	S1	1	80 00 00 00	9	S2.6 S3.2	00 80 00 00	S7
2		2	40 00 00 00	17	S4.6 S5.2	00 00 80 00	
3		3	20 00 00 00	23	S6.4	00 00 02 00	
4		4	10 00 00 00	31	S8.4	00 00 00 02	
5	S2	1	08 00 00 00	13	S3.6 S4.2	00 08 00 00	S6
6		2	04 00 00 00	28	S7.5 S8.1	00 00 00 10	
7		3	02 00 00 00	2	S1.3	40 00 00 00	
8		4	01 00 00 00	18	S5.3	00 00 40 00	
9	S3	1	00 80 00 00	24	S6.5 S7.1	00 00 01 00	S1
10		2	00 40 00 00	16	S4.5 S5.1	00 01 00 00	
11		3	00 20 00 00	30	S8.3	00 00 00 04	
12		4	00 10 00 00	6	S2.3	04 00 00 00	
13	S4	1	00 08 00 00	26	S7.3	00 00 00 40	S2
14		2	00 04 00 00	20	S5.5 S6.1	00 00 10 00	
15		3	00 02 00 00	10	S3.3	00 40 00 00	
16		4	00 01 00 00	1	S8.6 S1.2	80 00 00 00	
17	S5	1	00 00 80 00	8	S2.5 S3.1	01 00 00 00	S8
18		2	00 00 40 00	14	S4.3	00 04 00 00	
19		3	00 00 20 00	25	S6.6 S7.2	00 00 00 80	
20		4	00 00 10 00	3	S1.4	20 00 00 00	
21	S6	1	00 00 08 00	4	S1.5 S2.1	10 00 00 00	S4
22		2	00 00 04 00	29	S7.6 S8.2	00 00 00 08	
23		3	00 00 02 00	11	S3.4	00 20 00 00	
24		4	00 00 01 00	19	S5.4	00 00 20 00	
25	S7	1	00 00 00 80	32	S8.5 S1.1	00 00 00 01	S5
26		2	00 00 00 40	12	S3.5 S4.1	00 10 00 00	
27		3	00 00 00 20	22	S6.3	00 00 04 00	
28		4	00 00 00 10	7	S2.4	02 00 00 00	
29	S8	1	00 00 00 08	5	S1.6 S2.2	08 00 00 00	S3
30		2	00 00 00 04	27	S7.4	00 00 00 20	
31		3	00 00 00 02	15	S4.4	00 02 00 00	
32		4	00 00 00 01	21	S5.6 S6.2	00 00 08 00	

Table 26. The P permutation table.

B The pairs XOR distribution tables of the S boxes

Input	Output XOR															
XOR	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	A_x	B_x	C_x	D_x	E_x	F_x
0_x	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_x	0	0	0	6	0	2	4	4	0	10	12	4	10	6	2	4
2_x	0	0	0	8	0	4	4	4	0	6	8	6	12	6	4	2
3_x	14	4	2	2	10	6	4	2	6	4	4	0	2	2	2	0
4_x	0	0	0	6	0	10	10	6	0	4	6	4	2	8	6	2
5_x	4	8	6	2	2	4	4	2	0	4	4	0	12	2	4	6
6_x	0	4	2	4	8	2	6	2	8	4	4	2	4	2	0	12
7_x	2	4	10	4	0	4	8	4	2	4	8	2	2	2	4	4
8_x	0	0	0	12	0	8	8	4	0	6	2	8	8	2	2	4
9_x	10	2	4	0	2	4	6	0	2	2	8	0	10	0	2	12
A_x	0	8	6	2	2	8	6	0	6	4	6	0	4	0	2	10
B_x	2	4	0	10	2	2	4	0	2	6	2	6	6	4	2	12
C_x	0	0	0	8	0	6	6	0	0	6	6	4	6	6	14	2
D_x	6	6	4	8	4	8	2	6	0	6	4	6	0	2	0	2
E_x	0	4	8	8	6	6	4	0	6	6	4	0	0	4	0	8
F_x	2	0	2	4	4	6	4	2	4	8	2	2	2	6	8	8
10_x	0	0	0	0	0	0	2	14	0	6	6	12	4	6	8	6
11_x	6	8	2	4	6	4	8	6	4	0	6	6	0	4	0	0
12_x	0	8	4	2	6	6	4	6	6	4	2	6	6	0	4	0
13_x	2	4	4	6	2	0	4	6	2	0	6	8	4	6	4	6
14_x	0	8	8	0	10	0	4	2	8	2	2	4	4	8	4	0
15_x	0	4	6	4	2	2	4	10	6	2	0	10	0	4	6	4
16_x	0	8	10	8	0	2	2	6	10	2	0	2	0	6	2	6
17_x	4	4	6	0	10	6	0	2	4	4	4	6	6	6	2	0
18_x	0	6	6	0	8	4	2	2	2	4	6	8	6	6	2	2
19_x	2	6	2	4	0	8	4	6	10	4	0	4	2	8	4	0
$1A_x$	0	6	4	0	4	6	6	6	6	2	2	0	4	4	6	8
$1B_x$	4	4	2	4	10	6	6	4	6	2	2	4	2	2	4	2
$1C_x$	0	10	10	6	6	0	0	12	6	4	0	0	2	4	4	0
$1D_x$	4	2	4	0	8	0	0	2	10	0	2	6	6	6	14	0
$1E_x$	0	2	6	0	14	2	0	0	6	4	10	8	2	2	6	2
$1F_x$	2	4	10	6	2	2	2	8	6	8	0	0	0	4	6	4
20_x	0	0	0	10	0	12	8	2	0	6	4	4	4	2	0	12
21_x	0	4	2	4	4	8	10	0	4	4	10	0	4	0	2	8
22_x	10	4	6	2	2	8	2	2	2	2	6	0	4	0	4	10
23_x	0	4	4	8	0	2	6	0	6	6	2	10	2	4	0	10
24_x	12	0	0	2	2	2	2	0	14	14	2	0	2	6	2	4
25_x	6	4	4	12	4	4	4	10	2	2	2	0	4	2	2	2
26_x	0	0	4	10	10	10	2	4	0	4	6	4	4	4	2	0
27_x	10	4	2	0	2	4	2	0	4	8	0	4	8	8	4	4
28_x	12	2	2	8	2	6	12	0	0	2	6	0	4	0	6	2
29_x	4	2	2	10	0	2	4	0	0	14	10	2	4	6	0	4
$2A_x$	4	2	4	6	0	2	8	2	2	14	2	6	2	6	2	2
$2B_x$	12	2	2	2	4	6	6	2	0	2	6	2	6	0	8	4
$2C_x$	4	2	2	4	0	2	10	4	2	2	4	8	8	4	2	6
$2D_x$	6	2	6	2	8	4	4	4	2	4	6	0	8	2	0	6
$2E_x$	6	6	2	2	0	2	4	6	4	0	6	2	12	2	6	4
$2F_x$	2	2	2	2	2	6	8	8	2	4	4	6	8	2	4	2
30_x	0	4	6	0	12	6	2	2	8	2	4	4	6	2	2	4
31_x	4	8	2	10	2	2	2	2	6	0	0	2	2	4	10	8
32_x	4	2	6	4	4	2	2	4	6	6	4	8	2	2	8	0
33_x	4	4	6	2	10	8	4	2	4	0	2	2	4	6	2	4
34_x	0	8	16	6	2	0	0	12	6	0	0	0	0	8	0	6
35_x	2	2	4	0	8	0	0	0	14	4	6	8	0	2	14	0
36_x	2	6	2	2	8	0	2	2	4	2	6	8	6	4	10	0
37_x	2	2	12	4	2	4	4	10	4	4	2	6	0	2	2	4
38_x	0	6	2	2	2	0	2	2	4	6	4	4	4	6	10	10
39_x	6	2	2	4	12	6	4	8	4	0	2	4	2	4	4	0
$3A_x$	6	4	6	4	6	8	0	6	2	2	6	2	2	6	4	0
$3B_x$	2	6	4	0	0	2	4	6	4	6	8	6	4	4	6	2
$3C_x$	0	10	4	0	12	0	4	2	6	0	4	12	4	4	2	0
$3D_x$	0	8	6	2	2	6	0	8	4	4	0	4	0	12	4	4
$3E_x$	4	8	2	2	2	4	4	14	4	2	0	2	0	8	4	4
$3F_x$	4	8	4	2	4	0	2	4	4	2	4	8	8	6	2	2

Table 27. The pairs XOR distribution table of S1.

Input XOR	Output XOR															
	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	A_x	B_x	C_x	D_x	E_x	F_x
0_x	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_x	0	0	0	4	0	2	6	4	0	14	8	6	8	4	6	2
2_x	0	0	0	2	0	4	6	4	0	0	4	6	10	10	12	6
3_x	4	8	4	8	4	6	4	2	4	2	2	4	6	2	0	4
4_x	0	0	0	0	0	6	0	14	0	6	10	4	10	6	4	4
5_x	2	0	4	8	2	4	6	6	2	0	8	4	2	4	10	2
6_x	0	12	6	4	6	4	6	2	2	10	2	8	2	0	0	0
7_x	4	6	6	4	2	4	4	2	6	4	2	4	4	6	0	6
8_x	0	0	0	4	0	4	0	8	0	10	16	6	6	0	6	4
9_x	14	2	4	10	2	8	2	6	2	4	0	0	2	2	2	4
A_x	0	6	6	2	10	4	10	2	6	2	2	4	2	2	4	2
B_x	6	2	2	0	2	4	6	2	10	2	0	6	6	4	4	8
C_x	0	0	0	4	0	14	0	10	0	6	2	4	4	8	6	6
D_x	6	2	6	2	10	2	0	4	0	10	4	2	8	2	2	4
E_x	0	6	12	8	0	4	2	0	8	2	4	4	6	2	0	6
F_x	0	8	2	0	6	6	8	2	4	4	4	6	8	0	4	2
10_x	0	0	0	8	0	4	10	2	0	2	8	10	0	10	6	4
11_x	6	6	4	6	4	0	6	4	8	2	10	2	2	4	0	0
12_x	0	6	2	6	2	4	12	4	6	4	0	4	4	6	2	2
13_x	4	0	4	0	8	6	6	0	0	2	0	6	4	8	2	14
14_x	0	6	6	4	10	0	2	12	6	2	2	2	4	4	2	2
15_x	6	8	2	0	8	2	0	2	2	2	2	2	2	14	10	2
16_x	0	8	6	4	2	2	4	2	6	4	6	2	6	0	6	6
17_x	6	4	8	6	4	4	0	4	6	2	4	4	4	2	4	2
18_x	0	6	4	6	10	4	0	2	4	8	0	0	4	8	2	6
19_x	2	4	6	4	4	2	4	2	6	4	6	8	0	6	4	2
$1A_x$	0	6	8	4	2	4	2	2	8	2	2	6	2	4	4	8
$1B_x$	0	6	4	4	0	12	6	4	2	2	2	4	4	2	10	2
$1C_x$	0	4	6	6	12	0	4	0	10	2	6	2	0	0	10	2
$1D_x$	0	6	2	2	6	0	4	16	4	4	2	0	0	4	6	8
$1E_x$	0	4	8	2	10	6	6	0	8	4	0	2	4	4	0	6
$1F_x$	4	2	6	6	2	2	2	4	8	6	10	6	4	0	0	2
20_x	0	0	0	2	0	12	10	4	0	0	0	2	14	2	8	10
21_x	0	4	6	8	2	10	4	2	2	6	4	2	6	2	0	6
22_x	4	12	8	4	2	2	0	0	2	8	8	6	0	6	0	2
23_x	8	2	0	2	8	4	2	6	4	8	2	2	6	4	2	4
24_x	10	4	0	0	0	4	0	2	6	8	6	10	8	0	2	4
25_x	6	0	12	2	8	6	10	0	0	8	2	6	0	0	2	2
26_x	2	2	4	4	2	2	10	14	2	0	4	2	2	4	6	4
27_x	6	0	0	2	6	4	2	4	4	4	8	4	8	0	6	6
28_x	8	0	8	2	4	12	2	0	2	6	2	0	6	2	0	10
29_x	0	2	4	10	2	8	6	4	0	10	0	2	10	0	2	4
$2A_x$	4	0	4	8	6	2	4	4	6	6	2	6	2	2	4	4
$2B_x$	2	2	6	4	0	2	2	6	2	8	8	4	4	4	8	2
$2C_x$	10	6	8	6	0	6	4	4	4	2	4	4	0	0	2	4
$2D_x$	2	2	2	4	0	0	0	2	8	4	4	6	10	2	14	4
$2E_x$	2	4	0	2	10	4	2	0	2	2	6	2	8	8	10	2
$2F_x$	12	4	6	8	2	6	2	8	0	4	0	2	0	8	2	0
30_x	0	4	0	2	4	4	8	6	10	6	2	12	0	0	0	6
31_x	0	10	2	0	6	2	10	2	6	0	2	0	6	6	4	8
32_x	8	4	6	0	6	4	4	8	4	6	8	0	2	2	2	0
33_x	2	2	6	10	2	0	0	6	4	4	12	8	4	2	2	0
34_x	0	12	6	4	6	0	4	4	4	0	4	6	4	2	4	4
35_x	0	12	4	6	2	4	4	0	10	0	0	8	0	8	0	6
36_x	8	2	4	0	4	0	4	2	0	8	4	2	6	16	2	2
37_x	6	2	2	2	6	6	4	8	2	2	6	2	2	2	4	8
38_x	0	8	8	10	6	2	2	0	4	0	4	2	4	0	4	10
39_x	0	2	0	0	8	0	10	4	10	0	8	4	4	4	4	6
$3A_x$	4	0	2	8	4	2	2	2	4	8	2	0	4	10	10	2
$3B_x$	16	4	4	2	8	2	2	6	4	4	4	2	0	2	2	2
$3C_x$	0	2	6	2	8	4	6	0	10	2	2	4	4	10	4	0
$3D_x$	0	16	10	2	4	2	4	2	8	0	0	8	0	6	2	0
$3E_x$	4	4	0	10	2	4	2	14	4	2	6	6	0	0	6	0
$3F_x$	4	0	0	2	0	8	2	4	0	2	4	4	4	14	10	6

Table 28. The pairs XOR distribution table of S2.

Input XOR	Output XOR															
	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	A_x	B_x	C_x	D_x	E_x	F_x
0_x	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_x	0	0	0	2	0	4	2	12	0	14	0	4	8	2	6	10
2_x	0	0	0	2	0	2	0	8	0	4	12	10	4	6	8	8
3_x	8	6	10	4	8	6	0	6	4	4	0	0	0	4	2	2
4_x	0	0	0	4	0	2	4	2	0	12	8	4	6	8	10	4
5_x	6	2	4	8	6	10	6	2	2	8	2	0	2	0	4	2
6_x	0	10	6	6	10	0	4	12	2	4	0	0	6	4	0	0
7_x	2	0	0	4	4	4	4	2	10	4	4	8	4	4	4	6
8_x	0	0	0	10	0	4	4	6	0	6	6	6	6	0	8	8
9_x	10	2	0	2	10	4	6	2	0	6	0	4	6	2	4	6
A_x	0	10	6	0	14	6	4	0	4	6	6	0	4	0	2	2
B_x	2	6	2	10	2	2	4	0	4	2	6	0	2	8	14	0
C_x	0	0	0	8	0	12	12	4	0	8	0	4	2	10	2	2
D_x	8	2	8	0	0	4	2	0	2	8	14	2	6	2	4	2
E_x	0	4	4	2	4	2	4	4	10	4	4	4	4	4	2	8
F_x	4	6	4	6	2	2	4	8	6	2	6	2	0	6	2	4
10_x	0	0	0	4	0	12	4	8	0	4	2	6	2	14	0	8
11_x	8	2	2	6	4	0	2	0	8	4	12	2	10	0	2	2
12_x	0	2	8	2	4	8	0	8	8	0	2	2	4	2	14	0
13_x	4	4	12	0	2	2	2	10	2	2	2	2	4	4	4	8
14_x	0	6	4	4	6	4	6	2	8	6	6	2	2	0	0	8
15_x	4	8	2	8	2	4	8	0	4	2	2	2	2	6	8	2
16_x	0	6	10	2	8	4	2	0	2	2	2	8	4	6	4	4
17_x	0	6	6	0	6	2	4	4	6	2	2	10	6	8	2	0
18_x	0	8	4	6	6	0	6	2	4	0	4	2	10	0	6	6
19_x	4	2	4	8	4	2	10	2	2	2	6	8	2	6	0	2
$1A_x$	0	8	6	4	4	0	6	4	4	8	0	10	2	2	2	4
$1B_x$	4	10	2	0	2	4	2	4	8	2	2	8	4	2	8	2
$1C_x$	0	6	8	8	4	2	8	0	12	0	10	0	4	0	2	0
$1D_x$	0	2	0	6	2	8	4	6	2	0	4	2	4	10	0	14
$1E_x$	0	4	8	2	4	6	0	4	10	0	2	6	4	8	4	2
$1F_x$	0	6	8	0	10	6	4	6	4	2	2	10	4	0	0	2
20_x	0	0	0	0	0	4	4	8	0	2	2	4	10	16	12	2
21_x	10	8	8	0	8	4	2	4	0	6	6	6	0	0	2	0
22_x	12	6	4	4	2	4	10	2	0	4	4	2	4	4	0	2
23_x	2	2	0	6	0	2	4	0	4	12	4	2	6	4	8	8
24_x	4	8	2	12	6	4	2	10	2	2	2	4	2	0	4	0
25_x	6	0	2	0	8	2	0	2	8	8	2	2	4	4	10	6
26_x	6	2	0	4	4	0	4	0	4	2	14	0	8	10	0	6
27_x	0	2	4	16	8	6	6	6	0	2	4	4	0	2	2	2
28_x	6	2	10	0	6	4	0	4	4	2	4	8	2	2	8	2
29_x	0	2	8	4	0	4	0	6	4	10	4	8	4	4	4	2
$2A_x$	2	6	0	4	2	4	4	6	4	8	4	4	4	2	4	6
$2B_x$	10	2	6	6	4	4	8	0	4	2	2	0	2	4	4	6
$2C_x$	10	4	6	2	4	2	2	2	4	10	4	4	0	2	6	2
$2D_x$	4	2	4	4	4	2	4	16	2	0	0	4	4	2	6	6
$2E_x$	4	0	2	10	0	6	10	4	2	6	6	2	2	0	2	8
$2F_x$	8	2	0	0	4	4	4	2	6	4	6	2	4	8	4	6
30_x	0	10	8	6	2	0	4	2	10	4	4	6	2	0	6	0
31_x	2	6	2	0	4	2	8	8	2	2	2	0	2	12	6	6
32_x	2	0	4	8	2	8	4	4	8	4	2	8	6	2	0	2
33_x	4	4	6	8	6	6	0	2	2	2	6	4	12	0	0	2
34_x	0	6	2	2	16	2	2	2	12	2	4	0	4	2	0	8
35_x	4	6	0	10	8	0	2	2	6	0	0	6	2	10	2	6
36_x	4	4	4	4	0	6	6	4	4	4	4	4	0	6	2	8
37_x	4	8	2	4	2	2	6	0	2	4	8	4	10	0	6	2
38_x	0	8	12	0	2	2	6	6	2	10	2	2	0	8	0	4
39_x	2	6	4	0	6	4	6	4	8	0	4	4	2	4	8	2
$3A_x$	6	0	2	2	4	6	4	4	4	2	2	6	12	2	6	2
$3B_x$	2	2	6	0	0	10	4	8	4	2	4	8	4	4	0	6
$3C_x$	0	2	4	2	12	2	0	6	2	0	2	8	4	6	4	10
$3D_x$	4	6	8	6	2	2	2	2	10	2	6	6	2	4	2	0
$3E_x$	8	6	4	4	2	10	2	0	2	2	4	2	4	2	10	2
$3F_x$	2	6	4	0	0	10	8	2	2	8	6	4	6	2	0	4

Table 29. The pairs XOR distribution table of S3.

Input	Output XOR															
XOR	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	A_x	B_x	C_x	D_x	E_x	F_x
0_x	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_x	0	0	0	0	0	16	16	0	0	16	16	0	0	0	0	0
2_x	0	0	0	8	0	4	4	8	0	4	4	8	8	8	8	0
3_x	8	6	2	0	2	4	8	2	6	0	4	6	0	6	2	8
4_x	0	0	0	8	0	0	12	4	0	12	0	4	8	4	4	8
5_x	4	2	2	8	2	12	0	2	2	0	12	2	8	2	2	4
6_x	0	8	8	4	8	8	0	0	8	0	8	0	4	0	0	8
7_x	4	2	6	4	6	0	16	6	2	0	0	2	4	2	6	4
8_x	0	0	0	4	0	8	4	8	0	4	8	8	4	8	8	0
9_x	8	4	4	4	4	0	8	4	4	0	0	4	4	4	4	8
A_x	0	6	6	0	6	4	4	6	6	4	4	6	0	6	6	0
B_x	0	12	0	8	0	0	0	0	12	0	0	12	8	12	0	0
C_x	0	0	0	4	0	8	4	8	0	4	8	8	4	8	8	0
D_x	8	4	4	4	4	0	0	4	4	8	0	4	4	4	4	8
E_x	0	6	6	4	6	0	4	6	6	4	0	6	4	6	6	0
F_x	0	6	6	4	6	4	0	6	6	0	4	6	4	6	6	0
10_x	0	0	0	0	0	8	12	4	0	12	8	4	0	4	4	8
11_x	4	2	2	16	2	4	0	2	2	0	4	2	16	2	2	4
12_x	0	0	0	8	0	4	4	8	0	4	4	8	8	8	8	0
13_x	8	2	6	0	6	4	0	6	2	8	4	2	0	2	6	8
14_x	0	8	8	0	8	0	8	0	8	8	0	0	0	0	0	16
15_x	8	4	4	0	4	8	0	4	4	0	8	4	0	4	4	8
16_x	0	8	8	4	8	8	0	0	8	0	8	0	4	0	0	8
17_x	4	6	2	4	2	0	0	2	6	16	0	6	4	6	2	4
18_x	0	8	8	8	8	4	0	0	8	0	4	0	8	0	0	8
19_x	4	4	4	0	4	4	16	4	4	0	4	4	0	4	4	4
$1A_x$	0	6	6	4	6	0	4	6	6	4	0	6	4	6	6	0
$1B_x$	0	6	6	4	6	4	0	6	6	0	4	6	4	6	6	0
$1C_x$	0	8	8	8	8	4	0	0	8	0	4	0	8	0	0	8
$1D_x$	4	4	4	0	4	4	0	4	4	16	4	4	0	4	4	4
$1E_x$	0	6	6	0	6	4	4	6	6	4	4	6	0	6	6	0
$1F_x$	0	0	12	8	12	0	0	12	0	0	0	0	8	0	12	0
20_x	0	0	0	8	0	0	0	12	0	0	0	12	8	12	12	0
21_x	0	4	8	0	8	4	8	8	4	0	4	4	0	4	8	0
22_x	8	2	2	0	2	4	8	6	2	8	4	6	0	6	6	0
23_x	4	6	2	8	2	4	0	2	6	0	4	6	8	6	2	4
24_x	0	6	6	4	6	4	0	6	6	0	4	6	4	6	6	0
25_x	0	8	4	4	4	0	0	4	8	8	0	8	4	8	4	0
26_x	0	6	6	0	6	4	8	2	6	8	4	2	0	2	2	8
27_x	4	6	2	8	2	4	0	2	6	0	4	6	8	6	2	4
28_x	16	4	4	0	4	4	4	4	4	4	4	4	0	4	4	0
29_x	0	6	2	8	2	4	0	2	6	8	4	6	8	6	2	0
$2A_x$	0	2	2	16	2	4	4	2	2	4	4	2	16	2	2	0
$2B_x$	8	0	4	0	4	8	16	4	0	0	8	0	0	0	4	8
$2C_x$	8	4	4	4	4	0	8	4	4	8	0	4	4	4	4	0
$2D_x$	4	2	6	4	6	8	0	6	2	0	8	2	4	2	6	4
$2E_x$	16	0	0	0	0	16	0	0	0	0	16	0	0	0	0	16
$2F_x$	16	0	0	0	0	0	16	0	0	16	0	0	0	0	0	16
30_x	0	6	6	4	6	4	0	6	6	0	4	6	4	6	6	0
31_x	0	8	4	4	4	0	0	4	8	8	0	8	4	8	4	0
32_x	16	6	6	4	6	0	4	2	6	4	0	2	4	2	2	0
33_x	0	2	6	4	6	8	8	6	2	0	8	2	4	2	6	0
34_x	0	12	12	8	12	0	0	0	12	0	0	0	8	0	0	0
35_x	0	4	8	0	8	4	8	8	4	0	4	4	0	4	8	0
36_x	0	2	2	4	2	0	4	6	2	4	0	6	4	6	6	16
37_x	0	2	6	4	6	8	8	6	2	0	8	2	4	2	6	0
38_x	0	4	4	0	4	4	4	4	4	4	4	4	0	4	4	16
39_x	0	6	2	8	2	4	0	2	6	8	4	6	8	6	2	0
$3A_x$	0	4	4	0	4	8	8	4	4	8	8	4	0	4	4	0
$3B_x$	16	4	4	0	4	0	0	4	4	0	0	4	0	4	4	16
$3C_x$	0	4	4	4	4	0	8	4	4	8	0	4	4	4	4	8
$3D_x$	4	2	6	4	6	8	0	6	2	0	8	2	4	2	6	4
$3E_x$	0	2	2	8	2	12	4	2	2	4	12	2	8	2	2	0
$3F_x$	8	4	0	8	0	0	0	0	4	16	0	4	8	4	0	8

Table 30. The pairs XOR distribution table of S4.

Input XOR	Output XOR															
	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	A_x	B_x	C_x	D_x	E_x	F_x
0_x	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_x	0	0	0	4	0	10	8	6	0	4	2	2	12	10	2	4
2_x	0	0	0	4	0	10	6	4	0	6	4	2	4	8	10	6
3_x	8	2	4	6	4	4	2	2	6	8	6	4	4	0	2	2
4_x	0	0	0	8	0	4	10	6	0	6	6	4	8	6	0	6
5_x	12	2	0	4	0	4	8	2	4	0	16	2	0	2	0	8
6_x	0	8	4	6	4	6	2	2	4	4	6	0	6	0	2	10
7_x	2	0	4	8	4	2	6	6	2	8	6	2	2	0	6	6
8_x	0	0	0	2	0	8	10	4	0	4	10	4	8	4	4	6
9_x	8	6	0	4	0	6	6	2	2	10	2	8	6	2	0	2
A_x	0	6	8	6	0	8	0	0	8	10	4	2	8	0	0	4
B_x	4	2	2	4	8	10	6	4	2	6	2	2	6	2	2	2
C_x	0	0	0	10	0	2	10	2	0	6	10	6	6	6	2	4
D_x	10	4	2	2	0	6	16	0	0	2	10	2	2	4	0	4
E_x	0	6	4	8	4	6	10	2	4	4	4	2	4	0	2	4
F_x	4	4	0	8	0	2	0	2	8	2	4	2	8	4	4	12
10_x	0	0	0	0	0	4	4	12	0	2	8	10	4	6	12	2
11_x	6	6	10	10	4	0	2	6	2	4	0	6	2	4	2	0
12_x	0	2	4	2	10	4	0	10	8	6	0	6	0	6	6	0
13_x	0	0	6	2	8	0	0	4	4	6	2	8	2	8	10	4
14_x	0	12	2	6	4	0	4	4	8	4	4	4	6	2	4	0
15_x	4	8	0	2	8	0	2	4	2	2	4	2	4	8	8	6
16_x	0	6	10	2	14	0	2	2	4	4	0	6	0	4	6	4
17_x	0	6	8	4	8	4	0	2	8	4	0	2	2	8	6	2
18_x	0	10	8	0	6	4	0	4	4	4	6	4	4	4	0	6
19_x	0	4	6	2	4	4	2	6	4	2	2	4	12	2	10	0
$1A_x$	0	2	16	2	12	2	0	6	4	0	0	4	0	4	4	8
$1B_x$	2	8	12	0	0	2	2	6	8	4	0	6	0	0	8	6
$1C_x$	0	10	2	6	6	6	6	4	8	2	0	4	4	4	2	0
$1D_x$	4	6	2	0	8	2	4	6	6	0	8	6	2	4	2	4
$1E_x$	0	2	6	2	4	0	0	2	12	2	2	6	2	10	10	4
$1F_x$	0	6	8	4	8	8	0	6	6	2	0	6	0	6	2	2
20_x	0	0	0	8	0	8	2	6	0	4	4	4	6	6	8	8
21_x	0	0	0	6	6	2	6	4	6	10	14	4	0	0	4	2
22_x	14	4	0	10	0	2	12	2	2	2	10	2	0	0	2	2
23_x	2	0	0	4	2	2	10	4	0	8	8	2	6	8	0	8
24_x	6	2	8	4	4	4	6	2	2	6	6	2	6	2	2	2
25_x	6	0	0	8	2	8	2	6	6	4	2	2	4	2	6	6
26_x	12	0	0	4	0	4	4	4	0	8	4	0	12	8	0	4
27_x	12	2	0	2	0	12	2	2	4	4	8	4	8	2	2	0
28_x	2	8	4	6	2	4	6	0	6	6	4	0	2	2	2	10
29_x	6	4	6	8	8	4	6	2	0	0	2	2	10	0	2	4
$2A_x$	4	4	0	2	2	4	6	2	0	0	6	4	10	4	4	12
$2B_x$	4	6	2	6	0	0	12	2	0	4	12	2	6	4	0	4
$2C_x$	8	6	2	6	4	8	6	0	4	4	0	2	6	0	6	2
$2D_x$	4	4	0	4	0	6	4	2	4	12	0	4	4	6	4	6
$2E_x$	6	0	2	4	0	6	6	4	2	10	6	10	6	2	0	0
$2F_x$	10	4	0	2	2	6	10	2	0	2	2	4	6	2	2	10
30_x	0	4	8	4	6	4	0	6	10	4	2	4	2	6	4	0
31_x	0	6	6	4	10	2	0	0	4	4	0	0	4	6	12	6
32_x	4	6	0	2	6	4	6	0	6	0	4	6	4	10	6	0
33_x	8	10	0	14	8	0	0	8	2	0	2	4	0	4	4	0
34_x	0	4	4	2	14	4	0	8	6	8	2	2	0	4	6	0
35_x	0	4	16	0	8	4	0	4	4	4	0	8	0	4	4	4
36_x	4	4	4	6	2	2	2	12	2	4	4	8	2	4	4	0
37_x	4	2	2	2	4	2	0	8	2	2	2	12	6	2	8	6
38_x	0	4	8	4	12	0	0	8	10	2	0	0	0	4	2	10
39_x	0	8	12	0	2	2	2	2	12	4	0	8	0	4	4	4
$3A_x$	0	14	4	0	4	6	0	0	6	2	10	8	0	0	4	6
$3B_x$	0	2	2	2	4	4	8	6	8	2	2	2	6	14	2	0
$3C_x$	0	0	10	2	6	0	0	2	6	2	2	10	2	4	10	8
$3D_x$	0	6	12	2	4	8	0	8	8	2	2	0	2	2	4	4
$3E_x$	4	4	10	0	2	4	8	8	2	2	0	2	6	8	4	0
$3F_x$	8	6	6	0	4	2	2	4	4	2	8	6	2	4	6	0

Table 31. The pairs XOR distribution table of S5.

Input XOR	Output XOR															
	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	A_x	B_x	C_x	D_x	E_x	F_x
0_x	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_x	0	0	0	6	0	2	6	2	0	4	2	4	6	16	14	2
2_x	0	0	0	2	0	10	6	10	0	2	4	8	6	6	8	2
3_x	0	8	0	8	0	6	4	6	4	4	4	12	2	4	2	0
4_x	0	0	0	8	0	0	8	0	0	6	8	10	2	4	10	8
5_x	10	2	4	4	4	8	8	4	2	2	0	4	0	8	0	4
6_x	0	8	4	4	8	4	2	2	12	0	2	6	6	2	2	2
7_x	6	6	4	0	2	10	2	2	2	2	6	6	8	0	6	2
8_x	0	0	0	6	0	2	16	4	0	2	6	2	4	12	6	4
9_x	10	4	2	6	0	2	6	2	4	0	8	6	4	4	2	4
A_x	0	14	4	4	0	2	2	2	10	4	4	4	6	4	2	2
B_x	4	6	2	0	2	2	12	8	2	2	2	6	8	2	0	6
C_x	0	0	0	12	0	10	4	6	0	8	4	4	2	12	2	0
D_x	12	0	2	10	6	4	4	2	4	2	6	0	2	6	0	4
E_x	0	6	4	0	4	4	10	8	6	2	4	6	2	0	6	2
F_x	2	2	2	2	6	2	6	2	10	4	8	2	6	4	4	2
10_x	0	0	0	8	0	8	0	12	0	4	2	6	8	4	6	6
11_x	6	2	6	4	6	2	6	4	6	6	4	2	4	0	6	0
12_x	0	8	4	2	0	4	2	0	4	10	6	2	8	6	4	4
13_x	6	6	12	0	12	2	0	6	6	2	0	4	0	2	4	2
14_x	0	4	6	2	8	6	0	2	6	10	4	0	2	4	6	4
15_x	2	2	6	6	4	4	2	6	2	6	8	4	4	0	4	4
16_x	0	4	14	6	8	4	2	6	2	0	2	0	4	2	0	10
17_x	2	6	8	0	0	2	0	2	2	6	0	8	8	2	12	6
18_x	0	4	6	6	8	4	2	2	6	4	6	4	2	4	2	4
19_x	2	6	0	2	4	4	4	6	4	8	6	4	2	2	6	4
$1A_x$	0	6	6	0	8	2	4	6	4	2	4	6	2	0	4	10
$1B_x$	0	4	10	2	4	4	2	6	6	6	2	2	6	6	2	2
$1C_x$	0	0	8	2	12	2	6	2	8	6	6	2	4	0	4	2
$1D_x$	2	4	0	6	8	6	0	2	6	8	6	0	2	4	0	10
$1E_x$	0	10	8	2	8	2	0	2	6	4	2	4	6	4	2	4
$1F_x$	0	6	6	8	6	4	2	4	4	2	2	0	2	4	2	12
20_x	0	0	0	0	0	6	6	4	0	4	8	8	4	6	10	8
21_x	2	8	6	8	4	4	6	6	8	4	0	4	0	2	2	0
22_x	16	2	4	6	2	4	2	0	6	4	8	2	0	2	2	4
23_x	0	4	0	4	4	6	10	4	2	2	6	2	4	6	6	4
24_x	10	8	0	6	12	6	10	4	8	0	0	0	0	0	0	0
25_x	0	2	4	2	0	4	4	0	4	0	10	10	4	10	6	4
26_x	2	2	0	12	2	2	6	2	4	4	8	0	6	6	8	0
27_x	8	4	0	8	2	4	2	4	0	6	2	4	4	8	2	6
28_x	6	8	4	6	0	4	2	2	4	8	2	6	4	2	2	4
29_x	2	4	4	0	8	8	6	8	6	4	0	4	4	4	2	0
$2A_x$	6	0	0	6	6	4	6	8	2	4	0	2	2	4	6	8
$2B_x$	12	0	4	0	0	4	2	2	2	6	10	6	10	2	4	0
$2C_x$	4	2	6	0	0	6	8	6	4	2	2	8	4	6	4	2
$2D_x$	6	2	2	6	6	4	4	2	6	2	4	8	4	2	4	2
$2E_x$	4	6	2	4	2	4	4	2	4	2	4	6	4	10	4	2
$2F_x$	10	0	4	8	0	6	6	2	0	4	4	2	6	2	2	8
30_x	0	12	8	2	0	6	0	0	6	6	0	2	8	2	6	6
31_x	2	6	10	4	2	2	2	4	6	0	2	6	0	2	4	12
32_x	4	2	2	8	10	8	8	6	0	2	2	4	4	2	2	0
33_x	4	2	2	2	6	0	4	0	10	6	6	4	0	4	8	6
34_x	0	4	4	2	6	4	0	4	6	2	6	4	2	8	0	12
35_x	6	12	4	2	4	2	2	4	8	2	2	0	6	4	4	2
36_x	0	2	2	4	4	4	4	0	2	10	12	4	0	10	4	2
37_x	10	2	2	6	14	2	2	6	2	0	4	6	2	0	4	2
38_x	0	4	14	0	8	2	0	4	4	4	2	0	8	2	4	8
39_x	2	4	8	0	6	2	0	6	2	6	4	2	8	6	2	6
$3A_x$	8	4	0	4	6	2	0	4	6	8	6	0	6	0	4	6
$3B_x$	0	4	6	6	2	2	2	14	0	12	0	4	2	2	8	0
$3C_x$	0	6	16	0	2	2	2	8	4	2	0	12	6	2	2	0
$3D_x$	0	6	2	2	2	6	8	2	4	2	6	2	6	2	4	10
$3E_x$	4	2	2	4	4	0	6	10	4	2	4	6	6	2	6	2
$3F_x$	0	4	6	6	4	8	4	0	4	8	4	0	4	8	2	2

Table 32. The pairs XOR distribution table of S6.

Input	Output XOR															
XOR	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	A_x	B_x	C_x	D_x	E_x	F_x
0_x	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_x	0	0	0	2	0	4	4	14	0	12	4	6	2	6	6	4
2_x	0	0	0	0	0	12	2	2	0	4	0	4	8	12	6	14
3_x	8	2	12	2	6	8	6	0	6	4	4	2	2	0	0	2
4_x	0	0	0	8	0	4	4	8	0	8	8	12	2	6	2	2
5_x	6	0	0	2	8	0	8	4	0	2	6	0	10	6	6	6
6_x	0	2	12	0	8	4	8	2	4	4	4	2	6	0	6	2
7_x	4	6	4	12	0	4	2	0	0	14	2	6	4	0	0	6
8_x	0	0	0	8	0	0	6	10	0	4	12	4	6	6	0	8
9_x	10	8	4	8	6	2	2	0	2	6	8	2	0	6	0	0
A_x	0	10	6	2	12	2	4	0	4	4	6	4	4	0	0	6
B_x	0	2	2	2	4	8	6	4	4	0	4	2	6	4	2	14
C_x	0	0	0	4	0	4	8	4	0	2	6	0	14	12	8	2
D_x	6	6	2	4	2	6	4	6	6	4	8	8	0	2	0	0
E_x	0	12	10	10	0	2	4	2	8	6	4	2	0	0	2	2
F_x	2	0	0	0	6	8	8	0	6	2	4	6	8	0	6	8
10_x	0	0	0	4	0	2	8	6	0	6	4	10	8	4	8	4
11_x	6	10	10	4	4	2	0	4	4	0	2	8	4	2	2	2
12_x	0	0	8	8	2	8	2	8	6	4	2	8	0	0	8	0
13_x	4	4	2	2	8	6	0	2	2	2	0	4	6	8	14	0
14_x	0	8	6	2	8	8	2	6	4	2	0	2	8	6	0	2
15_x	4	4	8	2	4	0	4	10	8	2	4	4	4	2	0	4
16_x	0	6	10	2	2	2	2	4	10	8	2	2	0	4	10	0
17_x	8	2	4	2	6	4	0	6	4	4	2	2	0	4	8	8
18_x	0	16	2	2	6	0	6	0	6	2	8	0	6	0	2	8
19_x	0	8	0	2	4	4	10	4	8	0	6	4	2	6	2	4
$1A_x$	0	2	4	8	12	4	0	6	4	4	0	2	0	6	4	8
$1B_x$	0	6	2	6	4	2	4	4	6	4	8	4	2	0	10	2
$1C_x$	0	8	4	4	2	6	6	6	6	4	6	8	0	2	0	2
$1D_x$	4	4	4	0	0	2	4	2	4	2	2	4	10	10	8	4
$1E_x$	0	0	2	2	12	6	2	0	12	2	2	4	2	6	8	4
$1F_x$	2	2	10	14	2	4	2	4	4	6	0	2	4	8	0	0
20_x	0	0	0	14	0	8	4	2	0	4	2	8	2	6	0	14
21_x	4	2	6	2	12	2	4	0	6	4	10	2	4	2	2	2
22_x	10	6	0	2	4	4	10	0	4	0	12	2	8	0	0	2
23_x	0	6	2	2	2	4	6	10	0	4	8	2	2	6	0	10
24_x	4	2	0	6	8	2	6	0	8	2	2	0	8	2	12	2
25_x	2	0	2	16	2	4	6	4	6	8	2	4	0	6	0	2
26_x	6	10	0	10	0	6	4	4	2	2	4	6	2	4	2	2
27_x	4	0	2	0	2	2	14	0	4	6	6	2	12	2	4	4
28_x	14	4	6	4	4	6	2	0	6	6	2	2	4	0	2	2
29_x	2	2	0	2	0	8	4	2	4	6	4	4	6	4	12	4
$2A_x$	2	4	0	0	0	2	8	12	0	8	2	4	8	4	4	6
$2B_x$	16	6	2	4	6	10	2	2	2	2	2	2	4	2	2	0
$2C_x$	2	6	6	8	2	2	0	6	0	8	4	2	2	6	8	2
$2D_x$	6	2	4	2	8	8	2	8	2	4	4	0	2	0	8	4
$2E_x$	2	4	8	0	2	2	2	4	0	2	8	4	14	6	0	6
$2F_x$	2	2	2	8	0	2	2	6	4	6	8	8	6	2	0	6
30_x	0	6	8	2	8	4	4	0	10	4	4	6	0	0	2	6
31_x	0	8	4	0	6	2	2	6	6	0	0	2	6	4	8	10
32_x	2	4	0	0	6	4	10	6	6	4	6	2	4	6	2	2
33_x	0	16	6	8	2	0	2	2	4	2	8	4	0	4	6	0
34_x	0	4	14	8	2	2	2	4	16	2	2	2	0	2	0	4
35_x	0	6	0	0	10	8	2	2	6	0	0	8	6	4	4	8
36_x	2	0	2	2	4	6	4	4	2	2	4	2	4	16	10	0
37_x	6	6	6	8	4	2	4	4	4	0	6	8	2	4	0	0
38_x	0	2	2	2	8	8	0	2	2	2	0	6	6	4	10	10
39_x	4	4	16	8	0	6	4	2	4	4	2	6	0	2	2	0
$3A_x$	16	6	4	0	2	0	2	6	0	4	8	10	0	0	4	2
$3B_x$	2	0	0	2	0	4	4	4	2	6	2	6	6	12	12	2
$3C_x$	0	0	8	0	12	8	2	6	6	4	0	2	2	4	6	4
$3D_x$	2	4	12	2	2	2	0	4	6	10	2	6	4	2	0	6
$3E_x$	4	6	6	6	2	0	4	8	2	10	4	6	0	4	2	0
$3F_x$	14	0	0	0	8	0	6	8	4	2	0	0	4	8	4	6

Table 33. The pairs XOR distribution table of S7.

Input XOR	Output XOR															
	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	A_x	B_x	C_x	D_x	E_x	F_x
0_x	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_x	0	0	0	6	0	16	10	0	0	0	6	0	14	6	2	4
2_x	0	0	0	8	0	10	4	2	0	10	2	4	8	8	6	2
3_x	6	0	2	8	2	6	4	0	6	6	6	2	2	0	8	6
4_x	0	0	0	2	0	4	6	12	0	6	8	4	10	4	8	0
5_x	4	10	6	0	0	2	6	0	4	10	4	6	8	2	0	2
6_x	0	0	10	4	6	4	4	8	2	6	4	2	4	2	2	6
7_x	6	2	8	2	8	10	6	6	4	2	0	4	0	0	0	6
8_x	0	0	0	4	0	6	4	2	0	8	6	10	8	2	2	12
9_x	8	4	0	6	0	4	4	6	2	4	6	2	12	2	0	4
A_x	0	0	16	4	6	6	4	0	4	6	4	2	2	0	0	10
B_x	2	8	0	6	2	6	0	4	4	10	0	2	10	2	6	2
C_x	0	0	0	2	0	10	10	6	0	6	6	6	2	6	10	0
D_x	6	0	4	10	2	0	8	6	2	2	6	10	2	2	2	2
E_x	0	0	6	8	4	8	0	2	10	6	2	4	6	2	4	2
F_x	8	0	4	2	2	4	2	2	2	6	4	6	0	2	14	6
10_x	0	0	0	4	0	0	8	12	0	0	8	8	2	10	6	6
11_x	0	6	4	6	2	2	6	6	4	6	4	6	0	4	4	4
12_x	0	4	0	8	6	2	8	4	2	4	4	6	2	4	10	0
13_x	4	2	2	6	8	6	2	2	14	2	2	4	2	2	2	4
14_x	0	16	4	2	6	0	2	6	4	0	4	6	4	6	4	0
15_x	0	10	6	0	6	0	2	8	2	2	0	8	2	6	6	6
16_x	0	12	6	4	6	0	0	0	8	6	6	2	2	6	4	2
17_x	0	6	8	0	6	2	4	6	6	0	2	6	4	4	2	8
18_x	0	12	2	2	8	0	8	0	10	4	4	2	4	2	0	6
19_x	6	4	8	0	8	0	4	2	0	0	12	2	4	6	2	6
$1A_x$	0	4	6	2	8	8	0	4	8	0	0	0	6	2	0	16
$1B_x$	2	4	8	10	2	4	2	8	2	4	8	2	0	2	4	2
$1C_x$	0	12	6	4	6	4	2	2	6	0	4	4	2	10	2	0
$1D_x$	8	6	0	0	10	0	0	8	10	4	2	2	2	8	4	0
$1E_x$	0	4	8	6	8	2	4	4	10	2	2	4	2	0	6	2
$1F_x$	4	2	4	2	6	2	4	0	2	6	2	2	2	16	8	2
20_x	0	0	0	16	0	4	0	0	0	14	6	4	2	0	4	14
21_x	0	0	2	10	2	8	10	0	0	6	6	0	10	2	2	6
22_x	8	0	6	0	6	4	10	2	0	6	8	0	4	4	2	4
23_x	4	8	0	6	0	4	8	6	2	2	10	4	8	0	0	2
24_x	4	0	4	8	4	6	2	4	8	6	2	0	0	4	4	8
25_x	0	4	6	8	2	8	8	0	4	2	4	4	2	2	6	4
26_x	2	6	0	6	4	4	4	6	6	0	4	4	10	4	2	2
27_x	6	6	0	0	2	2	6	2	4	4	6	10	2	6	2	6
28_x	10	2	6	2	4	12	12	0	2	2	4	0	0	0	2	6
29_x	4	0	0	14	2	10	4	2	8	6	4	0	4	2	2	2
$2A_x$	8	8	0	2	0	2	4	0	2	6	8	14	2	8	0	0
$2B_x$	2	2	0	0	4	2	10	4	6	2	4	0	6	4	8	10
$2C_x$	2	6	6	2	4	6	2	0	2	6	4	0	6	4	10	4
$2D_x$	8	0	4	4	6	2	0	0	6	8	2	4	6	4	4	6
$2E_x$	6	2	2	4	2	2	6	12	4	0	4	2	8	8	0	2
$2F_x$	8	12	4	6	6	4	2	2	2	2	4	2	2	4	0	4
30_x	0	4	6	2	10	2	2	2	4	8	0	0	8	4	6	6
31_x	4	6	8	0	4	6	0	4	4	6	10	2	2	4	4	0
32_x	6	6	6	2	4	6	0	2	0	6	8	2	2	6	6	2
33_x	6	6	4	2	4	0	0	10	2	2	0	6	8	4	0	10
34_x	0	2	12	4	10	4	0	4	12	0	2	4	2	2	2	4
35_x	6	4	4	0	10	0	0	4	10	0	0	4	2	8	8	4
36_x	4	6	2	2	2	2	6	8	6	4	2	6	0	4	10	0
37_x	2	2	8	2	4	4	4	2	6	2	0	10	6	10	2	0
38_x	0	4	8	4	2	6	6	2	4	2	2	4	6	4	4	6
39_x	4	4	4	8	0	6	0	6	4	8	2	2	2	4	8	2
$3A_x$	8	8	0	4	2	0	10	4	0	0	0	4	8	6	8	2
$3B_x$	8	2	6	4	4	4	4	0	6	4	4	6	4	4	4	0
$3C_x$	0	6	6	6	6	0	0	8	8	2	4	8	4	2	4	0
$3D_x$	2	2	8	0	10	0	2	12	0	4	0	8	0	2	6	8
$3E_x$	6	4	0	0	4	4	0	10	6	2	6	12	2	4	0	4
$3F_x$	0	6	6	0	4	4	6	10	0	6	8	2	0	4	8	0

Table 34. The pairs XOR distribution table of S8.

References

[1] E. F. Brickell, J. H. Moore, M. R. Purtill, *Structure in the S-Boxes of the DES*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of CRYPTO'86, pp. 3–7, 1986.

[2] David Chaum, Jan-Hendrik Evertse, *Cryptanalysis of DES with a reduced number of rounds, Sequences of linear factors in block ciphers*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of CRYPTO'85, pp. 192–211, 1985.

[3] D. W. Davies, *Investigation of a Potential Weakness in the DES Algorithm*, 1987, private communication.

[4] Bert Den-Boer, *Cryptanalysis of F.E.A.L.*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of EUROCRYPT'88, pp. 293–300, 1988.

[5] Yvo Desmedt, Jean-Jacque Quisquater, Marc Davio, *Dependence of Output on Input in DES: Small Avalanche Characteristics*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of CRYPTO'84, pp. 359–376, 1984.

[6] W. Diffie, M. E. Hellman, *Exhaustive Cryptanalysis of the NBS Data Encryption Standard*, Computer, Vol. 10, No. 6, pp. 74–84, June 1977.

[7] H. Feistel, *Cryptography and Data Security*, Scientific American, Vol. 228, No. 5, pp. 15–23, May 1973.

[8] M. E. Hellman, *A Cryptanalytic Time-Memory Tradeoff*, IEEE Trans. Inform. Theory, Vol. 26, No. 4, pp. 401–406, July 1980.

[9] M. E. Hellman, R. Merkle, R. Schroppel, L. Washington, W. Diffie, S. Pohlig and P. Schweitzer, *Results of an Initial Attempt to Cryptanalyze the NBS Data Encryption Standard*, Technical Report, SEL 76-042, Stanford university, September 1976.

[10] Ralph C. Merkle, *A Fast Software One-Way Hash Function*, Journal of Cryptology, Vol. 3, No. 1, pp. 43–58, 1990.

[11] Shoji Miyaguchi, *FEAL-N specifications*, technical note, NTT, 1989.

[12] Shoji Miyaguchi, *The FEAL cipher family*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of CRYPTO'90, pp. 627–638, 1990.

[13] S. Miyaguchi, K. Ohta, M. Iwata, *128-bit hash function (N-Hash)*, proceedings of SECURICOM'90, pp. 123–137, March 1990.

[14] Shoji Miyaguchi, Akira Shiraishi, Akihiro Shimizu, *Fast Data Encryption Algorithm FEAL-8*, Review of electrical communications laboratories, Vol. 36, No. 4, pp. 433–437, 1988.

[15] National Bureau of Standards, *Data Encryption Standard*, U.S. Department of Commerce, FIPS pub. 46, January 1977.

[16] Ingrid Schaumuller-Bichl, *Zur Analyse des Data Encryption Standard und Synthese Verwandter Chiffriersysteme*, Ph.D. Thesis, Linz university, May 1981.

[17] Ingrid Schaumuller-Bichl, *Cryptanalysis of the Data Encryption Standard by the Method of Formal Coding*, Lecture Notes in Computer Science, Cryptography, proceedings of the Workshop on Cryptography, Burg Feuerstein, Germany, March 29–April 2, pp. 235–255, 1982.

[18] Ingrid Schaumuller-Bichl, *On the Design and Analysis of New Cipher Systems Related to the DES*, technical report, Linz university, 1983.

[19] Akihiro Shimizu, Shoji Miyaguchi, *Fast Data Encryption Algorithm FEAL*, Lecture Notes in Computer Science, Advances in Cryptology, proceedings of EUROCRYPT'87, pp. 267–278, 1987.

[20] Akihiro Shimizu, Shoji Miyaguchi, *Fast Data Encryption Algorithm FEAL*, Abstracts of EUROCRYPT'87, pp. VII-11–VII-14, April 1987.

Contents

List of Tables

List of Figures