Crypto-Archaeology: Unearthing Design Methodology of DES S-Boxes

Sankhanil Dey [1], Ranjan Ghosh [2*]
[1,2] Institute of Radio Physics and Electronics, University of Calcutta, 92 APC Road, Kolkata, India
[*]sdrpe_rs@caluniv.ac.in

Abstract: US defence sponsored the DES program in 1974 and released it in 1977. It remained as a well-known and well-accepted block cipher until 1998. Thirty-two 4-bit DES S-Boxes are grouped in eight each with four and are put in public domain without any mention of their design methodology. S-Boxes, 4-bit, 8-bit or 32-bit, find a permanent seat in all future block ciphers. In this paper, while looking into the design methodology of DES S-Boxes, we find that S-Boxes have 128 balanced and non-linear Boolean Functions, of which 102 used once, while 13 used twice and 92 of 102 satisfy the Boolean-Function-level Strict Avalanche Criterion. All the S-Boxes satisfy the Bit Independence Criterion. Their Differential Cryptanalysis exhibits better results than the Linear Cryptanalysis. However, no S-Boxes satisfy the S-Box-level SAC analyses. It seems that the designer emphasized satisfaction of Boolean-Function-level SAC and S-Box-level BIC and DC, not the S-Box-level LC and SAC.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3285v1 | CC BY 4.0 Open Access | rec: 26 Sep 2017, publ: 26 Sep 2017

Crypto-Archaeology: Unearthing Design Methodology of DES S-Boxes

Sankhanil Dey[1] and Ranjan Ghosh[2],
Institute of Radio Physics and Electronics[1,2],
University of Calcutta.

Abstract. US defense sponsored the DES program in 1974 and released it in 1977. It remained as a well-known and well-accepted block cipher until 1998. Thirty-two 4-bit DES S-Boxes are grouped in eight each with four and are put in public domain without any mention of their design methodology. S-Boxes, 4-bit, 8-bit or 32-bit, find a permanent seat in all future block ciphers. In this paper, while looking into the design methodology of DES S-Boxes, we find that 32 S-Boxes have 128 balanced and non-linear Boolean Functions, of which 102 used once, while 13 used twice and 92 of 102 satisfy the Boolean-Function-level Strict Avalanche Criterion. All the S-Boxes satisfy the Bit Independence Criterion. Their Differential Cryptanalysis exhibits better results than the Linear Cryptanalysis. However, no S-Boxes satisfy the S-Box-level SAC analyses. It seems that the designer emphasized satisfaction of Boolean-Function-level SAC and S-Box-level BIC and DC, not the S-Box-level LC and SAC.

Keywords: Block Cipher, Boolean Function, Data Encryption Standard, DES S-Boxes, Design Methodology.

1 Introduction.

Through the initiative of US defense since 1974 [1], the Data Encryption Standard (DES), a cryptographic block ciphering module, was developed and released in 1977. Feistel and Coppersmith, who designed the earlier Lucifer block cipher [2], took a leading role in DES design [3] and a team of researchers at IBM participated in its development. In early seventies, Feistel first conceived the idea of a modern computer-based block cipher [4], [5] and developed Lucifer with many variants in terms of key-text pairs as (48-bit each), (64-bit and 32-bit) and (128-bit each) [6] the first one was used in electronic banking. Among all the variants, the 128-bit pair was considered best because of its large key size. During seventies and eighties, many elite researchers used to consider that DES is prone to brute-force attack [7], [8], [9] since it has only 56-bit key to encrypt 64-bit text. The DES was a permutation and substitution based bit-level encryption algorithm. All permutation and substitution tables used in DES were put in public domain along with the algorithm, but with no mention of their design methodology. This created a strong feeling among researchers [10], [11] that by suppressing the design rule the government desires to gain an edge in eavesdropping to private messages and the feeling became so pervasive that with time a consortium took shape to break DES. In 1990, Biham and Shamir [12] reported

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3285v1 | CC BY 4.0 Open Access | rec: 26 Sep 2017, publ: 26 Sep 2017

breaking DES following a differential cryptanalysis and in 1993, Matsui [13] reported the same by linear cryptanalysis. In 1998, the Electronic Frontier Foundation, following a call from RSA Data Security [14], spent $2.5 million to build a dedicated Deep Crack” hardware based on VLSI technology and searched the DES encryption key within 56 hours [15]. Good reports on history of breaking DES appeared in [16], [17]. It is interesting to note that in 2006 an identical initiative based on FPGA technology achieved the same goal with much less cost [18].

The S-Boxes are the most important component in DES, since the substitution process exhibits non-linear features. This has led to a spur of research activities on S-Boxes, of 4-bit [19], [20], [21] of 8-bit [22] and of 32-bit [23] and also on many different DES-like ciphers [24], [25], [26]. It is interesting to note that after the breaking of DES by differential [13] and linear [14] cryptanalysis, Coppersmith in a paper [27] mentioned eight criteria that were considered while designing the DES S-Boxes and highlighted the fact that these were reasonably resistant to differential cryptanalysis, but did not mention the design methodology of the S-Boxes. Schneier [28] extensively reviews coppersmith’s submission.

2. Design Methodology Adopted for DES S-Boxes.

A 4-bit S-box has 16 integer values between 0 and 15 randomly arranged in 16 columns of one row and can be represented by four 1-bit Boolean functions each of 16 bits corresponding to 16 possibilities of 4 input bits. One thus requires 128 Boolean functions to represent 32 S-Boxes. If the column element values become identical to each of its column index values, the S–Box becomes an identity S-Box. Traveres [21] first proposed defining S-Boxes using four Boolean functions. Using a route based on irreducible polynomials from among 4 such 4-bit polynomials, one can also form an S-Box. It seems, Coppersmith took the Boolean function route and formed 32 S-Boxes using 128 Boolean functions. In the present paper, we concentrate on the Boolean function approach and in the following Secs. 2.1 through 2.7, computational procedures regarding their formation, Strict Avalanche and Bit Independence Criteria are discussed following differential and linear cryptanalysis of S-Boxes.

2.1. Algebraic Normal Form (ANF) of a Boolean Function (BF)

A Boolean Function (BF) with 4-bit input providing 1-bit output is a mapping from $(0,1)^4$ to (0, 1). Possible input combinations for 4 bits are $2^4 = 16$, and this gives 16 coefficients of ANF, bf(x) as,

$$bf\ (x) = a_c + (a_0x_0 + a_1x_1 + a_2\ x_2 + a_3x_3) + (a_4\ x_0\ x_1 + a_5\ x_0\ x_2 + a_6\ x_0\ x_3 + a_7\ x_1\ x_2 + a_8\ x_1\ x_3 + a_9\ x_2\ x_3) + (a_{10}\ x_0\ x_1\ x_2 + a_{11}\ x_0\ x_1\ x_3 + a_{12}\ x_0\ x_2\ x_3 + a_{13}\ x_1\ x_2\ x_3) + a_{14}\ x_0\ x_1\ x_2\ x_3.$$

where, x represents 4-bit input $\{x_3x_2x_1x_0\}$, bf assumes 1-bit output and ‘+’ represents XOR operation. Here a_c is a constant coefficient, a_0, a_1, a_2 and a_3 are four 1-bit linear coefficients, a_4, $a_{5,}$ a_6, a_7, a_8 and a_9 are six 2-bit non-linear coefficients, a_{10}, a_{11}, a $_{12}$ and a_{13} are 3-bit non-linear coefficients and a_{14} is a 4-bit non-linear coefficient. A 4-bit truth table, providing one 16-bit output vector for four 16-bit input vectors, each one which is attached to x_3, x_2, x_1 and x_0, gives 1-bit binary value for each of the 16 coefficients which can be termed as the ANF representation of the BF. For a BF, ANF representation is an alternative to the truth table representation. Of the 65536 (= 2^{16}) 4-bit BFs, 32 (= 2^5) are linear and the rest 65504 are non-linear. For linear BFs, the coefficients of product terms in ANF are zero. A balanced BF has equal number of 0s and 1s in its output, i. e. its hamming weight is $2^{(4-1)} = 2^3 = 8$. Hence out of the 65504 ($=2^{16} - 2^5$) non-linear BFs, 12870 (= ${}^{16}C_8$) are balanced. The issue is to find out how many among 12870 are cryptographically relevant. However, in the present paper we are interested in the 128 BFs and in their ANFs used in the 32 DES S-Boxes.

2.2. 4-BFs and 4-ANFs for a 4-bit S-Box: 128 BFs and 128 ANFs for 32 S-Boxes

A 4-bit S-Box has 16 column elements in one row; the input of the 16 column index values of a row are sequential 4-bit pattern varying from {0000} to {1111} and the output are non-sequential 4-bit integer values in text in hex-format between 0 and f. The 16 sequential column index values of 4-bit are stored in 4bt16in[16][4]. The elemental values of 16 columns of all of the 32 S-Boxes are stored in hex-format in 32 rows in a text-file, the row text-data are read, transformed into integer ASCII data and stored in sb[32][16] – these are in turn transformed into 4 bits and are stored in 4bt16out[32][16][4]. All the DES S-Boxes are shown in hex-format in column 4 of Table 1 [32]. The 1st bit of the 4 bits of each element of 16 columns of an S-Box taking sequentially gives the 1st BF (bf_0); 2nd bit, the 2nd BF (bf_1); 3rd bit, the 3rd BF (bf_2) and 4th bit, the 4th BF (bf_3). The bit data of all the 32 S-Boxes stored in 4bt16out[32][16][4] are stored in 16bt4bf[32][4][16] following the formalism stated above and one thus gets 128 Boolean Functions. The sequential input bit data available in 4bt16in[16][4] are stored in 16bit4x[4][16] as four 16-bit x-vectors. The 128 BFs are shown in column 7 in Table 1[32] in a group of 4 for each S-Box. Putting 16 bits of one Boolean function corresponding to its sequential 4-bit inputs, one obtains the ANF coefficients. The pseudo code for the same is as follows,

```
for(s=0;s<32;s++) {
for(j=0;j<4;j++) {
for(i=0;i<16;i++){
switch(i) {
i=0 : a[i]= bf[s][j][i];
i=1 : a[i]= a[0]^bf[s][j][i];
i=2 : a[i]= a[0]^bf[s][j][i];
i=3 : a[i]= a[0]^a[1]^a[2]^bf[s][j][i];
i=4 : a[i]= a[0]^bf[s][j][i];
i=5 : a[i]= a[0]^a[1]^a[4]^bf[s][j][i];
i=6 : a[i]= a[0]^a[2]^a[4]^bf[s][j][i];
i=7 : a[i]= a[0]^a[1]^a[2]^a[3]^a[4]^a[5]^a[6]
                  ^bf[s][j][i];
i=8 : a[i]= a[0]^bf[s][j][i];
i=9 : a[i]= a[0]^a[1]^a[8]^bf[s][j][i];
i=10: a[i] = a[0]^a[2]^a[8]^bf[s][j][i];
i=11: a[i] = a[0]^a[1]^a[2]^a[8]^a[3]^a[9]^a[10]
                 ^bf[s][j][i];
i=12: a[i] = a[0]^a[4]^a[8]^bf[s][j][i];
i=13: a[i] = a[0]^a[1]^a[4]^a[8]^a[5]^a[9]^a[12]
                 ^bf[s][j][i];
i=14: a[i] = a[0]^a[2]^a[4]^a[8]^a[6]^a[12]^a[10]
                 ^bf[s][j][i];

i=15: a[i] = a[0]^a[1]^a[2]^a[8]^a[3]^a[9]^a[10]
                 ^a[4]^a[5]^a[6]^a[7]^
                 a[11]^a[12]^a[13] ^a[14]^bf[s][j][i];
                                 }
                        }
                }
      }
```

All 1-bit ANF coefficients are shown in column 9 of Table 1 [32]. The 16 ANF coefficients are shown in Table 1 as being split in five categories separated by hyphens, (1) 1st bit: constant coefficient {a_c} (2) next 4 bits: 1-bit four linear coefficients {a_0, a_1, a_2 and a_3} (3) next 6 bits: 2-bit six non-linear coefficients {a_4, a_5, a_6, a_7, a_8 and a_9} (4) next 4 bits: 3-bit four non-linear coefficients {a_{10}, a_{11}, a_{12} and a $_{13}$} and (5) last 1 bit: 4-bit one non-linear coefficients {a_{14}}.

2.3. Strict Avalanche Criterion (SAC) of Boolean Function

A 4-bit Boolean Function {bf} has four 16-bit input vectors{x_3,x_2,x_1,x_0}and one 16-bit output vector. It satisfies SAC, if the 50% of the 16-bit output vector changes if one of the four 16-bit input vectors is complemented. Here x_0 corresponds to LSB of all the possible sixteen 4-bit element values in an S-Box. Webster and Travares [21] first introduced the idea of SAC, by stating that every output bit should change with a probability ½, in the event one input variable of a Boolean function is complemented. The probability of changing each output bit being ½ indicates extreme uncertainty in predicting a particular output bit to change. This is the theoretical essence of SAC.

On inspecting the bit patterns of all the 16 bits of x_3, x_2, x_1, x_0, one notes that these are ‘00ff’, ‘0f0f’, ‘3333’ and ‘5555’ respectively in hex. The following algorithm in effect executes the SAC searching criterion of {bf} on complementing x_3 vectors,

$$\texttt{wt\{(bf\&00ff)\^{}(bf>>8\&00ff)\}+ wt\{(bf\&ff00)\^{}(bf>>8\&ff00)\}}\ \texttt{= N3=}\ \texttt{NL}_3 + \texttt{NU}_3\ .$$

Similarly, for identical purpose, one has to right shift ‘bf’ by 4 for x2, for x_1 by 2 and for x_0 by 1 and execute the above algorithm as,

wt{(bf&0f0f)^(bf>>4&0f0f)}+wt{(bf&f0f0)^(bf>>4)&f0f0)}
= N2 = NL_2 +NU_2 .
wt{(bf&3333)^(bf>>2&3333)}+wt{(bf&cccc)^(bf>>2&cccc)}
= N1 = NL_1 +NU_1 .
wt{(bf&5555)^(bf>>1&5555)}+wt{(bf& aaaa)^(bf>>1&aaaa)}
= N0 = $N1_0$ + NU_0 .

According to Traveres [19], [21], a {bf} satisfies SAC, if at least one of the four N-values becomes 8 with equal lower-half and upper-half N-values. The results of SAC tests for all the 128 BFs are shown in column 13 of Table 1 [32].

Kim, Matsumoto and Imai [29] proposed a different algorithm and stressed that all the four N-values should satisfy the SAC criterion, for a {bf} to satisfy SAC. Both the algorithms give identical N-values. The algorithm of Kim, Matsumoto and Imai is elaborated in Sec. 2.7.

2.4. 2 NL BFs and (output) Bit Independence Criterion (BIC): BIC for a S-Box

By (output) Bit Independence Criterion, it is to be ensured that the output bit i of one BF and output bit j of another BF act independent of each other, i.e. bit i and j are not co-related to each other [20]. If both the two BFs are linear, any two-output bits of the two linear BFs change with a correlation of ±1 indicating that xor of i and j should change with probability 1 or 0. i. e. either it remains same or flips. This indicates that the issue of nonlinearity of BF is closely related to BIC. One can look for the satisfaction of the BIC criterion between 2 NL BFs, if the SAC criterion is applied to the output BF = BF1 XOR BF2. The BIC criterion on BF1 and BF2 together is said to be satisfied, if the SAC criterion presented in Sec. 2.3 above on one BF, is satisfied within a range of 40% to 60% of output bits of BF, instead of just 50%. Algorithmically one can write the BIC criterion on BF as follows:

wt {(bf&00ff)^(bf>>8&00ff)}+wt{(bf&ff00)^(bf>>8&ff00)}
= N3= NL_3 + NU_3 .
wt {(bf&0f0f)^(bf>>4&0f0f)}+wt{(bf&f0f0)^(bf>>4)&f0f0)}
= N2 = NL_2 +NU_2 .
wt {(bf&3333)^(bf>>2&3333)}+wt{(bf&cccc)^(bf>>2&cccc)}
=N1= NL_1 +NU_1 .
wt {(bf&5555)^(bf>>1&5555)}+wt{(bf&aaaa)^(bf>>1&aaaa)}
= N0=$N1_0$ + NU_0 .

Here the BIC criterion on BF1 and BF2 is considered to be satisfied, if any one of the four N's lies between 6 and 10. For an S-Box having 4 non-linear BFs, one has to look for satisfaction of BIC criterion for 6 set of 2 BFs. The BIC results of all the 32 DES S-Boxes are shown in column 14 of Table 1 [32].

2.5. Differential Cryptanalysis (DC) of DES S-Boxes:

The differential cryptanalysis (DC) of S-Box endeavors to record the number of various output difference patterns that arises due to the fixed input difference in a Differential Distribution Table (DDT). The DES team at IBM, even though they had the complete knowledge of DC [27] during seventies, had not reported the same in time. Biham and Shamir [13] first reported DC in 1990 for DES-like S-Boxes. If many output difference patterns do not find place in the DDT for all possible input difference patterns and if such number of non-occurrences become greater that 50%, one can conclude that it is difficult to forecast the existence of those differences indicating the S-Box DC resistant.

The 4-bit indices of an S-Box are sequentially put in X[16][4] entries and the corresponding data in the S-Box are put in Y[16][4] entries. One considers a fixed input difference Xdash[i][j] taken from one input vector of X[16][4] and calculates row index ‘a’ of the table[a][b]. It computes Xstar[16][4] from the XOR of Xdash[i][j] and X[16][4] and calculates the index ‘m’ corresponding to four bits of each Xstar[i][4], finds Ystar[m][4]=Y[m][4]. For each Ystar[m][4] it computes Ydash[i][j] from the XOR of Xdash[i][j] and X[16][4]. The column index ‘b’ of table[a][b] is computed for each of Ydash[i][j] and the table[a][b] is upgraded by unity meaning that Ydash[i][j] has occurred once and is noted in the matrix element (a,b) of the table[a][b]. The related program design can be written as,

```
//Initialize the table[16][16] array
for (a=0;a<16;a++) for (b=0; b<16;b++) table
[a][b]=0; for (k=0; k<16;k++){
for (j=0; j<4;j++) xdash[k][j]=x[k][j];
for (j=0, l=0; j<4, l<4; j++, l++)
t[l]=xdash[k][j]; a=b2d ( t);
for (i=0; i<16; i++){
for(j=0;j<4;j++)xstar[i][j]=xdash[k][j]^x[i][j];
for(j=0,l=0;j<4,l<4;j++,l++)t[l]=xstar[k][j];
m=b2d(t);
for(j=0;j<4;j++)ystar[m][j]=y[m][j];
for(j=0;j<4;j++)ydash[i][j]=ystar[m][j]^y[i][j]
for(j=0,l=0;j<4,l<0;j++,l++)t[l]=ydash[i][j];
b=b2d(t);
table[a][b]++;
}     }
```

2.6. Linear Cryptanalysis of DES S-Boxes:

The basic idea of Linear Cryptanalysis of S-Box is to see if a linear relationship of its sixteen 4-bit inputs (X_i) exists with its sixteen 4-bit outputs (Y_j). Matsui [14] first introduced the idea of Linear Cryptanalysis in 1994 by devising a mechanism to look for linear relationship between input and output of DES ciphers. Heys [31] in a tutorial considering the idea of linear cryptanalysis of Matsui, proposed a Linear Approximation Table (LAT) for linear cryptanalysis of a 4-bit S-Box. One should note that for a particular input, there are 16 entries in the LAT for all the output patterns and for a particular output, there are identical entries in the LAT for all input patterns. The number of times (C_{ij}) the linear relationship between a particular input and a particular output is satisfied is recorded [14] and the numbers (N_{ij}) beyond 8 are appropriately entered in the LAT. An entry in the LAT becomes zero if N_{ij} = 8, and for such entry the probability of a linear relationship between a particular input and output in the S-Box is ½. If the number of zeros in the LAT is more than 50%, one can conclude that for majority of cases no predictable linear relation between inputs and outputs exists and the S-Box is LC resistant.

The algorithm to execute the linear cryptanalysis for 4-bit S-Boxes following Heys [31] considers 4–bit Boolean variables A_i and B_j whose i and j are the decimal indices varying from 0 to 15 and A_i and B_j are taking corresponding bit values from [0000] to [1111]. The algorithm to fill the (16 x 16) elements of the LAT is,

```
for (i=0;i<16;i++) {
A=0;
for(k=0;k<16;k++)
A=A+($A_{i0}$.$X_{k0}$+$A_{i1}$.$X_{k1}$+$A_{i2}$.$X_{k2}$+$A_{i3}$.$X_{k3}$)%2;
for (j=0;j<16;j++){
B=0;
for(k=0;k<16;k++)B= B+($B_{j0}$.$Y_{k0}$+$B_{j1}$.$Y_{k1}$+$B_{j2}$.$Y_{k2}$+$B_{j3}$.$Y_{k3}$)%2;
$S_{ij}$  = (A+B)%2;
if ($S_{ij}$==0) $C_{ij}$++;
$N_{ij}$  = $C_{ij}$ - 8;
      }
}
```

It may be noted that the first element in LAT is represented by $N_{0,0}$ while the last, by $N_{16,16.}$ The linear Boolean functions are those, which contain no product terms of the input variables.

2.7. Strict Avalanche Criterion of S-Boxes:

One 4-bit S-Box is composed of four 16-bit Boolean Functions, bf0, bf1, bf2 and bf3 and according to Kim, Matsumoto and Imai [29] an S-Box satisfies SAC criterion, provided all the Boolean functions individually satisfies the SAC criterion for the four input vectors. The Kim, Matsumoto and Imai proposed to flip one bit of a particular position of one input and to note the bf-value before the flip and after the flip and if they are different, the result of the flip on {bf} is noted as '1'. The same bit position is flipped for all 16 inputs and the results of the flip on 'bf' are noted. Out of the 16 operations of flipping the bit of a particular input position, if the results on 'bf' becomes '1' for eight times and if such identical results are obtained for flipping bits in other positions of all inputs, the {bf} can be considered to satisfy SAC. Once similar cases happen for other bf's, one can state that the S-Box satisfies SAC.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3285v1 | CC BY 4.0 Open Access | rec: 26 Sep 2017, publ: 26 Sep 2017

The algorithm of Kim, Matsumoto and Imai can be written is as,

$$D_{i,v}=F_i(x) \oplus F_i(x \oplus e_v) \text{ and } Wt\,(D_{i,v}) = 2^{k-1} \quad . \quad i, v=0,1,\ldots\ldots.,k-1 .$$

Sokolov [30] applied the idea of Kim, Matsumoto and Imai [30] to synthesize one 4-bit S-Box and one 8-bit S-Box, both satisfy SAC. The flipping of bits on particular positions are made by proposing 1-bit four e_v vectors as, e_0 {0001}, e_1 {0010}, e_2 {0100} and e_3 {1000}. The pseudo code of the algorithm can be written as,

```
//for a particular S-Box one value of s is taken for
which there are four bf
for (i=0; i<16; i++) for (j=0; j<16; j++) D[i][j] = 0;
for(f=0; f<4; f++) {
for(v=0; v<4; v++) {
switch (v) {
 0:ev[3]=0; ev[2]=0; ev[1]=0; ev[0]=1;
 1:ev[3]=0; ev[2]=0; ev[1]=1; ev[0]=0;
 2:ev[3]=0; ev[2]=1; ev[1]=0; ev[0]=0;
 3:ev[3]=1; ev[2]=0; ev[1]=0; ev[0]=0;
 }
for(x=0; x<16; x++) {
for(j=0,l=0;j<4,l<4;j++,l++)
t[l] = 16bt4x[s][i][j] ^ ev[j]
m=b2d(t);
r=16bt4bf[s][f][x] ^ 16bt4bf[s][f][m] if
(r==1) D[f][v]++;
            } // end of for loop of x
          } // end of for loop of v
        } // end of for loop of f
```

If all entries in (16 x 16) D- matrix is found to 8, the given S-Box is SAC satisfied and cryptographically strong.

3. Result and Discussion.

The computational results of 32 DES S-Boxes divided in 8 groups, each group having four, are presented in Table 1 [32] in four pages, with two groups in one page. It is noted that of all the required 128 BFs, 102 are non-repeating and 13 pairs are repeating ones and all 115 are balanced and non-linear. Among the repeating 13 pairs, 10 have the non-linearity 4, while the rest 3 have 2. Of the 102 non-repeating BFs, 82 have the non-linearity 4, while the rest 20, have 2. The 13 BF pairs with non-linearity 2, of which 3 are repeating ones, do not satisfy SAC, as defined by Travares [21], while the 102 BFs with non-linearity 4, of which 10 are the repeating ones do satisfy the SAC. The 4-bit ANF coefficient {a_{14}}, of all the 115 BFs is absent and does not contribute to non-linearity.

All the S-Boxes satisfy the BIC as defined by Travares [21] and also satisfy DC as defined by Biham and Shamir [13] and the result of DC lies between 162 and 174. All S-Boxes except the S-Box 12 do not satisfy LC as defined by Matsui [14] since all assume results less than 128, except the one assuming 131.

Following Travares [21], a BF is considered to satisfy SAC if it becomes SAC proof with respect one of the four input vectors. If one considers the Travares's condition of BF-level SAC and imposes a condition that one of the four BFs of an S-Box satisfies Travares's SAC, as the necessary condition for the S-Box to satisfy SAC, all the 32 DES S-Boxes are found to satisfy S-Box-level SAC. However, Kim, Matsumoto and Imai [29] considered a BF SAC-proof when the concerned BF satisfies SAC with respect to all the four input vectors. They also considered an S-Box to satisfy SAC if all its four BFs independently satisfy their BF-level SAC.

Following the SAC proposed by Kim, Matsumoto and Imai [29], Sokolov[30] demonstrated that a 4-bit S-Box and a 8-bit S-Box do satisfy the SAC. The proposed SAC test of all the DES S-Boxes has been undertaken and it is found that no S-Boxes satisfy the S-Box-Level SAC. The related results are shown in the column 17 of Table 1 [32]. In fact, it is found that no BF of the 32 DES S-Boxes satisfy the BF-level SAC conditions proposed by Kim, Matsumoto and Imai.

4. Concluding Remarks

The DES S-Boxes are so designed that all of them become DC and BIC resistant. The present analysis indicates that the designers of DES S-Boxes had the complete knowledge of BFs and the S-Box-level BIC and DC, but not of the S-Box-level LC and SAC. It seems that out of the 12870 balanced and non-linear BFs, there should be sufficient BFs with non-linearity 4, such that no BFs with non-linearity 2 need not to be considered. Probably the compulsion to choose 26 BFs with non-linearity 2, was the need to make all the DES S-Boxes BIC and DC resistant. In order to resolve such a riddle, it is necessary to approach design of 4-bit S-Boxes in a comprehensive manner from the BF and ANF angles.

References

1. Data Encryption Standard, Federal Information Processing Standards Publication No. 46, National Bureau of Standards, January 15, 1977.
2. http://en.wikipedia.org/wiki/Feistal_Cipher
3. http://en.wikipedia.org/wiki/Data_Encryption_Standard
4. Feistel, H. "Block Cipher Cryptographic System", US Patent 3798359 (Filed June 30, 1971)
5. Feistel, H. "Cryptography and Computer Privacy", Scientific American, Vol. 228, No. 5, 15-23, May 1973.

6. http://en.wikipedia.org/wiki/Lucipher_(Cipher)
7. Diffe, W and Hellman, M."Exhaustive Cryptanalysis of the NBS Data Encryption Standard", Computer Vol. 10, No. 6, pp 74-84, June 1977
8. Konheim, A."Cryptography: A primer", Johm Wiley and Sons, 1981.
9. Meyer, C. and Matyas, S."Cryptography - A New Dimension in Computer Data Security", Wiley Interscience, 1982.
10. Diffe, W."Cryptographyc Technology: Fiften yera Forecast", Published in "Advances in Cryptography - A Report on Crypto81" Edited by Allen Gersho, Report 82-04, Dept. CSE, University of California, Santa Barbara, 1982.
11. Sorin, A."Lucifer: A Cryptographic Algorithm", Cryptologia, Vol. 8, No. 1, pp. 22-35, 1984
12. Biham, E. and Shamir, A."Differential Cryptanalysis of DES-like cryptosystems", (1990), http://sota.gen.nz/crypt_blues/biham91differential.pdf
13. Matsui, M."Linear Cryptanalysis of DES Cipher", Advances in Cryptology (EuroCrypt'93), LNCS 765 pp 386-397, 1993,
14. RSA Security. “RSA to Launch ‘DES Challenge II’ at Data Security Conference." Dec,1997. 25 June 2001. www.rsasecurity.com/news/pr/971217.html
15. http://en.wikipedia.org/wiki/EEF_DES_Cracker
16. http://news.cnet.com/Record-set-in-charge-56-bit-crypto/2100-1017-3-220333.html
17. Paul Van De Zande,” The Day DES Died”, SANS Institute 2001.
18. http://www.sciengines.com/copacobana/faq.html
19. Webster, A.F. and Travares, S.E."On Design of S-Boxes", Advances in Cryptology : Proc. of Crypto 85, Springer-Verlag pp. 523-534, 1986,
20. Zeng, K.C., Yang, J.H. and Dai, Z.T.”Pattern of Entropy drop of the key in an S-Box of the DES”, Advances in Cryptology : Proc. of Crypto 87, Springer-Verlag, pp. 438-444, 1988.
21. Adams, C. and Tavares, S."The Structured Design of Cryptographically Good S-Boxes", Journal of Cryptology, Vol. 3, pp. 27-34, 1990.
22. Hussain, I. et. al., "Construction of Cryptographically Strong 8x8 S-boxes”, World Applied Sciences Journal, Vol. 13, No. 11, pp.2389-2395, 2011
23. Adams, C. and Tavares, S."A Note on the Generation and Counting of Bent Functions”, Ieee Trans. Information Theory, Vol. 36, pp. 1170-1173, 1990.
24. Bruce Schneier, Description of a New Variable-Length Key, 64-bit Block Cipher (Blowfish). Fast Software Encryption 1993: 191–204.
25. Bruce Schneier, The Blowfish Encryption Algorithm—One Year Later, Dr. Dobb's Journal, 20(9), p. 137, September 1995.

26. C.M. Adams. (1997). "Constructing Symmetric Ciphers Using the CAST Design Procedure", Designs, Codes, and Cryptography, 12(3), pp. 283–316.
27. Coppersmith, D."The Data Encryption Standard (DES) and its strength against attacks", IBM Journal of Research and Development Archive Vol. 38, No.3, pp. 15-23, pp. 243-250, 1994.
28. Bruce Schneier, “Applied Cryptography”, 2nd ed. John Wiley, 1996.
29. K.Kim,T.Matsumoto, and H.Imai” A recursive construction method of S-Boxes Satisfying the Strict Avalanche Criterion.”in Proc. of CRYPTO’90(Springer-Verlag,1990), pp. 565-574
30. A.V.Sokolov. “Constructive Method for the Synthesis of Nonlinear S-Boxes Satisfying the Strict Avalanche Criterion”,ISSN 0735-2727,Radioelectronics and Communication Systems,2013,vol.56,No.8,pp.415-423,Alerton Press Inc. 2013.
31. H.M.Heys, “A Tutorial on Linear and Differential cryptanalysis.” , http://www.engr.mun.ca/~howard/PAPERS/ldc_tutorial.pdf.
32. “Table 1. Analysis of DES S-Boxes” https://www.academia.edu/attachments/32974731/download_file