International Journal of Computer Applications (0975 - 8887)
Volume 168 - No.5, June 2017

Differential Cryptanalysis on Block Ciphers: New Research Directions

Vikas Tiwari
Research Fellow
CR Rao AIMSCS
UOH Campus, Hyderabad-500046, India

Priyanka Garg
Sr. Project Technical Assistant
IIT-Bombay
Mumbai, India

Ajeet Singh
Research Fellow
CR Rao AIMSCS
UOH Campus, Hyderabad-500046, India

ABSTRACT

Differential Cryptanalysis is a powerful technique in cryptanalysis, applied to symmetric-key block ciphers. It is a chosen plain-text attack which means the cryptanalyst has some sets of the plain-text and the corresponding cipher-text pairs of his choice. These pairs of the plain-text are related by a constant difference. Basically it is the study of how differences in input information can affect the resultant difference at the output.
In this paper, differential cryptanalysis is applied on substitution-permutation network and data encryption standards cipher. The survey is based on the analysis of a simple, yet realistically structured, basic Substitution-Permutation Network cipher. Along with this, the paper also presents our contribution in this paper as well as our future research work.

Keywords

Differential Cryptanalysis, Symmetric Key, Substitution Permutation Network (SPN), Security, Differential Attack

1. INTRODUCTION

Cryptosystems are generally divided in two types: *Symmetric Key Cryptosystems*, where same key is used by the sender and the receiver for encryption and decryption respectively. Thus key need to be kept as private, hence the Symmetric key Cryptosystems can also be known as private key cryptosystems. The secure distribution of key associated with symmetric key cryptosystems is a challenging task. Data Encryption Standard (DES) and Advanced Encryption Standards (AES) are examples of symmetric key cryptosystems.[1][2] Unlike symmetric key cryptosystems, *asymmetric key cryptosystems*, which uses two keys, called private key and public key. It relies on one key for encryption and the other for decryption. These two keys are different but are related. The RSA algorithm is an example of asymmetric key cryptosystems.
The Differential cryptanalysis was developed by Biham et al. in 1990 [6]. It is one of the seminal work in the area of cryptanalysis. It is chosen plain-text attack. In Differential Cryptanalysis, the main task is to study the propagation of differences from round to round inside the cipher and find specific differences, which propagate with relatively high probability. Such pairs of input-output differences can be used to recover some bits of the secret key.[7]

1.1 Motivation and Contribution

Block cipher is a procedure for encrypting plaintext where key and algorithm are applied to a data block. An example of such a symmetric key cryptosystem is Data Encryption Standard (DES). Originally in 1970's it was developed by IBM. Later many researchers have performed cryptanalysis on DES upto specific rounds.
In this paper, we have presented in depth literature review and have performed our cryptanalysis on 3-Round DES. Further work will be extended as the cryptanalysis on more number of rounds on DES.

1.2 Organization of the paper

Rest of the paper is organized as - section 2 discusses some required preliminaries. Related work is presented in section 3. In section 4, we have discussed differential cryptanalysis on SPN. In section 5, we have discussed attack on 3-Round DES. Finally section 6 concludes the paper.

2. PRELIMINARIES

Substitution Permutation Network (SPN):- It is a mechanism to used in designing a block cipher. Here substitution does confusion and permutation does diffusion.[7][8]
Confusion is described as being the use of enciphering transformations that complicate the determination of how the statistics of the cipher-text depend on the statistics of the plain-text. This is achieved by using a complex substitution algorithm. While **Diffusion** dissipates the statistical structure of the plain-text within the cipher-text so that attacker cannot determine plain-text corresponding to the cipher-text.
The principle of diffusion and confusion is achieved by applying substitution and permutation to the plain-text over and over again. Iterated Block Cipher is based on this principle. Thus SPN is a type of iterated block cipher. A basic SPN structure is shown in Figure 1. It has four rounds. Each round consists of substitution, permutation and key mixing. The input size of plain-text is 16 bit and key size is 32 bit. Firstly we convert our 32-bit key into round keys of 16-bit each with help of key scheduling algorithm. Now each of these round key is xor-ed with the input it gets in every round. SPN cipher takes 16-bit block of plain-text as input and divides it into four sub-blocks of 4-bit each. Now each of these sub-block goes into key mixing block. After key mixing, it goes to the S-box as shown in Table 1. The fundamental property of an S-box is that it is

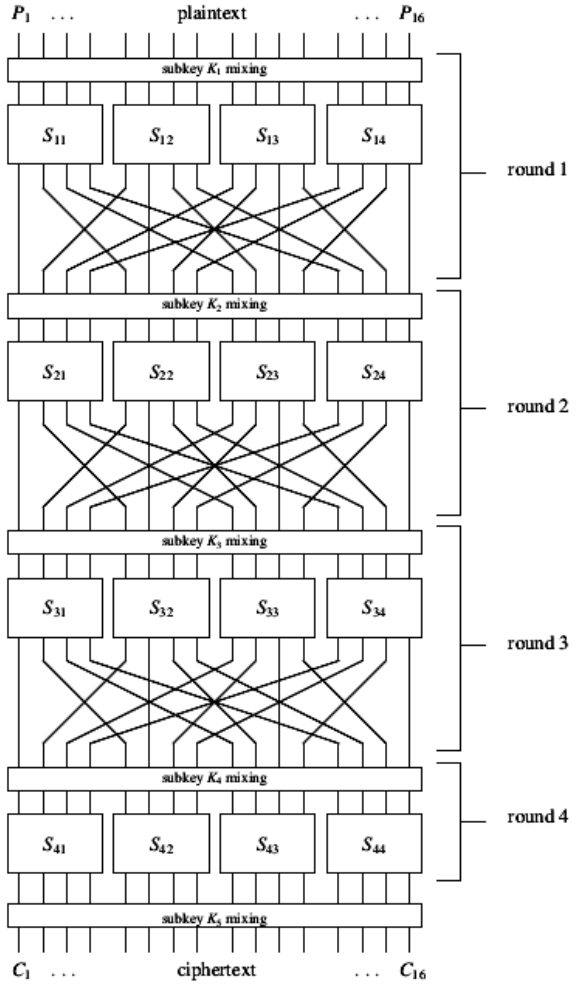

Fig. 1. Basic SPN Structure[1]

a nonlinear mapping, that is, the output bits can not be represented as a linear function of the input bits.

I/P	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
O/P	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7

Table 1: Substitution Box

Now, output of s-box is permuted by permutation box. P-box performs the permutation of the bit position. It is shown below in Table 2.

I/P	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
O/P	1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16

Table 2: Permutation Box

3. RELATED WORK

Firstly IBM has designed iterated cryptosystem called Lucifer [11], to overcome the increasing need for the data and information security in its products. The complete design and structure of the Data Encryption Standard came in existence in [12]. The procedure of Formal Coding, where formal expression of each bit in the ciphertext is the XOR SOP form of the bits of the plaintext and the key was presented in [13]. The manipulations in a formal way of these expressions may reduce the key search attempt.

The prime objective of differntial cryptanalysis, which was developed by Biham et al. in 1990 [6], is to study the propagation of differences from one round to last round inside the cipher and find the appropriate differences, which propagate with relatively high probability. Such pairs of input-output differences can be utilized to recover some bits of the secret key.[7]

Schaumuller-Bichl [14] [15] explored this method and formulates that it needs a significant amount of system memory, which makes the idea impractical. In 1987, Davies [16] given a known plaintext cryptanalytic attack on DES.

Over past years, several cryptosystems which are standards of DES were presented. Schaumuller-Bichl proposed three such types of cryptosystems [14] [17]. Another standard is the Fast Data Encryption Algorithm (FEAL). It was designed to be more efficiently practical and implementable on an 8-bit microprocessor. It's first version had four rounds.[18] Later it was broken by Den Boer [19] using a chosen plaintext attack. The inventors of FEAL given a new version, called FEAL-8, with 8-rounds [20] [21].

4. DIFFERENTIAL CRYPTANALYSIS ON SPN

For differential attack on SPN, we need to first find a differential characteristic (sequence of input and output differences) for one round that has high probability and then for the whole cipher, as output difference from one round corresponds to the input difference for the next round [3]. After determining differential characteristics, we can derive the key used in the last round of the cipher. To construct high probability differential characteristics, we should examine properties of nonlinear part of our cipher, i.e S-boxes.

DDT table is constructed for every different s-box. Since we are using the same s-box in SPN, only one DDT table will be constructed. DDT stores the no of occurrences of output difference C', for a given input difference P'.

DDT table for SPN cipher is shown in Table 5. It is constructed in the following manner. Let P' denote the input difference. Now we will search for P and P^* (called plain-text pairs) whose difference is P'. Then each of these P and P^* is passed through s-box to find corresponding C and C^* (called cipher-text pairs). This C and C^* is xor-ed to get output difference C'.

Lets consider P' = 0100. For each pair P and P^* having difference 0100, we will compute C = $\pi_s(P)$, $C^* = \pi_s(P^*)$ and C' = C $\oplus$ C^*, where $\oplus$ denotes the x-or operation. This is shown in Table 3 where P' = P $\oplus$ P^*

From the last column of Table 3, we will count the occurrence of each value as mentioned in Table 4. We can see that only five of the possible 16 values occur. Now this result will be stored in DDT table in row which is equal to input difference 0100 and each column in this row will be filled with the count of the output difference from Table 4. This is shown in DDT Table 5.

In this manner we can construct the DDT table by taking every possible value of P'.

P	P^*	C	C^*	C'
0000	0100	1110	0010	1100
0001	0101	0100	1111	1011
0010	0110	1101	1011	0110
0011	0111	0001	1000	1001
0100	0000	0010	1110	1100
0101	0001	1111	0100	1011
0110	0010	1011	1101	0110
0111	0011	1000	0001	1001
1000	1100	0011	0101	0110
1001	1101	1010	1001	0011
1010	1110	0110	0000	0110
1011	1111	1100	0111	1011
1100	1000	0101	0011	0110
1101	1001	1001	1010	0011
1110	1010	0000	0110	0110
1111	1011	0111	1100	1011

Table 3: Sample Difference Pairs of S-box

0000	0001	0010	0011	0100	0101	0110	0111
0	0	0	2	0	0	6	0
1000	1001	1010	1011	1100	1101	1110	1111
0	2	0	4	2	0	0	0

Table 4: Count of every value from Difference Pairs of S-box

P'/C'	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	2	0	0	0	2	0	2	4	0	4	2	0	0
2	0	0	0	2	0	6	2	2	0	2	0	0	0	0	2	0
3	0	0	2	0	2	0	0	0	0	4	2	0	2	0	0	4
4	0	0	0	2	0	0	6	0	0	2	0	4	2	0	0	0
5	0	4	0	0	0	2	2	0	0	0	4	0	2	0	0	2
6	0	0	0	4	0	4	0	0	0	0	0	0	2	2	2	2
7	0	0	2	2	2	0	2	0	0	2	2	0	0	0	0	4
8	0	0	0	0	0	0	2	2	0	0	0	4	0	4	2	2
9	0	2	0	0	2	0	0	4	2	0	2	2	2	0	0	0
A	0	2	2	0	0	0	0	0	6	0	0	2	0	0	4	0
B	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
C	0	2	0	0	2	2	2	0	0	0	0	2	0	6	0	0
D	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
E	0	0	2	4	2	0	0	0	6	0	0	0	0	0	2	0
F	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

Table 5: Difference Distribution Table

4.1 Constructing Differential Characteristics

The construction of differential characteristics is illustrated with an example. Consider Figure 2, which involves S_{12}, S_{23}, S_{32} and S_{33} to construct differential characteristic. Note that, here we are using U_i to denote the input to the *i-th* round s-boxes and Y_i to denote the output of the *i-th* round s-boxes.

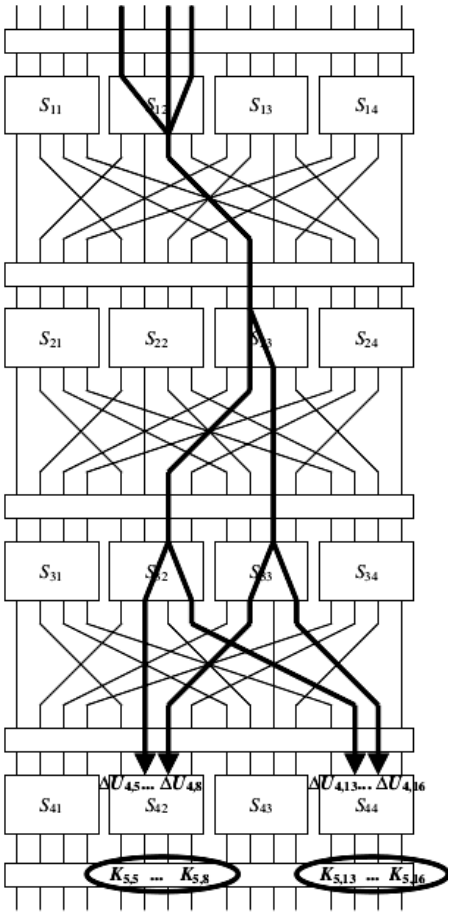

Fig. 2. Sample Differential Characteristic[2]

In round 1, suppose our 16 bit input is $P' = U'_1$ = 0000 1011 0000 0000. This input passes through four s-boxes as, 0000 through S_{11}, 1011 through S_{12}, 0000 through S_{13}, 0000 through S_{14} respectively as given in Figure 3.1. Only S_{12} box is active (An active s-box is the one whose input is non-zero.) as it has non-zero input and input to all other s-boxes is zero. So, all s-boxes gives output 0000 except S_{12} box which gives output as 0010 using substitution box Table 1. Now, 16 bit o/p from s-boxes Y'_1 is 0000 0010 0000 0000.
This 16 bit gets permuted through permutation box as shown in Table 2, that is, 7^{th} bit becomes 10^{th} bit and so on. The permuted output is U'_2 = 0000 0000 0100 0000. At this level we have completed round 1 where for fixed input difference P' = 1011, we are getting output difference C' = 0010. From DDT table, when input difference is 1011(B), output difference 0010(2) occurs eight times with probability 8/16 as we have 16 possible combinations. So, we get probability 8/16 from round 1 [9]. Output of round 1 is fed as input to the round 2.

In round 2, 16 bit input is U'_2 = 0000 0000 0100 0000. This input passes through four s-boxes as 0000 through S_{21}, 0000 through

[2]Howard M.Heys, "A tutorial on Linear and Differential Cryptanalysis," *Journal Cryptologia*, Volume 26, Issue 3, 2002.

S_{22}, 0100 through S_{23}, 0000 through S_{14} respectively. Here S_{23} box is active as its input is non-zero. So, S_{23} box gives output as 0110 using substitution box Table 1. The combined 16 bit o/p from s-boxes Y_2' is 0000 0000 0110 0000.
This 16 bit gets permuted through permutation box as shown in Table 2, that is, 10^{th} bit becomes 7^{th} bit and 11^{th} bit becomes 11^{th} bit and so on. The permuted output is U_3' = 0000 0010 0010 0000. At this level we have completed round 2 where for fixed input difference P' = 0100, we are getting output difference C' = 0110. From DDT table, when input difference is 0100(4), output difference 0110(6) occurs six times with probability 6/16 as we have 16 possible combinations. So, we get probability 6/16 from round 2. Output of round 2 is fed as input to the round 3.

In round 3, 16 bit input is U_3' = 0000 0010 0010 0000. This input passes through four s-boxes as 0000 through S_{31}, 0010 through S_{32}, 0010 through S_{33}, 0000 through S_{14} respectively. Here, S_{32} and S_{33} box is active as it has non-zero input. The output from S_{32} box is 0101 and from S_{33} box is 0101 using substitution box Table 1.
Now, 16 bit o/p from s-boxes Y_3' is 0000 0101 0101 0000.
This 16 bit gets permuted through permutation box as shown in Table 2, that is, 6^{th} bit becomes 6^{th} bit and 8^{th} becomes 14^{th} bit and so on.The permuted output is X_4' = 0000 0110 0000 0110. At this level we have completed round 3 where for fixed input difference P' = 0010, we are getting output difference C' = 0101. From DDT table, when input difference is 0010(2), output difference 0101(5) occurs six times with probability 6/16 as we have 16 possible combinations. In this case we have two active s-boxes, so we will have two probabilities. So, we get two probabilities of 6/16 from round 3. Output of round 3 is fed as input to the round 4.
In determining the probability given plain-text difference P'. we have assumed that differential of first round is independent of the differential of the second round and so on. Hence probability of all occurring is determined by the product of the probabilities.

S_{12}: P' = B $\rightarrow C'$ with probability = 8/16
S_{23}: P' = 4 $\rightarrow C'$ with probability = 6/16
S_{32}: P' = 2 $\rightarrow C'$ with probability = 6/16
S_{33}: P' = 2 $\rightarrow C'$ with probability = 6/16
Product of probabilities = (8/16)*(6/16)*(6/16)*(6/16)
= 27/1024

There will be many plain-text pairs whose P' = 0000 1011 0000 0000 and they all will be encrypted during this cryptanalysis process. We will select those differential characteristics which will occur with probability 27/1024. These high probability plain-text pairs with P' are termed as right pairs and other plain-text pairs are wrong pairs.

4.2 Extracting Key Bits

We will extract the subkey used in the last round of the cipher. In last round, we will find the partial subkey corresponding to the active s-boxes and the rest of the key bits can be determined through exhaustive search.
Each of the plain-text pair (P_1,P_2) is encrypted and their corresponding cipher-text pair (C_1,C_2) is stored. For each of the cipher-text pair and for every possible partial subkey (last key used in the encryption), we do partial decryption until the input of the last round is obtained. Let the result of the partial decryption be (C_1', C_2'). If the resulted (C_1', C_2') is equal to our desired result then the count of the corresponding partial subkey gets incremented. This process will be repeated for all possible partial subkeys and

for $(L_1, L_2) \leftarrow (0,0)$ **to** (F, F)
  do $Count[L_1, L_2] \leftarrow 0$
for each $(x, y, x^*, y^*) \in \mathcal{T}$
  do **if** $(y_{<1>} = (y_{<1>})^*)$ **and** $(y_{<3>} = (y_{<3>})^*)$
    then **for** $(L_1, L_2) \leftarrow (0,0)$ **to** (F, F)
      do $v^4_{<2>} \leftarrow L_1 \oplus y_{<2>}$
$v^4_{<4>} \leftarrow L_2 \oplus y_{<4>}$
$u^4_{<2>} \leftarrow \pi_S^{-1}(v^4_{<2>})$
$u^4_{<4>} \leftarrow \pi_S^{-1}(v^4_{<4>})$
$(v^4_{<2>})^* \leftarrow L_1 \oplus (y_{<2>})^*$
$(v^4_{<4>})^* \leftarrow L_2 \oplus (y_{<4>})^*$
$(u^4_{<2>})^* \leftarrow \pi_S^{-1}((v^4_{<2>})^*)$
$(u^4_{<4>})^* \leftarrow \pi_S^{-1}((v^4_{<4>})^*)$
$(u^4_{<2>})' \leftarrow u^4_{<2>} \oplus (u^4_{<2>})^*$
$(u^4_{<4>})' \leftarrow u^4_{<4>} \oplus (u^4_{<4>})^*$
if $((u^4_{<2>})' = 0110)$ **and** $((u^4_{<4>})' = 0110)$
then $Count[L_1, L_2] \leftarrow Count[L_1, L_2] + 1$
$max \leftarrow -1$
for $(L_1, L_2) \leftarrow (0,0)$ **to** (F, F)
  do **if** $Count[L_1, L_2] > max$
    then $max \leftarrow Count[L_1, L_2]$
$maxkey \leftarrow (L_1, L_2)$
output $(maxkey)$

Fig. 3. Algorithm for Differential Attack on SPN.[3]

for all chosen plain-text cipher-text pairs. Eventually, the partial subkey which has the maximum count is expected to be the actual partial subkey.
This complete differential cryptanalysis algorithm is presented in Figure 3. In this algorithm, τ represents set of all plain-text and corresponding cipher-text pairs used in this attack. L_1 and L_2 take hexadecimal values. π_s^{-1} is the inverse of the s-box, used to partially decrypt the cipher-text. The (x, x^*, y, y^*) corresponds to all plain-text and cipher-text pairs. $y_{<1>}$ denotes first four bits of y, $y_{<2>}$ denotes next four bits of y and so on. The Count[] array here stores the count value of each possible partial subkey. This algorithm also performs a filtering operation to discard wrong plaintext-ciphertext pairs. All plaintext-ciphertext pairs does not allow us to find the relevant key bits. Thus, we had performed operation $y_{<1>} = y^*_{<1>}$ and $y_{<3>} = y^*_{<3>}$ to filter out the pairs. The pairs for which this condition holds are called right pairs. This filtering operation increases the efficiency of the attack.
After getting the partial subkey with maximum count value, we will find its probability as, prob = count/number of pairs, where number of pairs are the generated chosen plaintext-cipher text pairs used in the attack. In this case, we expect the probability to be 27/1024, which confirms we have correct subkey.

5. ATTACK ON 3-ROUND DES

As we have seen DES has 16 rounds but for cryptanalysis purpose we have reduced DES to 'n' rounds where n = 3. For this attack we have neglected initial permutation (IP) and its inverse as they do not have effect on cryptanalysis [4],[6]. To attack 3-round DES,

[3]Douglas R. Stinson, "Cryptography Theory and Practice," *Chapman Hall/CRC*, Third Edition, 2006.

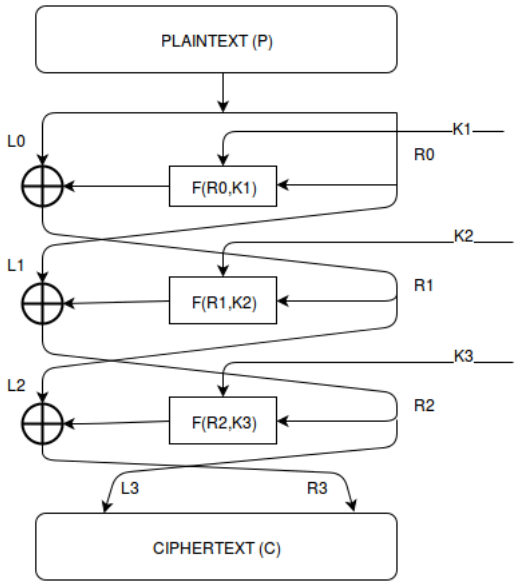

Fig. 4. 3-Round DES Structure

suppose we have a plain-text pair L_0R_0 and L_0^* R_0^* and corresponding cipher-text pair L_3R_3 and L_3^* R_3^*. A 3-round DES structure is shown in Figure 4. From this figure we can express R_3 as:

$$R_3 = L_2 \oplus f(R_2, K_3) \tag{1}$$

Since L_2 and R_1 are equal,

$$R_3 = R_1 \oplus f(R_2, K_3) \tag{2}$$

Further R_1 can be expressed as:

$$R_3 = L_0 \oplus f(R_0, K_1) \oplus f(R_2, K_3) \tag{3}$$

On giving L_0^* R_0^* as input to Figure 4.1, R_3^* can be expressed as,

$$R_3^* = L_0^* \oplus f(R_0^*, K_1) \oplus f(R_2^*, K_3) \tag{4}$$

R_3' is the xor-ed difference of R_3 and R_3^*. So, $R_3' = R_3 \oplus R_3^*$.

$$R_3' = L_0 \oplus f(R_0, K_1) \oplus f(R_2, K_3) \oplus L_0^* \oplus f(R_0^*, K_1) \oplus f(R_2^*, K_3) \tag{5}$$

As $L_0 \oplus L_{0^*} = L_0'$,

$$R_3' = L_0' \oplus f(R_0, K_1) \oplus f(R_0^*, K_1) \oplus f(R_2, K_3) \oplus f(R_2^*, K_3) \tag{6}$$

By taking $R_0 = R_0^*$ we get,

$$R_3' = L_0' \oplus f(R_2, K_3) \oplus f(R_2^*, K_3) \tag{7}$$

We know R_3' and L_0' so we can rewrite above equation as,

$$R_3' \oplus L_0' = f(R_2, K_3) \oplus f(R_2^*, K_3) \tag{8}$$

Let H and H^* be the two outputs of the eight s-boxes then,

$f(R_2,K_3) = P(H)$ and $f(R_2^*,K_3) = P(H^*)$

where P performs the permutation function. Then,

$P(H) \oplus P(H^*) = f(R_2,K_3) \oplus f(R_2^*,K_3)$

$$H' = H \oplus H^* = P^{-1}(R_3' \oplus L_0') \tag{9}$$

Now, $R_2 = L_3$ and $R_2^* = L_3^*$ are also known by this we can compute,

$$G = E(L_3) \tag{10}$$

and

$$G^* = E(L_3^*) \tag{11}$$

using the expansion function E. G and G^* are the input to the s-boxes in the 3rd round. We will use the triplet G, G^* and H' to attack.
Suppose we have, number of plain-text pairs and the corresponding cipher-text pairs as:

L_0R_0: "37580B1359ACEE20"
L_3R_3:"34E9174A5A2CB621"
$L_0^*R_0^*$: "264A020E59ACEE20"
$L_3^*R_3^*$: "023E68A49B1423D6"

From these pairs, we find the s-box inputs for round 3 from Equation (10) and (11). Here L_3 and L_3^* gets expanded to 48 bits that is:
G = 000110101001011101010010100010101110101001010100
G^*=000000000100000111111100001101010001010100001000
We know that input to the s-box is I = G $\oplus$ K where K represents key. The exclusive or (x-or) of the inputs of the eight s-boxes is :

$$I \oplus I^* = (G \oplus K) \oplus (G^* \oplus K) \tag{12}$$

Thus, I $\oplus$ I^* = G $\oplus$ G^*. So from this we can conclude that input x-or does not depend on the key bits K.

G' = G $\oplus$ G^*

G' = 000110101101011010101110101111111111111111101011100
The output of the s-boxes H' is computed using Equation 9.

$L_0' = L_0 \oplus L_0^*$
L_0 = 001101110101100000000101100010011
L_0^* = 001001100100101000000001000001110
L_0' = 000100010001001000000100100011101
$R_3' = R_3 \oplus R_3^*$
R_3 = 010110100010110010110110000100001
R_3^* = 100110110001010000100011110100 10
R_3' = 110000010011100010010101111110011
$R_3' \oplus L_0'$ = 110100000010101010011100111011 10
$H' = P^{-1}(R_3' \oplus L_0')$
= 010110100000110100101110001001 11

Here P^{-1} is inverse initial permutation which is shown below in Table 6.

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Table 6: Inverse Initial Permutation

Now we have G, G^* and H'. For $1 \leq i \leq 8$, every six bits in G' (G_i') and four bits in H' (H_i'), we will find pairs whose x-or equal is to G_i' and on giving these pairs input to the s-box S_i their output x-or is equal to H_i'.
Let these pairs be denoted using Pairs(G_i', H_i'). If we knew G and G^* we could say,

$$G_i \oplus K_i \in Pairs(G_i', H_i') \tag{13}$$

From Equation 13, we can conclude that to find key value we can x-or the Pairs(G'_i, H'_i) value with G value. Next step is to tabulate these key values in eight counter array J_i. As each K_i is of 6 bits which would mean 0 to 63 in decimal, the array J_i would range from 0 to 63.
Continuing with previous example, we will find Pairs(G'_i, H'_i) using first 6 bits of the G'_1 and first 4 bits of the H'_1,

Pairs(000110,0101) ={110010, 110100}

Here G_1 = 000110, using Equation 13,

$$K_1 \in G_1 \oplus \text{Pairs}(000110,0101) = \{110100, 110010\}$$

Thus we will increment values 52(110100) and 50(110010) in the counter array J_1.
For next pair G'_2 and H'_2, the values will be incremented in the counter array J_2 and so on. We will repeat this process for all pairs in G' and C'. This whole method will be performed with more plaintext-ciphertext pairs until we get a unique value in each of the eight counter arrays J. The position of these unique values determine the key bits.
To get the initial 64-bit key we have to perform few more computations on the result obtained from these eight counter arrays. This is experimentally done and results are attached below.
We have taken three plaintext-ciphertext pairs and computed their G, G^*, G' and H' as explained above. Then found the Pairs(G'_i, H'_i) for $1 \leq i \leq 8$ and finally J values to be incremented in the eight counter arrays. These pairs are shown below denoted with L_0R_0, $L_0^*R_0^*$, L_3R_3 and $L_3^*R_3^*$.
L_0R_0=748502CD38451097 , L_3R_3=03C70306D8A09F10
$L_0^*R_0^*$ = 3874756438451097 , $L_3^*R_3^*$= 78560A0960E6D4CB
L_0R_0 = 486911026ACDFF31 , L_3R_3=45FA285BE5ADC730
$L_0^*R_0^*$ = 375BD31F6ACDFF31 , $L_3^*R_3^*$= 134F7915AC253457
L_0R_0=357418DA013FEC86 , L_3R_3=D8A31B2F28BBC5CF
$L_0^*R_0^*$ = 12549847013FEC86 , $L_3^*R_3^*$= 0F317AC2B23CB944

For the first pair we incremented at position number as given below in the eight counter arrays $J_1, J_2, ..., J_8$.
Pairs(0,9) = 0,7,40,47 , J_1 = 0,7,40,47
Pairs(7,6) = 2,53,12,59 , J_2 = 5,50,11,60
Pairs(56,5)=4,54,20,38,21,39,25,43,J_3=60,14,44,30,45,31,33,19
Pairs(14,13) = 50,39,14,44,18,48, J_4 = 11,41,0,34,28,62
Pairs(32,5) = 25,56 , J_5 = 57,24
Pairs(6,11) = 1,19 , J_6 = 7,21
Pairs(32,6) = 6,39,13,44 , J_7 = 28,7,45,12
Pairs(12,7) = 35,61,36,58 , J_8 = 47,49,40,54

The output for the second pair is given below. In the same eight counter arrays we incremented the values according to the output shown below,
Pairs(40,9) = 7,13,52,62 , J_1 = 47,37,28,22
Pairs(11,12) = 3,46,8,37,14,35,26,55,30,51,
J_2 = 8,37,3,46,5,40,17,60,21,56
Pairs(63,9)=44,58,J_3=5,19
Pairs(52,12) = 0,42,1,43,10,32,11,33,20,62,21,63,30,52,31,53,
J_4 = 52,30,53,31,62,20,63,21,32,10,33,11,42,0,43,1
Pairs(5,1) = 17,59,29,55 , J_5 = 20,62,24,50
Pairs(16,15) = 4,38,23,53 , J_6 = 20,54,7,37
Pairs(11,5) = 0,41,9,32,12,37,14,39,
J_7 = 11,34,2,43,7,46,5,44
Pairs(54,6) = 0,0 , J_8 = 54,54

Finally, third pair gave the values for eight counter arrays as given below,
Pairs(59,13) = 4,62,20,46,26,32 , J_1 = 63,5,47,21,33,27
Pairs(49,5) = 2,45,11,36,27,52 , J_2 = 51,28,58,21,42,5
Pairs(20,7)=7,53,16,34,J_3=19,33,4,54
Pairs(6,5) = 6,34,12,40 J_4 = 0,36,10,46
Pairs(35,13) = 17,29,37,41,55,59 , J_5 = 50,62,6,10,20,24
Pairs(54,11) = 1,2,5,6,17,18,49,50,53,54,57,58,
J_6 = 55,52,51,48,39,36,7,4,3,0,15,12
Pairs(37,2) = 3,62,7,58,9,52,26,39,27,38,31,34,
J_7 = 38,27,34,31,44,17,63,2,62,3,58,7
Pairs(31,11) = 46,53 , J_8 = 49,42
At last, we get our J arrays for three rounds. In these arrays we got the unique values at:

J1: 47, J2: 5, J3: 19, J4: 0, J5: 24, J6: 7, J7: 07, J8: 49

We convert these integer values into binary to get 48 bits. We will use key schedule for round three in DES [5][8][10] to get 48 bits of the key as shown below in Table.

51	27	10	36	25	58	9	33	43	50	60	18
44	11	2	1	49	34	35	42	41	3	59	17
61	4	15	30	13	47	23	6	12	29	62	5
37	28	14	39	54	63	21	53	20	38	31	7

Table 7: Key Schedule for Round 3

This key schedule is for 56 bits, so the rest of the bits will be unknown. Also, our key is of 64-bits. These extra 8 bits are parity bits which will be added based on odd parity. Since very few bits are unknown, we can apply exhaustive search and then calculate odd parity over them. The complete key (in hexadecimal format) is:

"1A624C8520DEC46"

6. CONCLUSION

Our contribution in this paper and further research directions are presented as below:-

6.1 Contribution

This paper looks into the design and cryptanalysis of symmetric block ciphers. We briefly explained the Substitution Permutation Network (SPN) and Data Encryption Standard (DES) with their implementation. The well known chosen plain-text attack, Differential Cryptanalysis, is practised for extraction of key bits. Next we have presented the construction of Difference Distribution Table (DDT) which is used to find high probable characteristic to attack on SPN for recovering the key.
We have applied this cryptanalysis technique to DES reduced to 3-round and 6-round where we have differentiated between wrong and right pairs so that we can discard wrong pairs to get relevant key bits. To our knowledge, differential cryptanalysis on DES was discussed theoretically and we have done it practically and got correct results.

6.2 Future Research Directions

In near future, we are intended to extend this attack to DES with more number of rounds. Another thing worth pursuing in DES is how to generate high probable differential characteristic.

7. REFERENCES

[1] Douglas R. Stinson, Cryptography Theory and Practice, Chapman Hall/CRC, Third Edition, 2006.

[2] William Stallings, Cryptography Theory and Network Security, Pear- son Education, Fourth Edition, 2006.

[3] Howard M.Heys, A tutorial on Linear and Differential Cryptanalysis, Journal Cryptologia, Volume 26, Issue 3, 2002.

[4] Douglas R. Stinson, Cryptography Theory and Practice, CRC Press, First Edition, 1995.

[5] Eli Biham, Adi Shamir, Differential Cryptanalysis of Data Encryption Standards, Springer-Verlag New York, Inc., First edition, 1993.

[6] Eli Biham, Adi Shamir, Differential Cryptanalysis of DES-like Cryp- tosystems, The Weizmann Institute of Science Department of Apllied Mathematics, July 1990.

[7] Feistel, H. 1973, Cryptography and Computer Privacy, Scientific American, 228(5): 15-23.

[8] H. M. Heys, S. E. Tavares, Substitution-Permutation Networks Resis- tant to Differential and Linear Cryptanalysis, Journal of Cryptology, Vol 9, No 1, pp. 1-19, 1996.

[9] K. Chun, S. Kim, S. Lee, S. H. Sung, S. Yoon, Differential and linear cr yptanalysis for 2-round SPNs, Information Processing Letters, Elsevier, 2002.

[10] National Bureau of Standards, Data Encryption Standard, G.S. De- partment of Commerce, FIPS pub. 46, January 1977.

[11] H. Feistel, Cryptography and data security, Scientific American, Vol. 228, No. 5, pp. 15-23, May 1973.

[12] National Bureau of Standars, Data Encryption Standard, FIPS publication, No. 46, U.S. Department of Commerce, January 1977.

[13] M. E. Hellman, R. Merkle, R. Schroppei, L. Washington, W. Diffie, S. Pohlig, P. Schweitzer, Results of an Initial Attempt to Cryptanalyze the NBS Data Encryption Standard, Standford University, September 1976.

[14] I. Schaumuller-Bichl, Zur Analyse des Data Encryption Standard und Synthese Verwandter Chiffriersysteme, Ph.D. Thesis, Linz University, May 1981.

[15] I. Schaumuller-Bichl, Cryptanalysis of the Data Encryption Standard by the method of formal coding, Cryptolooia, Proceedings of CR YPTO 82, pp. 235-255, 1982.

[16] D. W. Davies, Private communications.

[17] I. Schaumuller-Bichl, On the Design and Analysis of New Cipher Systems Related to the DES, Technical Report, Linz University, 1983.

[18] A. Shimizu, S. Miyaguchi, Fast Data Encryption Algorithm Feal, Abstracts of EUROCRYPT 87, pp. VII-11-VII-14, April 1987.

[19] B. Den Boer, Cryptanalysis of F.E.A.L, Advances in Cryptolooy, Proceedings of EUROCRYPT 88, pp. 293-300, 1988.

[20] A. Shimizu, S. Miyaguchi, Fast Data Encryption Algorithm Feai, Advances in Cryptolooy, Proceedinos of EUROCRYPT 87, pp. 267-278, 1987.

[21] S. Miyaguchi, A. Shiraishi, A. Shimizu, Fast data encryption algorithm Feal-8, Review of Electrical Communications Laboratories, Vol. 36, No. 4, pp. 433-437, 1988.