IJCSE International Journal of Computer Sciences and Engineering Open Access
Review Paper Volume-5, Issue-2 E-ISSN: 2347-2693

A New Perspective of Inferring from the output of Linear Cryptanalysis Attack

Dipanjan Bhowmik[1*], Avijit Datta[2], Sharad Sinha[3]

[1*]Department of Computer Science & Application, University of North Bengal, Siliguri, India
[2]Department of Computer Science & Application, University of North Bengal, Siliguri, India
[3]Department of Computer Science & Application, University of North Bengal, Siliguri, India

[*]*Corresponding Author: howzat.dipanjan@gmail.com*

Available online at: www.ijcseonline.org

Received: 05/Jan/2017 Revised: 12/Jan/2017 Accepted: 03/Feb/2017 Published: 28/Feb/2017

***Abstract*— In this paper the results obtained from Linear Cryptanalysis attack developed by Matsui has been analyzed in a different perspective and inferences were drawn accordingly[2]. Here, a simple toy cipher has been put to Linear Cryptanalysis attack in order to understand the attack from a different perspective. The results thus obtained may become a guide towards designing block ciphers that can withstand Linear Cryptanalysis attacks.**

***Keywords*— Linear Cryptanalysis, Linear Approximation Table[1], S-Box, Toy cipher, Parity.**

I. INTRODUCTION

If one feeds a random input with a particular property into a magic box and can guess the corresponding property in the output, the magic box is somewhat linear [5]. For example, one imagines that the box takes an input and adds one to it. Now, let's assume that the property which is looked at is whether the input/output is even. By feeding it an input, one knows the property will be opposite in the output every single time. In other words, adding one to an even number will always produce an odd number and vice versa. Thus, such a magic box will be completely linear with respect to divisibility by 2. In an iterative cipher, substitution box(s) (S-Box(s)) add non linearity to it. Ideally, an S-Box should accept an input with property X and produce an output having property Y exactly 50% of the time.

The property, which is being looked at in Linear Cryptanalysis is *Parity*[1][2][6][7][8].

Definition

Parity*: It is a Boolean value (a 0 or a 1), that is obtained if bitwise XOR operation is performed on some or all of the bits of a number expressed in its binary equivalent form. The group of bits that are being XORed is defined by another number generally referred to as the mask. The mask lets one ignore some particular bits of the input while calculating the parity. In order to calculate the parity, the mask value is bitwise ANDed with the input value, the bits of the resultant is then taken and XORed together to obtain the parity*[4]

II. GENERATING LINEAR APPROXIMATION TABLE

In order to determine whether or the S-Box(s) is(are linear, the masked parity concept is used. Every possible combination of input and output mask has to be tested for all possible inputs. Basically, a random input is taken, it is then masked with an input mask and its parity is obtained (Input Parity). Next, the original input is run through the S-Box and masked with the output mask. Then its parity is computed (Output Parity). If they match, then that particular combination of input and output mask holds true for that input. After doing this for every possible input against every possible pair of input/output masks, a table called the **Linear Approximation Table (LAT)** is constructed. Each entry in the LAT is a number indicating the number of times a particular input/output mask pair holds true when tested against all possible inputs. For example, if a certain S-Box takes 4 bit inputs and produces 4 bit outputs, then the LAT will be of dimension 16 x 16 and each entry will range from 0 to 16, indicating the number of successful matches between input and output parity.

Algorithm 1: Algorithm for generating LAT.

```
For i=0 to 2m -1
    For j=0 to 2n -1
        For k=0 to 2m -1
            If Parity (k AND i) =Parity(S-Box[k] AND j) then
                LAT[i][j] ←LAT[i][j] +1
```

where, LAT is a 2-D array of size *m x n*. *Parity()* refers to a function that calaulates the parity of the given input. *m* is the total number of bits fed as input to the S-box. *n* is the total number of bits produced as output by the S-box. *i*

ranges from *0* to *2m -1*, that represents all possible input masks. *j* ranges from *0* to *2n-1* representing all possible output masks. *k* ranges from *0* to *2m -1*, that represents all possible inputs to S-box.

Let us assume an S-box that takes 4 bit inputs and produces 4 bit output, as shown in fig. 1. Both the input as well as the output ranges from 0 to 15. Such an S-Box is bijective in nature.

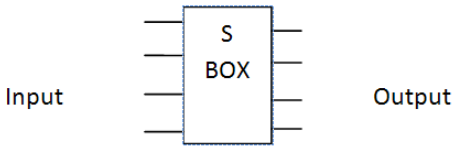

Fig 1: Substitution Box

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

Table1: Substitution Box

For such an S-Box, the algorithm to generate the LAT is modified as following:

Algorithm 2: Algorithm for generating LAT for the S-Box given in Table1

For i=0 to 15
  For j=0 to 15
    For k=0 to 15
      If Parity (k AND i) =Parity(S-Box[k] AND j) then
        LAT[i][j] ← LAT[i][j]+1

In this particular case, the generated LAT is of dimension 16 x 16. Table 2 depicts the LAT for the S-Box given in Table 1.

Similarly, the LAT for any of the DES S-Box can also be generated. For DES S-Box, the algorithm is modified as follows:

Algorithm 3: Algorithm for generating LAT for DES S-Box.

For i=0 to 15
  For j=0 to 63
    For k=0 to 15
      If Parity (k AND i) =Parity(S-Box[k] AND j) then
        LAT[i][j] ←AT[i][j]+1

Here, the LAT is of dimension 16 x 64, the reason being DES S-Box takes 4 bits as input and produces 6 bits as output.

III. PILING UP PRINCIPLE

The most fundamental tool used for linear cryptanalysis attack is the Piling Up Principle [1]. Let us consider two indicator random variables X1 and X2, and let us assume

$$P(X1 = i) = \begin{cases} p1, & i = 0 \\ 1 - p1, & i = 1 \end{cases}$$

and

$$P(X2 = i) = \begin{cases} p2, & i = 0 \\ 1 - p2, & i = 1 \end{cases}$$

Then, the probability of the relationship X1⊕X2 will be

$$P(X1 \oplus X2 = i) = \begin{cases} p1.p2 + (1-p1).(1-p2), & i = 0 \\ p1.(1-p2) + p2.(1-p1), & i = 1 \end{cases}$$

This means, X1 ⊕ X2 will be 0 when X1=X2 i.e. when either X1 as well as X2 are both 0 or X1 and X2 are both 1. And X1 ⊕ X2 will be 1 when X1≠ X2 i.e. when X1=0 and X2=1 or X1=1 and X2=0. Accordingly probabilities are computed, assuming X1 is independent of X2 and vice versa.

The deviation of the probability from 1/2 is of particular interest. Let us consider p1=1/2+ ε1 and p2=1/2+ε2, where ε1 and ε2 are the deviation of p1 and p2 respectively from 1/2 and are referred to as **probability bias**.

Now, $P(X1 \oplus X2=0)=(1/2 + \varepsilon1).(1/2+\varepsilon2) + (1-(1/2+\varepsilon1)).(1-(1/2+\varepsilon2)) =1/2+2.\varepsilon1.\varepsilon2$

So, probability bias of $X1 \oplus X2$ is given by $2.\varepsilon1.\varepsilon2$

Generally, if X1,X2,…Xn are n independent indicator random variables, then the probability of X1 ⊕ X2 ⊕ …⊕ Xn=0 is given by the **Piling Up Lemma**.

$$P(X1 \oplus X2 \oplus \ldots Xn = 0) = \tfrac{1}{2} + 2^{n-1} \cdot \prod_{i=1\ldots n} \varepsilon i \quad \ldots\ldots\ldots (1)$$

and the probability bias of X1⊕ X2 ⊕ …⊕ Xn=0 is given by

$$\varepsilon 1 \ldots n = 2^{n-1} \cdot \prod_{i=1\ldots n} \varepsilon i$$

Note that, P(X1 ⊕ X2 ⊕ …Xn =0) = 1/2 , if there exist some εi such that εi=0 or pi=1/2. And P(X1 ⊕ X2 ⊕ …Xn =0) = 0 or 1, if for all εi, εi=+1/2 or -1/2, respectively or pi=0 or 1, respectively.

		Output mask															
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
Input mask	**0**	16	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
	1	8	8	6	6	8	8	6	14	10	10	8	8	10	10	8	8
	2	8	8	6	6	8	8	6	6	8	8	10	10	8	8	2	10
	3	8	8	8	8	8	8	8	8	10	2	6	6	10	10	6	6
	4	8	10	8	6	6	4	6	8	8	6	8	10	10	4	10	8
	5	8	6	6	8	6	8	12	10	6	8	4	10	8	6	6	8
	6	8	10	6	12	10	8	8	10	8	6	10	12	6	8	8	6
	7	8	6	8	10	10	4	10	8	6	8	10	8	12	10	8	10
	8	8	8	8	8	8	8	8	8	6	10	10	6	10	6	6	2
	9	8	8	6	6	8	8	6	6	4	8	6	10	8	12	10	6
	A	8	12	6	10	4	8	10	6	10	10	8	8	10	10	8	8
	B	8	12	8	4	12	8	12	8	8	8	8	8	8	8	8	8
	C	8	6	12	6	6	8	10	8	10	8	10	12	8	10	8	6
	D	8	10	10	8	6	12	8	10	4	6	10	8	10	8	8	10
	E	8	10	10	8	6	4	8	10	6	8	8	6	4	10	6	8
	F	8	6	4	6	6	8	10	8	8	6	12	6	6	8	10	8

Table 2: Approximation Table generated for the S-Box given in fig. 1 using algorithm 2.

IV. ATTACKING A TOY CIPHER

Considering a toy cipher that takes 4 bit input goes through two iterations of key addition and block substitution and yields a 4 bit output. The following figure diagrammatically represents the toy cipher.

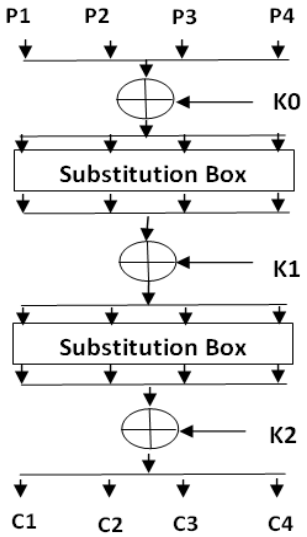

Fig 2: Toy Cipher

Where, P1, P2, P3, P4 together represent the 4 bit plain text C1, C2, C3, C4 represents 4 bit cipher text.

K0, K1, K2 are 4 bit subkeys.
Total key length is of 12 bits.
The cipher uses two identical S-Boxes, which is same as the S-Box described in Table 1.
The following algorithm implements the toy cipher:

Algorithm 4: Implementing Toy Cipher

```
Kye[] ←{k0,k1,k2}
S-Box[] ←{E,4,D,1,2,F,B,8,A,6,C,5,9,0,7}
For i=0 to 15       // 16 possible inputs
{     p=i
    For j= 0 to 1   // 2 iterations
          p ←S-Box [ p ⊕ Key[j]]
     C[i] ← p ⊕ Key[2] //final key whitening step
}
```

The output generated is listed in Table 3.

Plain Text	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Cipher Text	3	B	6	D	1	7	F	2	4	9	E	5	8	A	C	0

Table 3: Output of the Toy Cipher when Key[]={B,7,F}

The first step towards attacking the cipher begins by obtaining an equation of the form X1 ⊕ X2 ⊕…⊕ Xn = 0. Such an expression can be obtained using the generated LAT. In the example P(LAT[F][A])=12/16 or equivalently Bias(LAT[F][A})=4/16, where F is the input mask and A is the output mask. Although LAT[0][0]=16 it cannot be used.

Let Uij denote the j^{th} input bit of i^{th} S-Box and Vij denote the j^{th} output bit of the i^{th} S-Box.

So, P(U11 ⊕ U12 ⊕ U13 ⊕ U14 =V11 ⊕ V13)= 12/16

Let Kij denote the j^{th} bit of the i^{th} subkey, then U11 = P1 ⊕ K01, U12 =P2 ⊕ K02, U13 = P3 ⊕ K03, and U14 = P4 ⊕ K04, where Pi denotes the i^{th} plain text bit.

Therefore, P(P1 ⊕ K01 ⊕ P2 ⊕ K02 ⊕ P3 ⊕ K03 ⊕ P4 ⊕ K04 = V11 ⊕ V13)) = 12/16

or P (P1 ⊕ P2 ⊕ P3 ⊕ P4 ⊕ ∑K0 = V11 ⊕ V13) = 12/ 16

Since, U21 = V11 ⊕ K11 or, V11 = U21 ⊕ K11 and U23 = V23 ⊕ K13 or, V13 = U23 ⊕ K13

Hence, P (P1 ⊕ P2 ⊕ P3 ⊕ P4 ⊕∑K0 = U21 ⊕ K11 ⊕U23 ⊕ K13) = 12/ 16

or, P (P1 ⊕ P2 ⊕ P3 ⊕ P4 ⊕∑K0 ⊕ K11 ⊕ K13 = U21 ⊕U23) = 12/ 16

Let us assume K=∑K0 ⊕ K11 ⊕ K13, which can be either 0 or 1

Therefore, P (P1 ⊕ P2 ⊕ P3 ⊕ P4 ⊕ K= U21 ⊕U23) = 12/ 16

Or, P (P1⊕ P2⊕ P3 ⊕ P4 = U21⊕U23*)*

$$= \begin{cases} 12/16 & if\ K = 0 \\ 4/16 & if\ K = 1 \end{cases}$$

Now, as a linear expression with a high probability bias has been obtained, the ciphertext can be partially decrypted to obtain U2 (input to the 2^{nd} S-Box). The following algorithm does this.

Algorithm 5: Partially decrypting the ciphertext

C[] ←{ 3,B,6,D,1,7,F,2,4,9,E,5,8,A,C,0}

Isbox[] ←{E,3,4,6,1,C,A,F,7,D,9,6,B,2,0,5}

For k=0 to 15

{ pro[k]←0

For I = 0 to 15

{ pdc [k][i] ←isbox [C[i] ⊕ k]

If Parity (pdc[k][i] AND A) = Parity (I AND F) then

pro[k] ←pro[k] +1

}

}

It should be noted that *Parity (pdc[k][i] AND A) = Parity (I AND F*) is the algorithmic implementation of P1 ⊕ P2 ⊕ P3 ⊕ P4 = U21 ⊕ U23. Since, bitwise ANDing retrieves the required bits when ANDed with a mask having 1 in the required position in its binary equivalent.

The result obtained is presented in Table 4. From the result one can observe that probability when key=F is 12/16 which matches our expected probability, thereby indicating that **K2=F**.

It should be noted that, in this example, it so happened that there is only one candidate for K2, but generally there may be more than one candidates and all of them should be given due consideration.

For the next round, the partially decrypted cipher text is used with respect to **key=F** as the cipher text and the procedure defined as algorithm 5 is performed. That is, now C[]={B,1,D,4,0,7,E,2,6,A,3,9,F,C,8,5} The output yielded is listed in Table 5.

Key	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Probability	8/16	10/16	6/16	8/16	6/16	6/16	6/16	10/16	8/16	10/16	6/16	8/16	8/16	8/16	8/16	12/16

Table 4: Probabilities yielded by Algorithm 5.

Key	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Probability	12/16	12/16	8/16	4/16	12/16	12/16	4/16	0/16	8/16	8/16	8/16	8/16	8/16	8/16	8/16	8/16

Table 5: Probabilities yielded by Algorithm 5 using partially decrypted Ciphertext.

At this time the plaintext block P1, P2, P3, P4 or the input of the first S-Box i.e. U1, U2, U3, U4 are being computed, so the expected probability is computed as

P(P1 ⊕ P2 ⊕ P3 ⊕ P4 = P1 ⊕ P2 ⊕ P3 ⊕ P4) =1

Or,

P(P1 ⊕ P2 ⊕ P3 ⊕ P4 = P1 ⊕ P2 ⊕ P3 ⊕ P4 ⊕ ∑K0)

$$= \begin{cases} 1 & if\ \Sigma K0 = 0 \\ 0 & if\ \Sigma K0 = 1 \end{cases}$$

Or, P(P1 ⊕ P2 ⊕ P3 ⊕ P4 = P1 ⊕ K01 ⊕ P2 ⊕ K02 ⊕ P3 ⊕ K03 ⊕ P4 ⊕ K04)

$$= \begin{cases} 1 & if\ \Sigma K0 = 0 \\ 0 & if\ \Sigma K0 = 1 \end{cases}$$

Or, P(P1 ⊕ P2 ⊕ P3 ⊕ P4 = U11 ⊕ U12 ⊕ U13 ⊕ U14)

$$= \begin{cases} 1 & if\ \Sigma K0 = 0 \\ 0 & if\ \Sigma K0 = 1 \end{cases}$$

The expected probability matches with the observed probability for subkey **K1=7**. Therefore K1=7 with high degree of certainty.

So, the partially decrypted ciphertext for subkey=7 is retained, which is contained in pdc[7][i] for i=0 to 15. The partially cipher text for subkey=7 is given in Table 6.

Plain Text	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Partially decrypted Cipher Text	B	A	9	8	F	E	D	C	3	2	1	0	7	6	5	4

Table 6: partially cipher text for subkey =7

Now, in order to obtain the subkey K0, one simply needs to choose any pair of plaintext and partially decrypted ciphertext and perform a bitwise XOR operation. Say, one choose (4, F), then 4 ⊕ F = B, So, K0=B.
Thus, the actual key = {B, 7, F}, which is the key originally used in the example toy cipher.
It should be noted that, at every step of the attack, unique subkey values that match the expected probability are obtained, which may not be the case all the time, and in such situations where multiple subkeys match the expected probability, each of these subkeys has to be considered.

V. OBSERVATIONS

1. If LAT has one or more entries such that \Bias (LAT[i][j])| =1/2 (50%) and i=j (except for LAT[0][0]), then the S-Box can be broken by Linear Cryptanalysis attack. So, such an S-Box is a strict no for any cipher

2. If LAT has entries such that |Bias (LAT[i][j])| =1/2 and | Bias (LAT[j][k])| = 1/2 where i ≠ j ≠k and I,j,k≠0, then such a cipher can also be broken by Linear Cryptanalysis Attack.

3. If |Bias(LAT[i][j])| ≈1/2 where i≠j and there is no pair such that |Bias(LAT[i][j])|≈1/2 and |Bias(LAT[j][k])|≈1/2 where i ≠ j ≠k and I,j,k≠0, then after a certain number of iterations, Linear Cryptanalysis becomes ineffective. The observation is illustrated using the graph in fig.3.

VI. CONCLUSION

As the number of rounds of an iterative cipher increases and observations 1 and 2 does not hold, and then Linear Cryptanalysis becomes increasingly less effective. This is shown in fig 3.Ideally, the magnitude of the bias for each and every entry of LAT should be as low as possible (with the exception of LAT[0][0]).

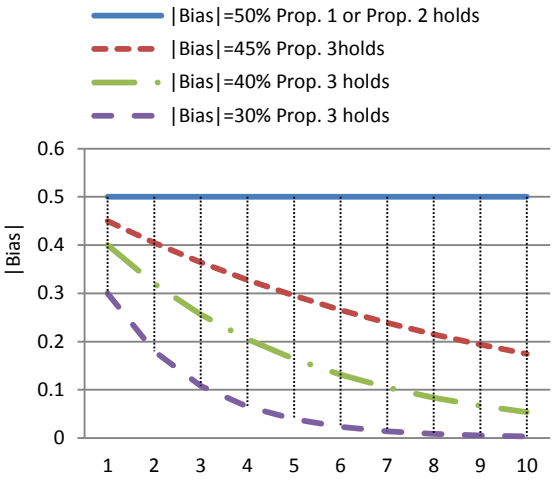

Fig 3: |Bias| Vs # iterations

REFERENCES

[1] Heys H M,"*A Tutorial on Linear And Differential Cryptanalysis*", Cryptologia, Vol. 25(3),pp189-221, 2002.

[2] Matsui M,"*Linear Cr4yptanalysis Method For DES Cipher*", Advance in Cryptlogy-EUROCRYPT'93, Springer, Verlag, 386-397, 1994.

[3] Jakobson B T, Abyar M., Nordholt P S, "*Linear And Differential Cryptanalysis*", pp. 234-242, 2006.

[4] Paar, C, Pelzl J, "*Understanding Cryptography"*, Berlin:Springer, Nerla, pp. 120-129, 2010.

[5] Bhowmik D, Datta A,Sinha S,"*Measuring the Diffusion Characteristic of Block Ciphers: The Bit Relationship Test (BRT)*",International Journal of Computer Science and Engineering,Vol.3(1), pp.76-80,Feb 2015.

[6] Sharma A., Thakur RS and Jaloree S., "Investigation of Efficient Cryptic Algorithm for Storing Video Files in Cloud", International Journal of Scientific Research in Computer Science and Engineering, Vol.4(6), pp.8-14, Dec 2016.

[7] Shah R. and Chouhan Y. S., "Encoding of Hindi Text Using Steganography Technique", International Journal of Scientific Research in Computer Science and Engineering, Vol.2(1), pp.22-28, Feb 2014.

[8] Bhowmick A., Kapur V. and Paladi S.T., "Concealing Cipher Data using an Amalgam of Image Steganography and two-level Image Cryptography", International Journal of Computer Sciences and Engineering, Vol.3(3), pp.7-12, Mar -2015.

AUTHORS PROFILE

Dipanjan Bhowmik is an UGC-SRF in the Department of Computer Science and Application, University of North Bengal. He received Master of Computer Application (MCA) degree in 2011 from University of North Bengal, WB, India. His research interest is Cryptology.

Avijit Datta is a Research Scholar in the Department of Computer Science and Application, University of North Bengal and Assistant Professor of Siliguri Institute of Technology, Siliguri. He received Master of Computer Application (MCA) degree in 2005 from UPTU, UP, India. His research interest is Cryptology.

Sharad Sinha is an Assistant Professor of University of North Bengal. He received Ph.D. degree in 2008 and Master of Computer Application (MCA) degree in 1992 from University of North Bengal, WB, India. His areas of interest are Cryptology.NLP,MIDI.

