Cryptography

Project II

Attacking Ciphertext with Linear Cryptanalysis

Name: Mümin Göker

Surname: Gayretli

Number: 514201018

1. Introduction

In this project linear cryptanalysis is applied to random generated ciphertexts and python is preferred as programming language. First of all, three SPNs in the project are designed according to project file. Then, a random master key is generated and four round keys are produced by using master key and equations in the project file. After that, random ciphertexts are obtained by using random generated plaintexts. At the end, candidate key value table is obtained. Rows of this table are summed and absolute value of final array is examined. Candidate key with maximum absolute value will give actual key value. Finding master key and other round keys is enough for decryption so I do not convert ciphertext to plaintext.

BLOCK CIPHER 1

2. Design of SPNs and Pair Generation

When s-boxes and permutation tables are designed for first block cipher, plaintext-ciphertext pair and key groups is are generated for linear cryptanalysis. Sample key ciphertext-plaintext pair and key group are given in the table 1 and 2 respectively. Codes of first block cipher design and random plaintext generation are given in appendix.

Plaintext ($P_1P_2....P_{16}$)	**Ciphertext ($C_1C_2....C_{16}$)**
1010000111101101	1000100101011101
0000101001010011	1111001010001010
0101011011110000	0101110111100101
0010100100100110	1100011001010010
1000000000001010	1010100000101000

Table 1: Random Generated Plaintext-Ciphertext Pairs

Master Key	**1111110010111000**
First Round Key	0111111001011100
Second Round Key	0011111100101110
Third Round Key	0111111001011100
Fourth Round Key	1000111111001011

Table 2: Key Generation Algorithm Results

3. Linear Approximation Table Generation

Goal of linear cryptanalysis is expressing SPN network by using linear equation. In order to obtain this equation, linear approximation table should be constituted. This operation is done by using [1] and linear approximation table for first block cipher is shown in the table 3.

		OUTPUT															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
INPUT	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	2	0	-2	-2	0	2	0	4	-2	0	-2	-2	0	-2	-4
	2	0	-2	0	2	2	-4	-2	-4	2	0	-2	0	0	2	0	-2
	3	0	0	0	0	0	4	0	-4	2	2	2	2	2	-2	2	-2
	4	0	2	0	-6	0	-2	0	-2	-2	0	-2	0	2	0	2	0
	5	0	0	-4	0	2	-2	2	2	2	2	2	-2	4	0	0	0
	6	0	0	4	0	2	2	2	-2	0	0	0	-4	2	2	-2	2
	7	0	-2	0	-2	4	2	-4	2	0	-2	0	-2	0	-2	0	-2
	8	0	-6	0	-2	-2	0	2	0	2	0	-2	0	0	-2	0	2
	9	0	0	0	0	0	0	0	0	2	-2	2	-2	-2	2	6	2
	A	0	0	0	0	4	0	4	0	0	-4	0	4	0	0	0	0
	B	0	-2	0	2	-2	0	2	0	-4	-2	0	-2	2	0	2	-4
	C	0	0	4	0	2	-2	2	2	0	4	0	0	-2	-2	2	-2
	D	0	2	0	2	0	-2	0	-2	0	-2	0	-2	0	-6	0	2
	E	0	-2	0	-2	0	-2	0	-2	-2	0	6	0	-2	0	-2	0
	F	0	0	4	0	-2	-2	-2	2	2	-2	2	2	4	0	0	0

Table 3: Linear Approximation Table for First Block Cipher

4. Constructing Linear Equation for First SPN

While creating linear equations, the input bits should be used as much as possible. However, active s-box number should be minimized. By using this information, following steps are tracked.

1- $P_1\ P_2\ P_3\ P_4$ bits are selected for the first step. As seen in the table three, output two($\mathbf{V_{1,3}}$) is chosen in order to maximized absolute value of bias.
   - **a.** $\mathbf{X_1 \oplus X_2 \oplus X_3 \oplus X_4 = Y_2}$**; Bias = 4/16 = 1/4.**
   - **b.** $\mathbf{V_{1,3} = U_{1,1} \oplus U_{1,2} \oplus U_{1,3} \oplus U_{1,4}}$
   - **c.** $\mathbf{U_{1,1} = P_1 \oplus K_{1,1}}$
   - **d.** $\mathbf{V_{1,3} = P_1 \oplus K_{1,1} \oplus P_2 \oplus K_{1,2} \oplus P_3 \oplus K_{1,3} \oplus P_4 \oplus K_{1,4}}$

2- After that step, permutation section directs $\mathbf{V_{1,3}}$ to $\mathbf{U_{2,9}}$. Output one($\mathbf{V_{2,12}}$) is selected according to input eight($\mathbf{U_{2,9}}$) and linear approximation table.
   - **a.** $\mathbf{X_1 = Y_4}$**; Bias = -6/16 = -3/8.**
   - **b.** $\mathbf{V_{2,12} = U_{2,9}}$
   - **c.** $\mathbf{U_{2,9} = V_{1,3} \oplus K_{2,9}}$

3- Permutation section directs $\mathbf{V_{2,12}}$ to $\mathbf{U_{3,15}}$. Output eight($\mathbf{V_{3,13}}$) is selected according to input two($\mathbf{U_{3,15}}$) and linear approximation table.
   - **a.** $\mathbf{X_3 = Y_1}$**; Bias = 4/16 = 1/4.**
   - **b.** $\mathbf{V_{3,13} = U_{3,15}}$
   - **c.** $\mathbf{U_{3,15} = V_{2,12} \oplus K_{3,15}}$

4- At the last step linear equation of SPN network is obtained.
   - a. Permutation layer directs $\mathbf{V_{3,13}}$ to $\mathbf{U_{4,4}}$. The equation $\mathbf{U_{4,4} = V_{3,13} \oplus K_{4,4}}$ is obtained at the end**.** After this step $\mathbf{V_{3,13}}$ is substituted iteratively by using blue equations in steps 1-3. Orange equation represents linear approximation of first SPN network.
   - b. $\mathbf{U_{4,4} = P_1 \oplus P_2 \oplus P_3 \oplus P_4 \oplus K_{1,1} \oplus K_{1,2} \oplus K_{1,3} \oplus K_{1,4} \oplus K_{2,9} \oplus K_{3,15}}$.

5- Bias value of network is calculated by using pilling up lemma => $\mathbf{2^2 * 1/4 * -3/8 * 1/4 = -3/32}$.

6- XOR operation result of key bits can be 1 or 0. We assume that they are zero and final equation will be
   - a. $\mathbf{U_{4,4} = P_1 \oplus P_2 \oplus P_3 \oplus P_4}$**. Construction path is visualized in the figure 1.**

7- After this steps candidate key value table is constructed by using random generated plaintext and ciphertext pairs. There are 16 possibilities for $K_{5,1}\ K_{5,2}\ K_{5,3}\ K_{5,4}$ from 0 to 15. We compute blue lines in figure 1 by XOR operation of a ciphertexts and each key possibility. After that, $\mathbf{U_{4,1}\ U_{4,2}\ U_{4,3}\ U_{4,4}}$ is computed by using inverse of s-box. Now we have 16 possible $\mathbf{U_{4,4}}$.

8- $\mathbf{U_{4,4}}$ and first four bits of plaintext ($\mathbf{P_1\ P_2\ P_3\ P_4}$) are subjected to XOR operation. If $\mathbf{U_{4,4} \oplus P_1 \oplus P_2 \oplus P_3 \oplus P_4}$ operation is equal to zero, +1 will be added to candidate key table since we assume that XOR operation result of key bits is equal to zero in step 6.

9- This operation is performed for all pair. Then rows of candidate table are summed iteratively and an array which shows the deviation amount for key possibility is obtained. The possible key whose absolute deviation value is highest corresponds to actual key. This value is pointed with blue in table 4 which is result array for first block cipher.

10- **We can assume that $K_{5,1}\ K_{5,2}\ K_{5,3}\ K_{5,4}$ = 1000 from table 4.**

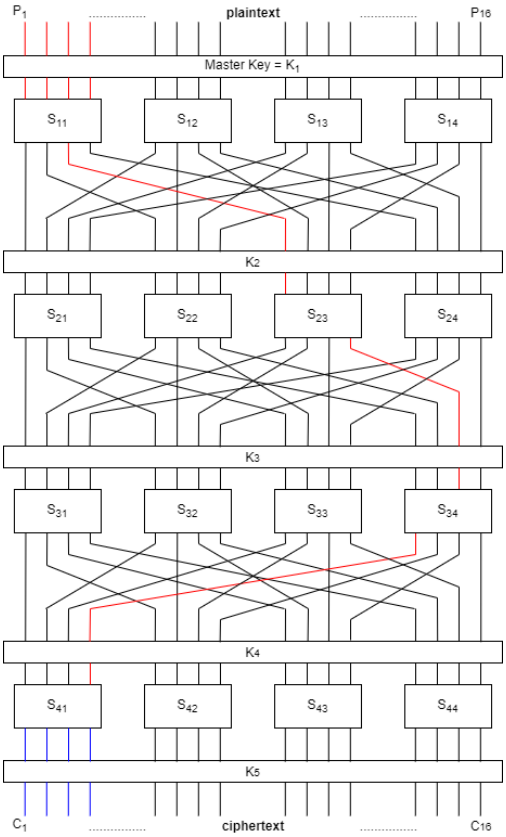

Figure 1: Visualization of Construction Path for First Four bits of First Block Cipher

Possible Key (K5,1 K5,2 K5,3 K5,4)	0	1	2	3	4	5	6	7
Deviation	-1090	-56	352	-1314	220	-754	-530	-188
Possible Key (K5,1 K5,2 K5,3 K5,4)	8	9	10	11	12	13	14	15
Deviation	1718	-20	308	762	-144	126	1082	-472

Table 4: Result Array for $K_{5,1}$ $K_{5,2}$ $K_{5,3}$ $K_{5,4}$ for First Block Cipher

5. Key Derivation

Linear cryptanalysis will be applied to other 12 bits of last round key. I do not show intermediate steps for ($K_{5,5}$ $K_{5,6}$ ….. $K_{5,16}$), but visualization of construction path, bias value and result array table are given for each four bits of last round key. Then, key derivation is performed in order to find other round keys and master key.

a. Finding $K_{5,5}$ $K_{5,6}$ $K_{5,7}$ $K_{5,8}$

- The linear equation is $U_{4,8} = P_1 \oplus P_2 \oplus P_3 \oplus P_4$ for $K_{5,5}$ $K_{5,6}$ $K_{5,7}$ $K_{5,8}$. This can be understood easily from figure 2.
- Bias value is $2^2 * 1/4(S_{11}) * -3/8(S_{23}) * 1/8(S_{34}) = -3/64$.
- Construction path and result array can be shown in figure 2 and table 5 respectively.
- **We can assume that $K_{5,5}$ $K_{5,6}$ $K_{5,7}$ $K_{5,8}$ = 1111 by using table 5.**

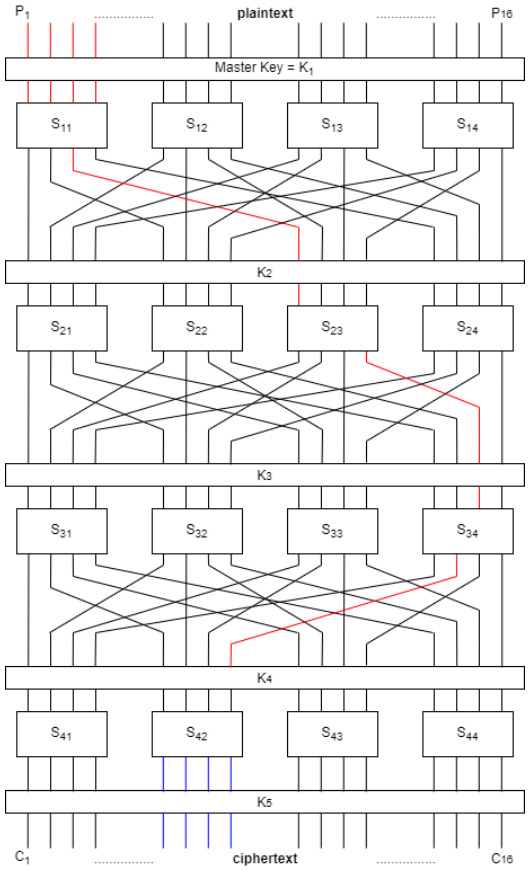

Figure 2: Visualization of Construction Path for Second Four bits of First Block Cipher

Possible Key (K5,5 K5,6 K5,7 K5,8)	0	1	2	3	4	5	6	7
Deviation	142	-346	-488	142	-670	78	78	-528
Possible Key (K5,5 K5,6 K5,7 K5,8)	8	9	10	11	12	13	14	15
Deviation	-94	648	-154	-78	368	-126	-142	1170

Table 5: Result Array for $K_{5,5}$ $K_{5,6}$ $K_{5,7}$ $K_{5,8}$ for First Block Cipher

b. Finding $K_{5,9}\,K_{5,10}\,K_{5,11}\,K_{5,12}\,K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16}$

- The linear equation of this step is $U_{4,12} \oplus U_{4,16} = P_1 \oplus P_2 \oplus P_3 \oplus P_4$. Two s-boxes are activated at the last step. Thus, I must try to guess $K_{5,9}\,K_{5,10}\,K_{5,11}\,K_{5,12}\,K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16}$. Since there are 256 possible keys, it is hard to show all possible key result. Thus, most frequent 16 key assumption will be shown in table 6.
- Bias value is $2^2 * 1/4(S_{11}) * -3/8(S_{23}) * -1/8(S_{34}) = 3/64$.
- Construction path and result array can be shown in figure 3 and table 6 respectively.
- **We can assume that $K_{5,9}\,K_{5,10}\,K_{5,11}\,K_{5,12}\,K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16} = 11001011$ by using table 6.**

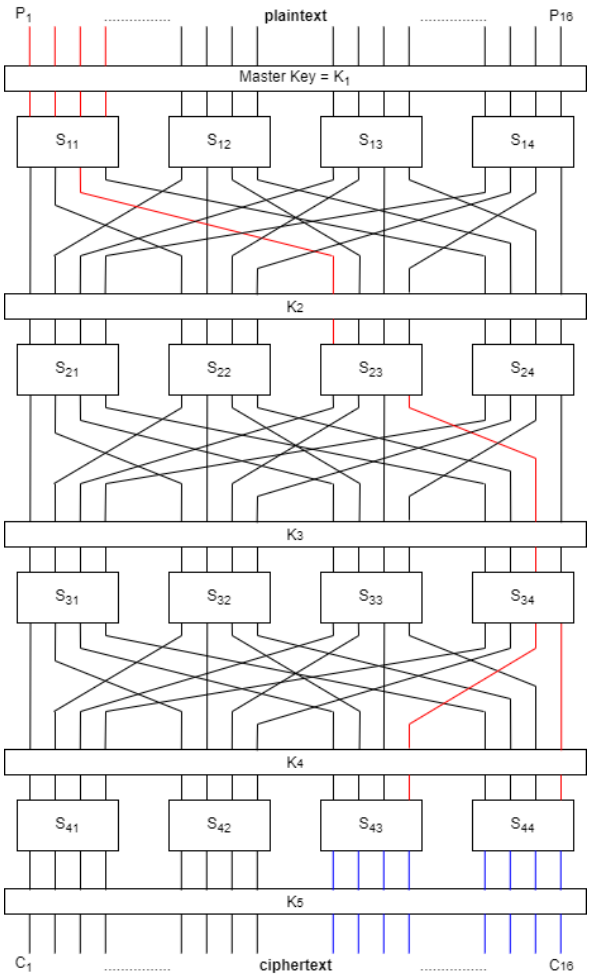

Figure 3: Visualization of Construction Path for Last Eight bits of First Block Cipher

Possible Key (K5,9 .... K5,16)	203	192	123	171	75	27	251	205
Deviation	1936	1084	1082	1040	1030	978	968	940
Possible Key (K5,9 .... K5,16)	43	195	198	200	16	168	160	197
Deviation	926	880	846	786	690	662	660	642

Table 6: Result Array for $K_{5,9}\,K_{5,10}\,K_{5,11}\,K_{5,12}\,K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16}$ for First Block Cipher

c. Finding $K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16}$

- Although an assumption is made in previous section for $K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16}$, I want to make another assumption in order to make my guess stronger. The linear equation is $U_{4,16} = P_1 \oplus P_2 \oplus P_3 \oplus P_4$ for $K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16}$. This can be understood easily from figure 4.
- Bias value is $2^2 * 1/4(S_{11}) * -3/8(S_{23}) * -1/8(S_{34}) = 3/64$.
- Construction path and result array can be shown in figure 4 and table 7 respectively.
- **We can assume that $K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16} = 1011$ by using table 7. This assumption matches with previous step.**

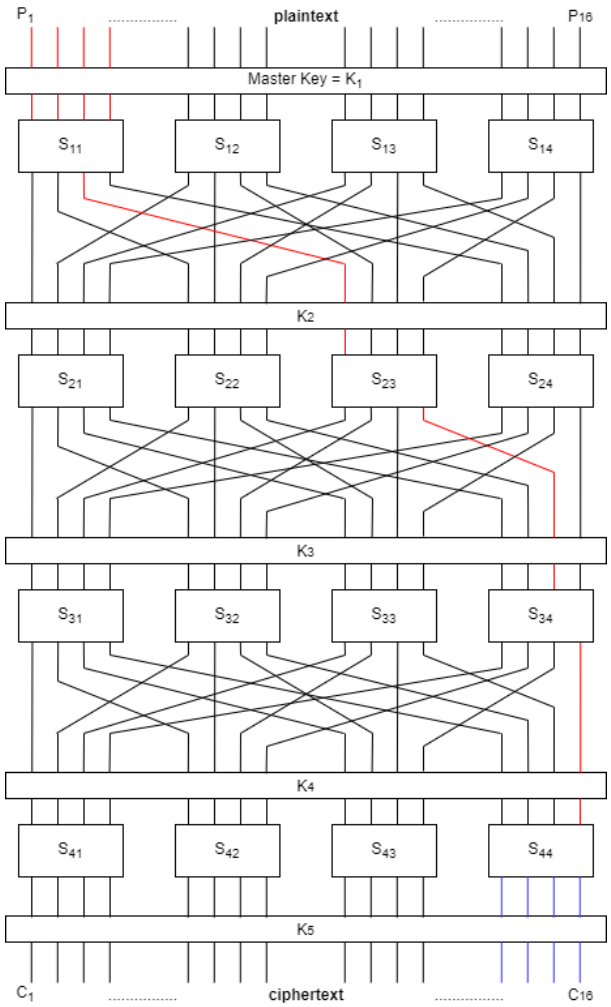

Figure 4: Visualization of Construction Path for Last Four Bits of First Block Cipher

Possible Key (K5,13 K5,14 K5,15 K5,16)	0	1	2	3	4	5	6	7
Deviation	-836	-392	-182	-568	98	-1690	-1958	-112
Possible Key (K5,13 K5,14 K5,15 K5,16)	8	9	10	11	12	13	14	15
Deviation	1974	-8	88	2364	302	552	162	206

Table 7: Result Array for $K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16}$ for First Block Cipher

- If we concatenate four assumptions, last round key assumption will be;
    1. **$K_{5,1}\,K_{5,2}$ …. $K_{5,15}\,K_{5,16}$ = 1000111111001011.**
- After this step we can obtain other round keys and master key. This operation is performed in four steps;
    1. $\mathbf{K_{i+1,15}} \oplus \mathbf{K_{i+1,4}} \oplus \mathbf{K_{i+1,3}} \oplus \mathbf{K_{i+1,1}}$ value assigned a variable(X)
    2. Shift current key one bit left.
    3. Assign $\mathbf{K_{i,16}} = \mathbf{0}$
    4. Assign variable X to $\mathbf{K_{i,0}}$.
- Codes for this operation is given in the appendix. After the key derivation codes run, we can say that our assumption is true for obtaining all round and master keys by looking table 8.

Master Key	1111110010111000
First Round Key	0111111001011100
Second Round Key	0011111100101110
Third Round Key	0111111001011100
Fourth Round Key	1000111111001011

Table 8: Key Derivation Algorithm Results

BLOCK CIPHER 2

- Linear cryptanalysis of second block cipher resembles to first one. Only difference is linear equations because of s-boxes. As in the previous sections, I do not show intermediate steps, but visualization of construction path, bias value and result array table are given for each four bits of last round key.

a. Finding $K_{5,1}\,K_{5,2}\,K_{5,3}\,K_{5,4}$

- The linear equation is $U_{4,1} = P_1 \oplus P_2 \oplus P_3 \oplus P_4$ for $K_{5,1}\,K_{5,2}\,K_{5,3}\,K_{5,4}$. This can be understood easily from figure 5.
- Bias value is $2^2 * 1/4(S_{11}) * 1/4(S_{24}) * -1/4(S_{31}) = -1/16$.
- Construction path and result array can be shown in figure 5 and table 9 respectively.
- **We can assume that $K_{5,1}\,K_{5,2}\,K_{5,3}\,K_{5,4} = 1000$ by using table 9.**

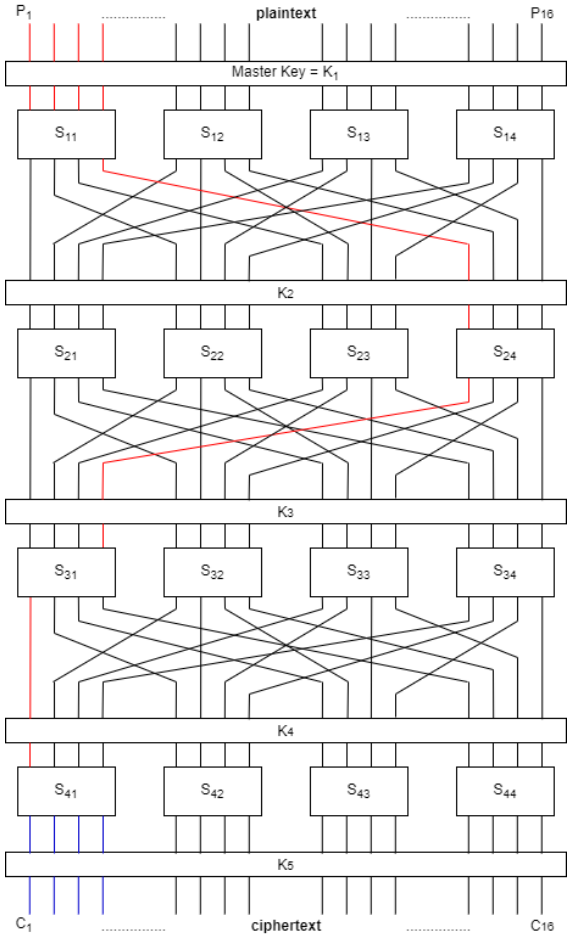

Figure 5: Visualization of Construction Path for First Four bits of Second Block Cipher

Possible Key (K5,1 K5,2 K5,3 K5,4)	0	1	2	3	4	5	6	7
Deviation	410	198	-884	508	-578	282	104	424
Possible Key (K5,1 K5,2 K5,3 K5,4)	8	9	10	11	12	13	14	15
Deviation	-1258	618	552	-144	-228	212	-478	262

Table 9: Result Array for $K_{5,1}\,K_{5,2}\,K_{5,3}\,K_{5,4}$ for Second Block Cipher

b. Finding $K_{5,5}\,K_{5,6}\,K_{5,7}\,K_{5,8}$

- The linear equation is $U_{4,5} = P_1 \oplus P_2 \oplus P_3 \oplus P_4$ for $K_{5,5}\,K_{5,6}\,K_{5,7}\,K_{5,8}$. This can be understood easily from figure 6.
- Bias value is $2^2 * 1/4(S_{11}) * 1/4(S_{24}) * 1/8(S_{31}) = 1/32.$
- Construction path and result array can be shown in figure 6 and table 10 respectively.
- **We can assume that $K_{5,5}\,K_{5,6}\,K_{5,7}\,K_{5,8}$ = 0011 by using table 10.**

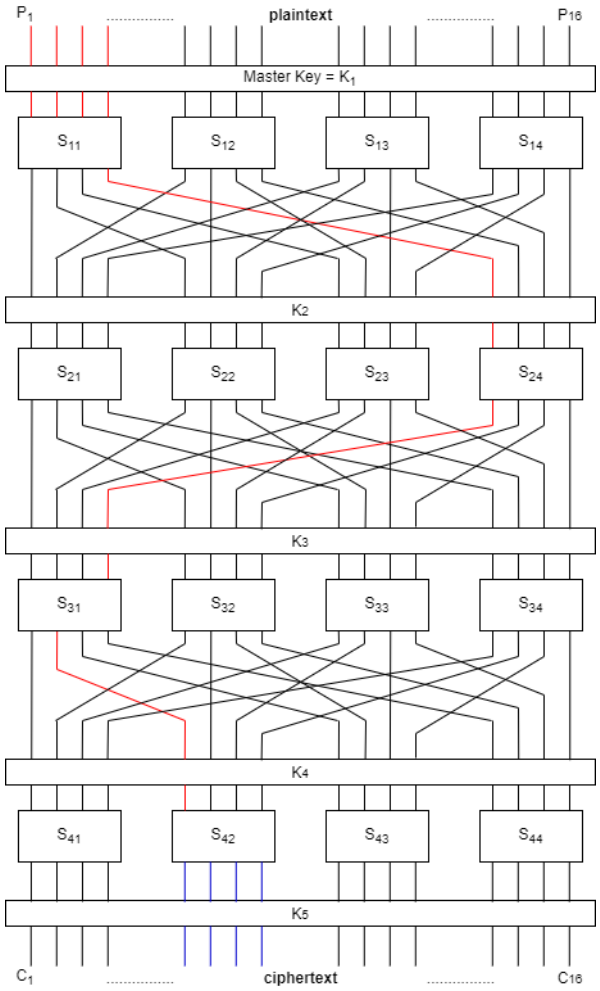

Figure 6: Visualization of Construction Path for Second Four bits of Second Block Cipher

Possible Key (K5,5 K5,6 K5,7 K5,8)	0	1	2	3	4	5	6	7
Deviation	-30	68	122	832	-42	436	134	464
Possible Key (K5,5 K5,6 K5,7 K5,8)	8	9	10	11	12	13	14	15
Deviation	-346	-494	-554	402	-318	126	-582	-218

Table 10: Result Array for $K_{5,5}\,K_{5,6}\,K_{5,7}\,K_{5,8}$ for Second Block Cipher

c. Finding $K_{5,9}\,K_{5,10}\,K_{5,11}\,K_{5,12}\,K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16}$

- The linear equation of this step is $U_{4,9} \oplus U_{4,13} = P_1 \oplus P_2 \oplus P_3 \oplus P_4$. Two s-boxes are activated at the last step. Thus, I must try to guess $K_{5,9}\,K_{5,10}\,K_{5,11}\,K_{5,12}\,K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16}$. Since there are 256 possible keys, it is hard to show all possible key result. Thus, most frequent 16 key assumption will be shown in table 11.
- Bias value is $2^2 * 1/4(S_{11}) * 1/4(S_{24}) * 1/8(S_{31}) = 1/32$.
- Construction path and result array can be shown in figure 7 and table 11 respectively.
- **We can assume that $K_{5,9}\,K_{5,10}\,K_{5,11}\,K_{5,12}\,K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16} = 11110110$ by using table 11.**

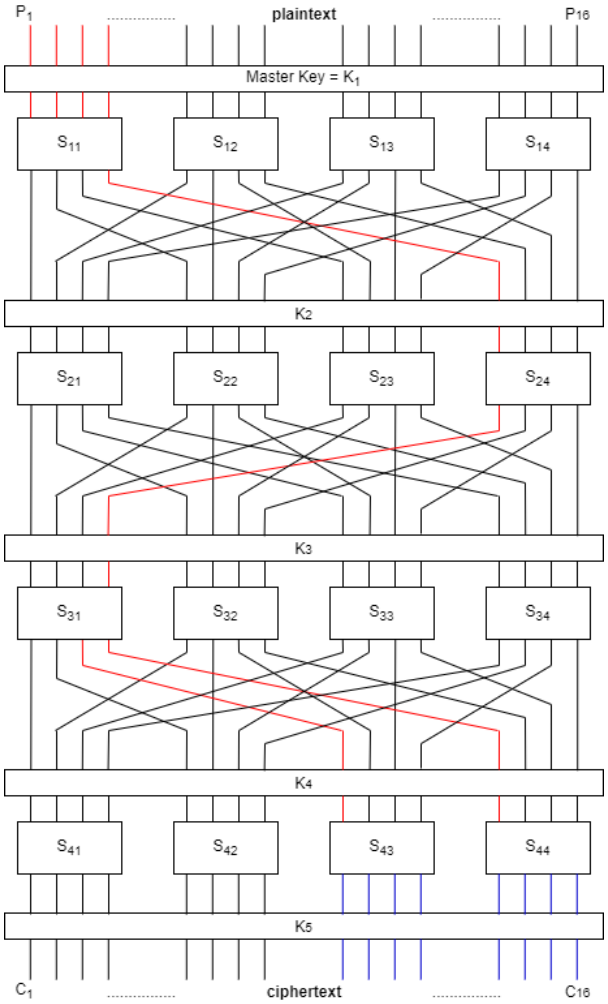

Figure 7: Visualization of Construction Path for Last Eight bits of Second Block Cipher

Possible Key (K5,9 .... K5,16)	246	242	102	255	92	249	38	150
Deviation	1342	850	848	834	816	754	740	734
Possible Key (K5,9 .... K5,16)	60	251	90	182	253	6	240	111
Deviation	706	676	656	598	596	592	580	572

Table 11: Result Array for $K_{5,9}\,K_{5,10}\,K_{5,11}\,K_{5,12}\,K_{5,13}\,K_{5,14}\,K_{5,15}\,K_{5,16}$ for Second Block Cipher

d. Finding $K_{5,13}$ $K_{5,14}$ $K_{5,15}$ $K_{5,16}$

- Although an assumption is made in previous section for $K_{5,13}$ $K_{5,14}$ $K_{5,15}$ $K_{5,16}$, I want to make another assumption in order to make my guess stronger. The linear equation is $U_{4,13} = P_1 \oplus P_2 \oplus P_3 \oplus P_4$ for $K_{5,13}$ $K_{5,14}$ $K_{5,15}$ $K_{5,16}$. This can be understood easily from figure 8.
- Bias value is $2^2 * 1/4(S_{11}) * 1/4(S_{24}) * -1/8(S_{31}) = -1/32$.
- Construction path and result array can be shown in figure 8 and table 12 respectively.
- **We can assume that $K_{5,13}$ $K_{5,14}$ $K_{5,15}$ $K_{5,16}$ = 0110 by using table 12. This assumption matches with previous step.**

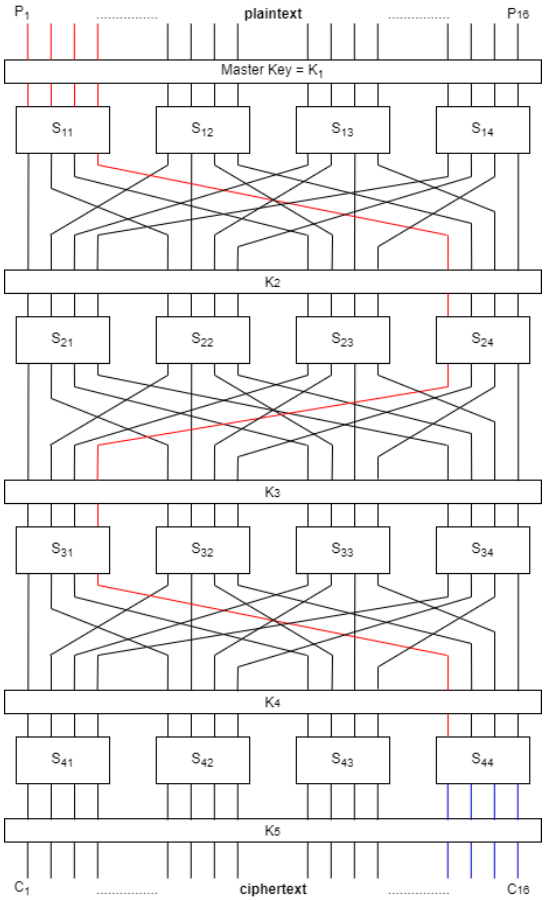

Figure 8: Visualization of Construction Path for Last Four Bits of Second Block Cipher

Possible Key (K5,13 K5,14 K5,15 K5,16)	0	1	2	3	4	5	6	7
Deviation	1090	-1106	336	-24	-86	-474	1512	-656
Possible Key (K5,13 K5,14 K5,15 K5,16)	8	9	10	11	12	13	14	15
Deviation	568	-928	562	-498	990	-902	140	-524

Table 12: Result Array for $K_{5,13}$ $K_{5,14}$ $K_{5,15}$ $K_{5,16}$ for Second Block Cipher

- I do not perform key derivation algorithm for this part because last round key assumption and the actual key match.

BLOCK CIPHER 3

- In this part, nearly same code is used with block cipher 2. The only difference is permutation part is removed from algorithm. Thus, it enables us to construct different linear equations whose bias are high. Since it is proved that we can derive master and other round keys, it is sufficent to perform only the first four bits of ciphertext cryptanalysis.
- I expect that this block cipher cryptanalysis will be more successful than the first two ones. However, it is observed that algorithm predict the actual key less consistently.

a. Finding $K_{5,1}\,K_{5,2}\,K_{5,3}\,K_{5,4}$

- The linear equation is $U_{4,1} \oplus U_{4,3} = P_1 \oplus P_2 \oplus P_3 \oplus P_4$ for $K_{5,1}\,K_{5,2}\,K_{5,3}\,K_{5,4}$. This can be understood easily from figure 9.
- Bias value is $2^2 * 1/4(S_{11}) * 1/4(S_{21}) * -1/4(S_{31}) = -1/16$.
- Construction path and result array can be shown in figure 9 and table 13 respectively.
- **We can assume that $K_{5,1}\,K_{5,2}\,K_{5,3}\,K_{5,4}$ = 0111 by using table 13.**

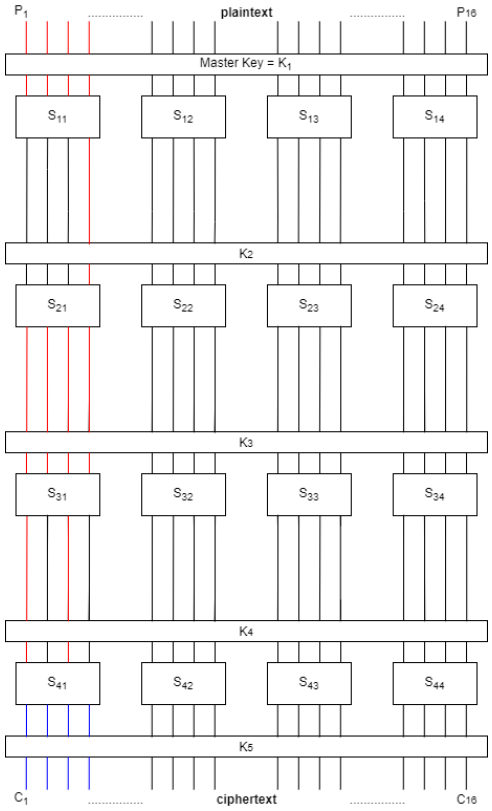

Figure 8: Visualization of Construction Path for Last Four Bits of Second Block Cipher

Possible Key (K5,1 K5,2 K5,3 K5,4)	0	1	2	3	4	5	6	7
Deviation	62	104	12	-144	6	-92	-74	154
Possible Key (K5,1 K5,2 K5,3 K5,4)	8	9	10	11	12	13	14	15
Deviation	-2	-22	-72	62	64	12	4	-74

Table 13: Result Array for $K_{5,1}\,K_{5,2}\,K_{5,3}\,K_{5,4}$ for Third Block Cipher

Comparison&Results

- First two block cipher perform substitution and permutation layer consecutively so they invoke many s-boxes and increase the active s-box in cryptanalysis. This leads decrement in bias and less successful prediction. However, third block cipher is not SPN network in practical because it does not have permutation layer. Since it cannot shuffle substitution outputs, attacker can easily constitute many linear equations with high bias.
- If these systems design in software instead of hardware, third block cipher may have some advantageous over first two one. Since our system is breakable, execution time of encryption is crucial. For instance, Alice sends a secret message to Bob. When Eve gets the secret message, it must decrypt it until Alice sends a new secret message. If encryption execution time becomes longer and longer, probability of Eve success will increase. I measure the execution time of decryption for 10000 samples and third block cipher gives much better solution than the first two.

Block Cipher	Encryption Execution Time for SW (second)
1	4.01
2	4.07
3	0.73

Table 14: Encryption Execution Time of Block Ciphers

- I observed that inverse of the s-box is same with original for block cipher 2 and 3. Thus, this saves the space in HW when secure communication system designs. Also, block cipher 3 can encrypt and decrypt 16 bits plaintext and ciphertext by using 4 bits network because it does not have permutation layer. Thus, it can save some space in HW.
- Existence of permutation layer may lead to activate more than one s-box in the end. If two s-boxes are activated in the end, candidate key table becomes larger and cryptanalysis system demands more memory and time. Thus, we can easily say that first two block cipher more secure than the last one.
- Another parameter of secure system is bias of linear equations. The higher bias value means lower secured system. Absolute value of first two block cipher biases are smaller than third one. This also shows that first two system is more secure. Average bias values for three block cipher are given in the table 15.

Block Cipher	Average Bias Values
1	15/256
2	10/256
3	16/256

Table 15: Average Bias Values of Block Ciphers

References

1- "Calculate Linear Approximation Tables for Sboxes." Gist. Accessed May 9, 2021. https://gist.github.com/mimoo/b2387d45fbbfcbb30553d791ed4b0ff7.

Appendix

```
from bitstring import BitStream, BitArray
import time

def reverse_Bits(n, no_of_bits):
    result = 0
    for i in range(no_of_bits):
        result <<= 1
        result |= n & 1
        n >>= 1
    return result

def bitArrToInt(value):
    out = 0
    for bit in value:
        out = (out << 1) | bit
    return out

def permutation(value):
    a = BitArray(uint=value, length=16)
    tempValue = a * 1
    for x in range(0,16):
        tempValue[x] = a[ptab_bc_1[x]]
    ret = bitArrToInt(tempValue)
    return ret

def get_bit(value, n):
    return ((value >> n & 1) != 0)

def set_bit(value, n):
    return value | (1 << n)

def clear_bit(value, bit):
    return value & ~(1<<bit)

def keyUpdate(value):
    tempValue = value >> 1
    last_bit = ((value%2) ^ ((value & (pow(2,2)))>>2) ^ ((value & (pow(2,4)))>>4) ^ ((value & (pow(2,5)))>>5))
    if(last_bit == 1):
        tempValue = set_bit(tempValue,15)
    return tempValue

def reverseBit(value):
    a0 = get_bit(value,0)
    a1 = get_bit(value,1)
    a2 = get_bit(value,2)
    a3 = get_bit(value,3)
    return (a0<<3 | a1<<2 | a2<<1 | a3)

def ciphterTextGenerator(plainText):
    roundInputs[0] = plainText
    for i in range(0,4):
        if i == 0:
            reversedInp = reverse_Bits(roundInputs[i],16)
        else:
            reversedInp = roundInputs[i]
        reversedKey = reverse_Bits(roundKeys[i],16)
```

```
      xorResult = reversedKey ^ reversedInp
      subsResult = sbox_bc_1[0x0f & xorResult] | (sbox_bc_1[((0x0f << 4) & xorResult) >> 4] << 4) | (sbox_bc_1[((0x0f << 8) &
xorResult) >> 8] << 8)  | (sbox_bc_1[((0x0f << 12) & xorResult) >> 12] << 12)
      permResult = permutation(subsResult)
      if i == 3:
         reversedKey = reverse_Bits(roundKeys[i],16)
         xorResult = reversedKey ^ roundInputs[i]
         subsResult = sbox_bc_1[0x0f & xorResult] | (sbox_bc_1[((0x0f << 4) & xorResult) >> 4] << 4) | (sbox_bc_1[((0x0f << 8)
& xorResult) >> 8] << 8)  | (sbox_bc_1[((0x0f << 12) & xorResult) >> 12] << 12)
         reversedKey2 = reverse_Bits(roundKeys[i+1],16)
         cipherTextRet = reversedKey2 ^ subsResult
         return cipherTextRet
      roundInputs[i+1] = permResult

np.savetxt('part1.csv', linearApproxTable, delimiter=',', fmt='%d')

NUMBER_OF_PAIR = 10000

plainText = np.zeros((NUMBER_OF_PAIR),dtype = int)
cipherText = np.zeros((NUMBER_OF_PAIR),dtype = int)
masterKey = randint(0,65535)

roundKeys = np.zeros((5),dtype = int)
roundInputs = np.zeros((5),dtype = int)

roundKeys[0] = masterKey

print("rkey ",0," =",format(roundKeys[0], '0{}b'.format(16)))
for i in range(0,4):
   roundKeys[i+1] = keyUpdate(roundKeys[i])
   print("rkey ",i+1," =",format(roundKeys[i+1], '0{}b'.format(16)))

start_time = time.time()

for j in range(0,NUMBER_OF_PAIR):
   plainText[j] = randint(0,65535)
   cipherText[j] = ciphterTextGenerator(plainText[j])

print("--- %s seconds ---" % (time.time() - start_time))
```

Table 16: Generating Random Ciphertexts and Round Keys for First Block Cipher

```
def getFourBits(value,index):
   return ((((0xf)<<(4*index)) & value)>>(4*index))

def plainTextBitOperation(value,index):
   temp = value * 1
   temp = temp & ((0xf)<<(index*4));
   temp = temp >> (index*4)
   return (get_bit(temp,0)) ^ (get_bit(temp,1)) ^ (get_bit(temp,2)) ^ (get_bit(temp,3))



def candidateGenrator_K1_4(plainTextFunc,cipherTextFunc):
   canditateArrayResult = []
   for j in range(0,16):
      uBit = sbox_inv_bc_1[getFourBits(cipherTextFunc,3) ^ j]
      result = plainTextBitOperation(plainTextFunc,0) ^ get_bit(uBit,0)
      if result == 0:
```

```
            result = -1
        canditateArrayResult.append(result)
    return canditateArrayResult

resArr = np.empty((0,16), int)

print(resArr)

for i in range(0,NUMBER_OF_PAIR):
    candRslt = candidateGenrator_K1_4(plainText[i],cipherText[i])
    resArr = np.append(resArr, [candRslt], axis=0)

b = np.sum(resArr,axis=0)
print(b)

np.savetxt('part1_K1_4.csv', b, delimiter=',', fmt='%d')

b = abs(b)
argSort = np.argsort(-1*b)

print("d=", format(plainText[i], '0{}b'.format(16)))
print("w=", format(reverse_Bits(roundKeys[4],16), '0{}b'.format(16)))

z = getFourBits(reverse_Bits(roundKeys[4],16),3)
print("ackey=",z,"\tbyte=",format(z, '0{}b'.format(4)))

for element in argSort:
    print("index=",element,"\tvalue=",b[element])
```

Table 17: Linear Cryptanalysis of First Block Cipher

```
def keyDerivation(value):
    tempValue = value << 1
    tempValue = clear_bit(tempValue,16)
    last_bit = get_bit(value,1) ^ get_bit(value,3) ^ get_bit(value,4) ^ get_bit(value,15)
    if(last_bit == 1):
        tempValue = set_bit(tempValue,0)

    return tempValue

for i in range(4, 0, -1):
    derivedKey = keyDerivation(roundKeys[i])
    if i != 1:
        print("round  key",i,"=",format(derivedKey, '0{}b'.format(16)))
    else:
        print("master key "," =",format(derivedKey, '0{}b'.format(16)))
```

Table 18: Key derivation Algorithm