Journal of Advances in Mathematics and Computer Science

Volume 38, Issue 8, Page 86-100, 2023; Article no.JAMCS.101670
ISSN: 2456-9968
(Past name: British Journal of Mathematics & Computer Science, Past ISSN: 2231-0851)

ULBC: An Ultra Light-weight Block Cipher

Shreyasi Ghorai [a], Nilanjan Datta [b] and Mrinal Nandi [c*]

[a]*Department of Mathematics, West Bengal State University, India.*
[b] *Institute for Advancing Intelligence, TCG CREST, India.*
[c]*Department of Statistics, West Bengal State University, India.*

Authors' contributions

This work was carried out in collaboration between all authors. Author SG designed the study, performed the statistical analysis, wrote the protocol, and wrote the first draft of the manuscript. Author SG and Author ND managed the analyses of the study. Author MN managed the literature searches. All authors read and approved the final manuscript.

Article Information

DOI: 10.9734/JAMCS/2023/v38i81793

Open Peer Review History:
This journal follows the Advanced Open Peer Review policy. Identity of the Reviewers, Editor(s) and additional Reviewers, peer review comments, different versions of the manuscript, comments of the editors, etc are available here: https://www.sdiarticle5.com/review-history/101670

Received: 12/04/2023
Accepted: 17/06/2023
Published: 23/06/2023

Original Research Article

Abstract

After explicitly observing the design criteria of two popular block ciphers, namely PRESENT and GIFT, we have proposed a new S-box that would be useful for designing a new light-weight block cipher, we name it as ULBC. The primary goal of the S-Box is to reduce the implementation cost, and make it cheaper than the two block ciphers GIFT and PRESENT. In this design, we have also developed a new property like BOGI which would be extremely helpful in building light-weight block ciphers. Depending on this property we can appropriately design permutation layer, such that no bad output will go bad input. ULBC is composed of a S-box, associated bit wiring and key schedule. Also, we have produced some of cryptanalytic attacks to assure certain security level. We have used two different methods to calculate differential characteristics and

*Corresponding author: E-mail: mrinal.nandi1@gmail.com;

linear approximation. By using *-DDT method we have produced tight bounds of them. We find that 24 rounds are sufficient to resist some cryptanalytic attack. It is also found that by using this cipher 64-bits plain-text can be encrypted into 64-bits cipher-text.

Keywords: Light-weight block cipher; S-box; BOGI.

1 Introduction

Lightweight encryption is very important field in sense of IoT. For connecting the different devices in a single network for collecting and exchanging data over Internet securely (means maintaining authenticity and integrity) with or without help of human interaction is IoT. In IoT, Things are devices which creates a link between the physical and digital words, after connecting to the internet. IoT is used in so many devices like digital machine, sensors, RFID tags, actuators and mobile phones. Now data processing and protocols have to be designed by low computing power, small area, limited battery supply and small memory size, which also provide data security, data authenticity, data integrity, confidentiality. Resources include area, memory size, processing power, power consumption and energy. Many IoT devices are considered as low resource device [1].

Major security mechanism demands higher computation power and more resources. For balancing data security and constrained resources lightweight ciphers are required. Since reverse encryption is very easy and stream ciphers don't have diffusion property, in between Block cipher and stream cipher, Block cipher is more essential schemes. This cryptographic scheme applies a deterministic algorithm with symmetric key to encrypt a block of text, rather than encrypting one bit at a time as in stream cipher. Designing of block cipher depends on three principles namely (i) high speed (ii) low area (iii) area performance trade off. A block cipher consists of linear and nonlinear layer. As linear operations are very much friendly with speed, speed only depends on nonlinear operations. In lightweight cryptography designer's focus has been predominantly to minimize the hardware area. Choosing 4 bit Sbox is better than 8-bits Sbox. To design a lightweight block cipher, designing a Sbox with balanced component function, high nonlinearity, high algebraic degree, low differential uniformity, low linearity, BOGI property, no fixed point [2] and designing a permutation layer and round key generation play important role. First, we have to focus on analysing implementation cost. Then design permutation in such a manner that good diffusion property will be achieved, otherwise single trail will be noticed, then it will be easy to attack. Counting number of active S-boxes after different number of rounds is one more important thing in perspective of Linear and Differential cryptanalysis. In Differential cryptanalysis, fix input and output difference and repeatedly search the next best possible differential characteristic, then add the probabilities. At present MILP is very crucial supporting tool for linear and Differential cryptanalysis.

Through MILP we can minimize number of active S-boxes and characteristics of block ciphers can be represented by linear constrains. Actually, each cryptanalysis will be mapped to a Optimization problem. In November 2011, Nicky Mohua, Wang, preneel and Gu started using this tool for Differential and Linear cryptanalysis of cipher Enocoro-128v2, AES. They provide a c-program for differential cryptanalysis of AES, which generate a lp file, used CPLEX optimization studio to solve it. In 2013, Sun et. al. provides two different methods for cryptanalysis namely H-representation method and Logical Condition Modelling. They have discussed Differential attack on XBlock cipher, PRESENT-80, LBlock, SIMON, PRESENT-128, DES(L). In these two methods we get minimum number of active S-box in differential and linear trail for different rounds. We use second method, i.e., Logical condition modelling in calculating number of active S-box of our cipher ULBC. After that in 2015, Abdelkhaled et. al. proposed new method for calculating such bounds namely, *-DDT method. Here we split DDT table in separate tables with fixed probability equals 1 and all others are zero. By using this method, we not only calculate active S-box, we get associated probability of such S-boxes. So, we will get more tight bound.

1.1 Related works

In IBM research program, Horst Feistel designed Lucifar in 1971. Lucifar is 128-bits key and 128-bit block length first invented block cipher. So many cipher designs are submitted in this research project, some of these ciphers are pure SPN's , others are Feistel. However, the first known example of a practical cipher designed around a SPN. After cryptanalytic attack, it is seen that Lucifar can be broken by 2^{36} chosen plaintext and 2^{36} time complexity. In the year 1976, IBM designed a new cipher borrowing ideas from Lucifar with key size 64 bits. But later on, they reduced effective keysize 64-bits to 56-bits by including parity bits in each byte of the key as MSB and also change the S-box. This corrected one is known as DES (Data encryption standard). This design is based on Feistel network. In 1993, DES was broken by Mitsuru due to linear cryptanalysis. FFF's DES cracker broke a DES key in 56 hours in JULY 1998. This prompted the NBS to introduce Triple-DES in 1999, start AES specification process [3].

In 1998, ANSIX9.52 and FIPS46-3 introduced the Triple-DES encryption algorithm. DES-X, GDES, DESL and DESXL were proposed later on. James Massey and Xuejia Lai designed International Data Algorithm (IDEA) in 1991. This is an iterative block cipher and also it is oldest SPN cipher. In operates on 64 bit blocks and keysize is 128 bits. MESH, the cast Family are formed from inspiration of the block cipher IDEA. For erase out the drawbacks of DES or IDEA, Bruce Schneier designed block cipher Blowfish in 1993. It has a 64-bits block size and key length in 448 bits, which is 16-round Feistel cipher. RC5, is a block cipher designed by RON Rivest in 1994 and published in 1995, which is still widely used block cipher. Key sizes can vary from 0 to 2048 bits and block sizes can be 32, 64 or 128 bits. In the year 1997, the block cipher SQUARE is invented by Joan Daeman and Vincent Rijmen. SQUARE is a APN with eight rounds, handling on 128-bit blocks and taking a 128-bit key. Twofish also is submitted to the AES context in 1998 by Bruce Schneier, John Kelsey, Doug Whiting, Wagner, Chris Hall and Niel's Ferguson. In uses the same Feistel structure a DES. The block size is 128-bits and key size is of 128, 192 or 256-bits. Skipjack is a block cipher designed by US National Security Agency, which is 64-bit block cipher and unbalanced Feistel Network, uses 80-bits key. RC6 is evolution of RC5. It became one of the five finalists, which is designed by RON Rivest with Matt Robshaw, Ray Sidney and Yigun Lisa Yin in 1998. Like RC5, it is also Feistel design.

Anderson, Biham and Knudsen designed a block cipher named Serpent in 1998. In become one of five finalists. It has block size of 128 bits and supports a key size of 128, 192 and 256. It is 32 round Substitution Permutation Network operating on a block of four 32-bits words. MISTY-1, MISTY-2 and KASUMI are 64-bit block ciphers designed by Matsui with 128 bit secret key. MISTY-1 and KASUMI are type-I Feistel Network and MISTY-2 is Matsui Network [3].

In January 1997, NIST launched a contest to find replacement of DES. Rijndael [4] is the winner of AES contest,which is designed by Joan Daeman and Vincent Rijmen. It is a SP network with 10, 12 or 14 rounds for key sizes of 128,192 and 256 bits respectively. Francois-Xavier Standaert et al designed block cipher ICEBERG in 2004, which is 64-bit block cipher and key size is 128 bits. This cipher uses two different 4-bit S-boxes called S_0 and S_1, where each round consists a S-box layer and two linear diffusion layer that sandwiches the key mixing. The block cipher HIGHT is introduced by NSRI, this is used as a standard encryption algorithm in South Korea. This is 8-branch type feistel network. In 2007 CLEFIA is developed by Sony and University of Nagoya, which is also 128 bit type 2 feistel network. In 2007, Andrey Bogdanov discovered a 64-bit lightweight block cipher PRESENT [5]. PRESENT-80 and PRESENT-128 are there of key bits 80 and 128 respectively, which is low cost device like RFID-tags. PRESENT is a SPN with 31 rounds. In CHES 2017, S.Banik, T.Peyrin, Y.Todo, Sasaki presented a block cipher GIFT [6]. There are two versions of GIFT, GIFT-64 and GIFT-128 depending on block length, both has key length 128 bit.

1.2 Our contribution

We revisit the block cipher PRESENT and GIFT construction. We see that implementation cost is high. Through exhaustive search of many S-boxes by using different iterative component function, we find out a S-box with lower implementation cost. But there is one problem arises, three out of four outputs are bad. To solve this problem, we introduce BOGI like property. By designing properly permutation layer, we can eliminate the problem. In PRESENT, all are good output, in GIFT all bad output will go to good input in one round, whereas in our cipher bad output will go to good input after three rounds. We have seen that PRESENT S-box has XOR count 9. RECTANGLE S-box has XOR count 7 and GIFT S-box has XOR count 7. To minimize implementation cost we had been trying to minimize XOR count, which is an important criterion. ULBC S-box has XOR count 6. Three or four rounds, as any cipher achieves security after so many rounds, so this will not be any issue.

As we know $\{NOT, NAND, NOR\}$ as N-operations cost 1 unit and $\{XOR, XNOR\}$ as X-operation cost 2 units. Under this metric, we found that PRESENT Sbox requires 4N+9X operations, which has cost 22units. While RECTANGLE Sbox requires 4N+7X operations, which has cost 18units. GIFT S-box has 4N+6X operation. Implementation cost of GIFT S-box is 16units. Implementation cost of ULBC S-box is 16 units. Differential and linear cryptanalysis are most powerful technique in security analysis of block cipher. Differential uniformity and linearity of PRESENT are 4 and 6 respectively. Differential uniformity and linearity of GIFT are 6 and 4 respectively. But Differential uniformity and linearity of ULBC are 4 and 4 respectively.

2 Preliminaries

DDT: Difference Distribution Table
LAT: Linear Approximation Table

2.1 Properties of S-Box

- **Algebraic Degree:** The algebraic degree, $\deg(f)$, is the number of variables in the highest order term with non-zero coefficient.

  Algebraic degree of this S-box is 3.
- **Balancedness:** In mathematics and computer science, a balanced Boolean function is a Boolean function whose output yields as many 0s as 1s over its total input set. This means that for a uniformly random input string of bits, the probability of getting a 1 is $\frac{1}{2}$.

  Here each component function of S-box function should be balanced i.e., they should produce same number of 0's and 1's in truth table representation.
- **Differential Uniformity:** Maximum value in DDT table for nonzero input and output difference corresponding to S-box.
- **Linearity:** Maximum value in LAT table for nonzero linear input sum and output sum corresponding to S-box.
- **Differential and Linear Branch Number:** Differential and Linear Branch Number of an $n \times n$ S-box is denoted as DBN(S) and LBN(S) respectively and defined as

  $$DBN(S) = \min_{\delta \neq 0, \Delta \neq 0, DDT(\delta,\Delta) \neq 0} \{wt(\delta) + wt(\Delta)\}$$

  where $S(x) \oplus S(x \oplus \delta) = \Delta$; and

  $$LBN(S) = \min_{\alpha,\beta \in F_2^n, C_S(\alpha,\beta) \neq 0} \{wt(\alpha) + wt(\beta)\}$$

  where

  $$C_S(\alpha,\beta) = \sum_{x \in F_2^n} (-1)^{\beta S(x) + \alpha x}$$

  Differential and Linear branch number of our S-box is 2. Branching number of PRESENT block cipher is 3 and branching number of GIFT block cipher is 2.
- **BOGI Property:** In PRESENT block cipher no single bit input difference produce single bit output difference. So BOGI Property [6] is not required since all input and output are good. In DDT table single bit input difference may produce single bit output difference with non-zero probability. BOGI means Bad output should go to good input. For example

Table 1. BOGI property

	0001	0010	0100	1000
0001	0	2	2	0
0010	0	0	0	0
0100	0	0	0	0
1000	0	2	0	0

Here Bad inputs are all non-zero row. Good inputs are all zero row. Bad outputs are all non-zero column, good outputs are all zero column. Bad inputs are $\{0001, 1000\}$. Good inputs are $\{0010, 0100\}$. Bad outputs are $\{0010, 0100\}$. Good outputs are $\{0001, 1000\}$. We observe that Bad output and Good inputs are same. This implies in first round if Bad output occurs and bit permutation is as PRESENT block cipher, then input difference 0010 will go difference 0010, which may cause of attack if in second round input bit difference 0010 produce output bit difference 0010 with non-zero probability. As bad output does not produce bad input with non-zero probability, in second round after applying S-box good input occurs. So, in second round all single bit difference does not go single bit difference, thus each single bit difference affects two bits of output difference. From our DDT table we see that input difference 0001 may create output difference 1000.

Table 2. DDT Table

	0001	0010	0100	1000
0001	0	0	0	4
0010	0	2	2	2
0100	0	0	4	0
1000	0	0	0	0

After permutation it goes to 0100. If the input difference is 0100, it may create 0100.After applying permutation fourth bit of an S-box of next round is affected. So, difference will be 1000. As single bit difference 1000 does not go to any single bit difference. Now if the S-box output bit difference is 0010, we apply such a permutation that it affects first bit of an S-box of second round. 0001 goes to 1000 for single bit to single bit movement. After applying permutation 1000 goes to 0100. 0100 may create difference 0100, after permutation it goes to 1000, which does not go to single bit difference.

Now if the input difference is 0100, it may create 0100. After applying permutation fourth bit of an S-box of next round is affected. So, difference will be 1000. Next if the input difference is 1000, it does not go to any single bit difference. Thus after 3 rounds no single trail we will get.

$0001 \xrightarrow{S} 1000 \xrightarrow{P} 0100 \xrightarrow{S} 0100 \xrightarrow{P} 1000$
$0010 \xrightarrow{S} 0010 \xrightarrow{P} 0001 \xrightarrow{S} 1000 \xrightarrow{P} 0100 \xrightarrow{S} 0100 \xrightarrow{P} 1000$
$0010 \xrightarrow{S} 1000 \xrightarrow{P} 0100 \xrightarrow{S} 0100 \xrightarrow{P} 1000$
$0010 \xrightarrow{S} 0100 \xrightarrow{P} 1000$
$0100 \xrightarrow{S} 0100 \xrightarrow{P} 1000$

Thus, for secured cipher, there has to be at least one good input. Otherwise, we will get a single trail of active S-boxes. Here we observe that unlike GIFT block cipher, after certain rounds bad output goes to good input.

3 ULBC Block Cipher: Specification and Design Rationale

This section proposes our new SPN based ultra-lightweight block cipher.

3.1 Specification

The block length of this cipher is 64 bits. In each round 3 steps are performed, namely, sub cell, permutation bit, add round key. The cipher accepts 64-bit plaintext to generate ciphertext. Then all steps are performed one by one.

□ **Sub Cell:** To minimize area we have used 4-bit S-box. 4-bit to 4-bit S-box $S : \mathbb{F}_2^4 \to \mathbb{F}_2^4$ of our cipher is given by

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S[x]	2	F	0	8	6	1	E	4	5	A	B	3	9	C	D	7

Here 64-bit $p_0p_1p_2....p_{63}$ input is taken and divided into 16 parts $(x_0x_1....x_{15})$, each part contains 4 bits. Then apply S-box function to each of the part and get the output of sub cell.

□ **Bit Permutation:** The bit permutation used in our cipher is given by the following table, where each bit is taken as i

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
P[i]	1	16	35	50	49	0	19	34	33	48	3	18	17	32	51	2

i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
P[i]	5	20	39	54	53	4	23	38	37	52	7	22	21	36	55	6

i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
P[i]	9	24	43	58	57	8	27	42	41	56	11	26	25	40	59	10

i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
P[i]	13	28	47	62	61	12	31	46	45	60	15	30	29	44	63	14

□ **Key Schedule:** As encryption algorithm is publicly available, Key Schedule Algorithm and secret key is very crucial in designing block cipher. Round keys have to be independent, otherwise it can not resist Key schedule attack. According to NIST minimum key length to resist brute force attack is 112 bits [7].

We choose 128 bit key register. Here we specify adding round key and round constants. 64-bit key $K = u_0u_1u_2u_3v_0v_1v_2v_3u_4u_5u_6u_7v_4v_5v_6v_7.......u_{28}u_{29}u_{30}u_{31}v_{28}v_{29}v_{30}v_{31}$ extracted from 128 bit key state. If cipher state is $s_i, i = 0, 1, \ldots, 63$, adding round keys is given by

$$s_{4i+1} = s_{4i+1} \oplus u_{2i} \text{ for } \ i = 0, 1, \ldots, 15$$

$$s_i = s_i \oplus 1 \text{ for } \ i = 62$$

$$s_{4i+3} = s_{4i+3} \oplus v_{2i+1} \text{ for } \ i = 0, 1, \ldots, 15$$

Now 6-bit round constant$(c = c_0c_1c_2c_3c_4c_5)$ is xored in the following bits

$$s_2 = s_2 \oplus c_0$$

$$s_6 = s_6 \oplus c_1$$

$$s_{10} = s_{10} \oplus c_2$$

$$s_{14} = s_{14} \oplus c_3$$

$$s_{18} = s_{18} \oplus c_4$$

$$s_{22} = s_{22} \oplus c_5$$

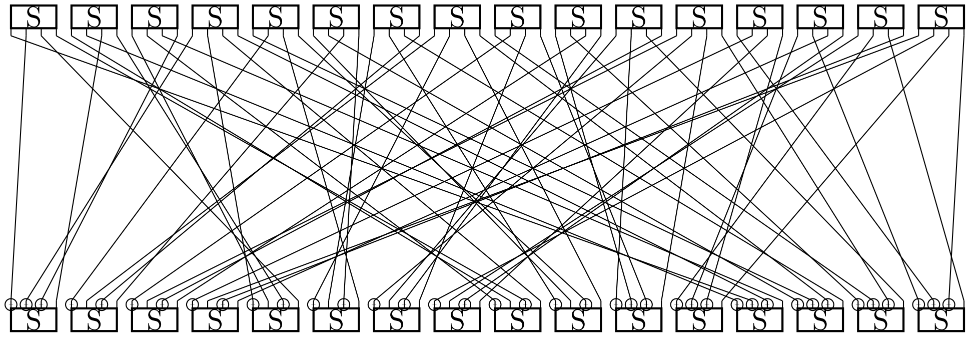

Fig. 1. Design of cipher

□ **Key register update:** In every round after extracting round key, key register must have to updated. 128-bit state key is divided into 8 parts, each part contains 16-bits i.e.$statekey = k_0k_1....k_7$. Now key register is updated as follows

$$k_0||k_1||...||k_7 = k_7 >> 12||S(k_0)||k_1||...||k_6$$

where $>> i$ is an i bit left rotation with in 16 bit word and $S(k_0)$ means applying S-box function to each nibble of k_0.

Design of our cipher is given as follows:

3.2 Design Rationale

□ **Choice of the S-box:** We have seen that PRESENT S-box has XOR count 9. RECTANGLE S-box has XOR count 7 and GIFT S-box has XOR count 7. To minimize implementation cost we had been trying to minimize XOR count, which is an important criterion. Our S-box has XOR count 6. This S-box is designed in a way that $x_0x_1x_2x_3$ is taken as input and $y_0y_1y_2y_3$ as output, which is given by

$$x_3 = XOR(x_3, (x_1 * x_0))$$
$$y_0 = XOR(x_0, x_3)$$
$$x_1 = XOR(x_1, y_0)$$
$$y_2 = XOR(x_2, (y_0 * x_1))$$
$$y_3 = XOR(x_3, (x_1 * y_2))$$
$$y_1 = XOR(x_1, NOT(y_3 * y_2))$$

Here we list down all the properties of our S-box:

- Algebraic degree of this S-box is 3.
- All component functions are balanced.
- There is no fixed point.
- Differential uniformity is 4
- Linearity is 4.

The DDT table 3 and the LAT table 4 of our S-box are given in the next page.

Table 3. DDT table of S-box

DIFF	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	0	0	2	0	2	4	0	4	0	0	2	0	2
2	0	0	2	0	2	2	0	2	2	2	0	2	0	0	2	0
3	0	4	2	0	0	0	2	0	0	0	2	0	0	0	2	4
4	0	0	0	0	4	0	4	0	0	0	0	0	4	0	4	0
5	0	0	0	4	2	0	2	0	0	4	0	0	2	0	2	0
6	0	0	2	0	0	0	2	0	2	2	0	2	2	2	0	2
7	0	4	2	0	0	0	2	4	0	0	2	0	0	0	2	0
8	0	0	0	4	0	2	0	2	0	0	0	4	0	2	0	2
9	0	0	0	4	0	0	0	0	4	4	4	0	0	0	0	0
A	0	0	2	0	0	2	2	2	2	2	0	2	2	0	0	0
B	0	4	2	0	2	0	0	0	0	0	2	0	2	4	0	0
C	0	0	0	4	0	2	0	2	0	0	0	4	0	2	0	2
D	0	0	0	0	2	2	2	2	0	0	0	0	2	2	2	2
E	0	0	2	0	2	0	0	0	2	2	0	2	0	2	2	2
F	0	4	2	0	2	4	0	0	0	0	2	0	2	0	0	0

Table 4. LAT table

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

4 Cryptanalysis

4.1 MILP Modeling for Differential and Linear Cryptanalysis

Whenever we design a block cipher, we have to provide some cryptanalytic attack resistant criteria. Among all of attacks differential and linear cryptanalysis are very important. In differential cryptanalysis we analyse how difference in plaintext moves towards in difference in ciphertext. Here we use the technique key Xoring can't affect in differences. Thus only we have to lookout S-box and permutation. In S-box, from DDT table, we come to know which difference propagation is possible and which are not. Permutation is used as of it. Corresponding to each difference propagation, a fixed probability is attained. Through H-representation, Logical condition Modelling and *-DDT method, we represent all linear inequalities corresponding to characterisation of DDT table in different ways. In wide trail strategy, we have to provide minimum number of active S-box with maximum differential probability. If number of active S-box is N_s and maximum differential probability is $2^{-\delta}$, then maximum probability of the differential characteristics is bounded by $(2^{-\delta})^{N_s}$. Here the attacker can query $(2^{-\delta})^{N_s}$ pairs, as any differences is related to some plaintext and ciphertext. Query of attacker is upper bounded by 2^{64}, as it is 64-bit cipher i.e., $\delta \cdot N_s > 64$. As δ is fixed, we have to examine after what number of rounds, number of active S-box satisfy this property. Then we can conclude this fixed number of rounds is sufficient to resist differential attack.

In linear cryptanalysis we approximate S-box by a linear expression with good bias value (away from 0). By this process we can build up a relation between plaintext bits and the bits prior to last round and key bits. Each round approximate S-box by linear expression separately and by using pilling up lemma taking XOR sum of all expression, we can get required probabilistic relation with bias value. Actually, by deriving relations between plaintext, ciphertext and key bits, known plaintext ciphertext attack can be done. For each plaintext ciphertext partial decryption can be done by using all possible values related to key bits of linear approximation. Now if there is large number of plaintexts, ciphertext, calculate counter of each key equals total number of plaintext-ciphertext pair satisfies by this key value. Accurate key has counter value $\frac{T}{2} + \frac{T}{32}$, where T is total number of plaintext-ciphertext pairs. In this attack, we can distinguish a cipher from random permutation and can recover some key bits immediately. In random permutation cases, all keys have counter value $\frac{T}{2}$.

At first from S-box we can calculate LAT table, which represents bias of all linear expressions. From this table using H-representation, Logical Condition Modelling and *-DDT, we can derive constraints representing LAT table in different ways. Key Xor will not affect the bias value. Permutation is used as of it. In wide trail strategy, we have to provide minimum number of active S-box with maximum bias value. Depending on the number of active S-boxes, we can approximate the value of correlation potential or linear hull effect. on other way by *-DDT method we can exactly calculate bias value. If bias of linear characteristics is ϵ, correlation potential will be $4\epsilon^2$. To resist linear attack, correlation potential have to be greater than 2^{-n}, where n is block length. Now for linear cryptanalysis, we have to calculate after how many numbers of rounds, this amount of correlation potential will be assured.

We use Mohua et. al.'s framework [4] for initially formulate S-box input output, linear transformation, nonzero input. A new binary variable b_i is introduced corresponding to each bit-difference. The value of this binary variable is 1 if and only if the corresponding difference is nonzero.

There is a binary variable S_j corresponding to S-box. S_j takes value 1 iff input to this s-box is nonzero. As our main focus is to calculate minimum number of active S-box, the objective function will be $\sum_j S_j$.

Now to formulate the concept S_j is 1 for non-zero input. Here for 4×4 S-box the required constraints are given by

$$\begin{gathered} b_{j_0} + b_{j_1} + b_{j_2} + b_{j_3} - S_j \geq 0 \\ S_j - b_{j_k} \geq 0, \quad k \in \{0,1,2,3\} \end{gathered}$$

where input difference to an S-box is $b_{j_0}b_{j_1}b_{j_2}b_{j_3}$ and output to an S-box is $y_{j_0}y_{j_1}y_{j_2}y_{j_3}$. The constraints to ensure that nonzero input difference in each S-box must results nonzero output difference are

$$\begin{gathered} b_{j_0} + b_{j_1} + b_{j_2} + b_{j_3} - i(j) \geq 0 \\ y_{j_0} + y_{j_1} + y_{j_2} + y_{j_3} - o(j) \geq 0 \\ i(j) - b_{j_k} \geq 0, \quad k \in \{0,1,2,3\} \\ o(j) - y_{j_k} \geq 0, \quad k \in \{0,1,2,3\} \\ i(j) - o(j) \geq 0, \quad o(j) - i(j) \geq 0 \end{gathered}$$

Now to characterize DDT or LAT table as constraints of MILP, different methods are there. In H-representation method, Sun et al. first finds solution space bounded by possible values, then eliminating impossible one figure out more accurate solution space. In Logical condition modelling, Sun et. al. [8] represent S-box as product of sum form of Boolean function. Each component term of function corresponds to one constraint of MILP. Abdelkhalek et al. in their paper [9] enlightening further a new way for tight bound of differential probability and linear hull apart from calculating number of active S-boxes.

4.1.1 H-representation for differential attack

As we know, convex hull of set of points is smallest convex set containing all points. It also can be described by solution of a set of finitely many equations and inequalities. This is called H-representation of convex hull.

Now to generate constraints related to actual characterization of DDT or LAT table, all possible input output differences taken as a point in $\mathbb{R}^{2n}$, for $n \times n$ S-box. Next, we have to compute H-representation of convex hull of all possible input output differential pattern. In sage math, define points related to all possible pattern, then generate a polyhedron. Next using inequality-generator, we can find out inequalities that bounds the feasible region. Each inequality cut out certain number of impossible differential patterns.

In our cipher we got 279 inequalities from H-representations. Using all these inequalities is very much time killing algorithm. Thus, there is a greedy algorithm to select best inequality from this huge number of inequalities. Best inequality is that inequality, which cut off maximum number of impossible patterns. Now select best inequality, such that maximum number of impossible patterns removed and remove these impossible one from total set of impossible patterns then continue similar process. After certain times, total set of impossible patterns will be nullified. Collect all best inequalities and use it in MILP program.

4.1.2 Logical condition modelling for differential and linear attack

To represent S-box, $f(x,y)$ be a Boolean function of 2n bits inputs x and 1 bit output y. Now the value of $f(x,y)$ is 1 if and only if input-output differential or linear pattern is possible in DDT/LAT and 0 otherwise. To describe all characteristics of DDT/LAT table, we have to generate linear inequality corresponding to impossible differential, such that the constraints exclude that impossible pattern. As there are so many impossible patterns, large no. of equation will occur. We have to minimize the no. of equations using QM-algorithm. In Logic Friday we enter the truth table corresponding to DDT/LAT table, to generate logic function corresponding to it. Minimizing by QM-algorithm logic function corresponding to DDT table is given by $f(x,y) =$

$(x_0 + x_1' + x_2 + y_1' + y_3')(x_0' + x_1 + x_2 + x_3' + y_1')(x_1 + x_2 + y_1' + y_3)(x_0 + x_1 + x_3 + y_0 + y_1' + y_2' + y_3)(x_0' + x_1 + x_3 + y_0 + y_2 + y_3)(x_1' + x_2' + x_3 + y_0 + y_3')(x_0' + x2 + x_3 + y_3)(x_3 + y_0' + y_1 + y_2' + y_3)(x_1' + x_2' + x_3' + y_0' + y_3')(x_1 + x_2' + x_3 + y_0' + y_1' + y_3')(x_0 + x_1' + x_2' + x_3 + y_0' + y_2' + y_3)(x_0' + x_1' + x_3 + y_0' + y_1' + y_2 + y_3)(x_0' + x_2' + x_3' + y_1' + y_2')(x_0 + x_2 + x_3 + y_3')(x_2' + x_3 + y_0 + y_1 + y_2)(x_2' + x_3' + y_0' + y_1 + y_2)(x_2 + y_0 + y_1 + y_2 + y_3')(x_3' + y_0 + y_1 + y_2 + y_3)(x_2' + y_0 + y_1 + y_2' + y_3')(x_2 + x_3 + y_0' + y_1 + y_2)(x_3' + y_0' + y_1 + y_2' + y_3')(x_1' + x_2 + y_1 + y_3)(x_0' + x_1' + x_2 + x_3' + y_1)(x_0 + x_1 + x_2 + y_1 + y_3')(x_2 + y_0 + y_1 + y_2' + y_3)(x_0 + x_2' + x_3' + y_2 + y_3)(x_0 + x_1' + x_2' + y_0 + y_1' + y_2)(x_1 + x_2' + x_3' + y_0 + y_1' + y_3')(x_0 + x_1 + x_2' + y_0' + y_1' + y_2)(x_0' + x_1' + x_2' + y_0 + y_1' + y_2')(x_0' + x_1 + y_0' + y_1' + y_2' + y_3)$

Then minimizing by QM-algorithm logic function corresponding to LAT table is given by $f(x,y) =$

$(x_0+x_3'+y_0+y_2)(x_0'+x_3+y_0+y_2+y_3)(x_0+x_1'+y_1+y_2+y_3)(x_0+x_3+y_0'+y_2)(x_1'+y_0+y_1+y_2)(x_0'+x_1+y_0+y_2'+y_3')(x_0'+x_1'+x_3'+y_0'+y_2'+y_3)(x_0'+x_1'+x_2+y_0'+y_1'+y_2')(x_0'+x_1'+x_3+y_0'+y_2'+y_3')(x_0'+x_1'+x_2'+y_0'+y_1+y_2')(x_1+y_0+y_1'+y_2)(x_0+x_1+y_0+y_2'+y_3)(x_0+x_1+x_2'+x_3'+y_3')(x_0+x_1+x_2+y_0'+y_2')(x_0+x_1+x_2+x_3+y_3')(x_0+x_2+y_0'+y_2+y_3)(x_1+x_2'+x_3'+y_0+y_1+y_3)(x_0+x_2'+y_0+y_1+y_2)(x_0+x_1+x_3+y_2'+y_3)(x_1+x_2+x_3'+y_0+y_1'+y_3)(x_3'+y_0+y_1+y_2+y_3)(x_1+x_2'+x_3+y_0+y_1'+y_3)(x_1+x_2+x_3+y_0+y_1+y_2')(x_3'+y_0+y_1'+y_2+y_3')(x_0+x_2'+y_0'+y_1'+y_2+y_3')$

where the number of terms is minimized by QM-algorithm. Now each term can be converted to an inequality. Constraint corresponding to first term of our logic function is given by

$$x_0 + (1 - x_1) + x_2 + (1 - y_1) + (1 - y_3) \geq 1.$$

Number of active S-boxes in different rounds for differential attack are 1(1 -round), 2(2- round), 3(3 -round), 4(4 -round), 5(5- round), 9(7- round), 11(8-round). Maximum differential probability in 8 round is 2^{-22} . Number of active S-boxes in different rounds for Linear attack are 1(1-round), 2(2-round), 3(3-round), 4(4-round), 5(5-round), 6(6-round), 7(7-round), 8(8-round), 9(9 round).

4.1.3 *-DDT

By using Logical Conditioning Model, we can calculate number of active S-box. By knowing this we can't know S-box corresponding to which difference probability is active. From this we can only assure certain bounds of probability. To know the exact value, we have to use *-DDT method. In this method we generate linear constraints for a *-DDT based on logical condition modelling.

The idea is to split DDT table into multiple tables for each probability and assign different variable corresponding to different table. All the non-trivial entries of our S-box are either $2(2^{-3})$ or $4(2^{-2})$, and we prepare two separate table. In one table we put 1 for 2^{-3} and 0 for others. Another one table is constructed by putting 1 for 2^{-2} and 0 for others. The main DDT table is given below (see Table 5).

Table 5. Logical conditioning model result

DIFFERENCE	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	0	0	2	0	2	4	0	4	0	0	2	0	2
2	0	0	2	0	2	2	0	2	2	2	0	2	0	0	2	0
3	0	4	2	0	0	0	2	0	0	0	2	0	0	0	2	4
4	0	0	0	0	4	0	4	0	0	0	0	0	4	0	4	0
5	0	0	0	4	2	0	2	0	0	4	0	0	2	0	2	0
6	0	0	2	0	0	0	2	0	2	2	0	2	2	2	0	2
7	0	4	2	0	0	0	2	4	0	0	2	0	0	0	2	0
8	0	0	0	4	0	2	0	2	0	0	0	4	0	2	0	2
9	0	0	0	4	0	0	0	0	4	4	4	0	0	0	0	0
A	0	0	2	0	0	2	2	2	2	2	0	2	2	0	0	0
B	0	4	2	0	2	0	0	0	0	0	2	0	2	4	0	0
C	0	0	0	4	0	2	0	2	0	0	0	4	0	2	0	2
D	0	0	0	0	2	2	2	2	0	0	0	0	2	2	2	2
E	0	0	2	0	2	0	0	0	2	2	0	2	0	2	2	2
F	0	4	2	0	2	4	0	0	0	0	2	0	2	0	0	0

And the 2^{-2}-DDT and 2^{-3}-DDT are given below (see Table 6).

Table 6. Split DDT table

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

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

Next we introduce variables $Q_{2^{-3}}, Q_{2^{-2}} \in \{0, 1\}$ and conditional constraints such that the inequalities corresponding to table for 2^{-3} is effective only when $Q_{2^{-3}} = 1$ and the inequalities corresponding to table for 2^{-2} is effective only when $Q_{2^{-2}} = 1$. We also introduce another indicator variable Q that takes 1 if S-box is active and 0 otherwise. We then set $Q = Q_{2^{-3}} + Q_{2^{-2}}$ for each S-box. If S-box is active, the set of linear inequalities for only one pb-DDT is imposed. Let $< \vec{a}, (\vec{x}, \vec{y}) >\geq b$ be linear inequalities, then the conditional constraints are represented by

$< \vec{a}, (\vec{x}, \vec{y}) > +M(1 - Q_{pb}) \geq b$ where M is a sufficiently big integer.

The objective function will be minimizing $\sum -log_2(pb)Q_{pb}$.

Here our objective function is $3 * Q_{2^{-3}} + 2 * Q_{2^{-2}}$.

Differential probability of our cipher using *-DDT method upto 9 rounds are 2^{-2},2^{-4}, 2^{-6}, 2^{-9},2^{-11}, 2^{-16}, 2^{-16}, 2^{-20}, 2^{-24}. On taking average we can conclude that 24 rounds are required to resist differential attack. Bias value in Linear cryptanalysis of our cipher using this method upto 8 rounds are 2^{-2}, 2^{-3}, 2^{-4}, 2^{-5}, 2^{-7}, 2^{-8}, 2^{-10}, 2^{-11}. On taking average it shows that 23 rounds are required to resist linear attack.

4.2 Invariant subspace attack

Key idea of Invariant Subspace Attack [10] on SPN is to find invariant affine subspace transition and related weak keys. By analysing weak keys, design a proper key schedule to rule out all types of invariant subspace transitions. In this kind of transition $SP : F_2^n \to F_2^n$ (keyless round function), if $u \oplus A$ is an affine subspace of F_2^n and $u \oplus A \to v \oplus A'$, [11] then $v \oplus A'$ is also an affine subspace. Thus $u \oplus A$ transforms $v \oplus A'$, weak keys are in $A \cap A' \oplus u \oplus v$. Now if $A = A'$, $u \oplus A \xrightarrow{\text{SP}} v \oplus A \xrightarrow{\text{key xor}} u \oplus A$. In Chosen Plaintext Attack, if all block of Plaintexts is in invariant subspace $A \oplus u$, ciphertext will be determined as it will be in same subspace.

Now using the idea $u \oplus A$ is affine subspace in F_2^n, $v \oplus A$ is also an affine subspace and after brute force search, we can find out one dimensional invariant subspace transitions and related weak keys.

$$\begin{array}{rl}
\{0,6\} \oplus 3 \to \{0,6\} \oplus 4, & K \in \{0,6\} \oplus 1 = \{0,6\} \oplus 7 \\
\{0,A\} \oplus 2 \to \{0,A\} \oplus 0, & K \in \{0,A\} \oplus 2 = \{0,A\} \oplus 8 \\
\{0,D\} \oplus 0 \to \{0,D\} \oplus 1, & K \in \{0,D\} \oplus 1 = \{0,D\} \oplus C \\
\{0,E\} \oplus 4 \to \{0,E\} \oplus 9, & K \in \{0,E\} \oplus 3 = \{0,E\} \oplus D
\end{array}$$

Also there are two 2-dimensional invariant subspace transitions

$$\begin{array}{rl}
0 \oplus \{0,5,9,C\} \to 1 \oplus \{0,3,4,7\}, & K = 1 \\
0 \oplus \{0,4,B,F\} \to 1 \oplus \{0,8,2,A\}, & K = 1
\end{array}$$

Now by observing weak key sets 1, 2, 8 are there. We Xor 4 to nibble, to get rid of invariant subspace transition. This way we can prevent invariant subspace attack.

4.3 Impossible differential attack

If input difference Δ_1 can't reach to output difference Δ_2 after some particular rounds. Then (Δ_1, Δ_2) is called impossible differential. By exploiting this property and choosing such kind of plaintext and its corresponding ciphertext, subkey values can be guessed by partial encryption and decryption. The related subkeys will be wrong. Thus, by the help of this property we can discard some subkeys. So possible subkey search space will be smaller.

Our cipher achieves full diffusion only after 3 rounds. By using MILP [12] and setting different input difference and output differences, we analyse the possible differential pairs and related S-boxes [13], [14]. We exhaustively tested input difference and output differences satisfying the following conditions:

- The input difference activates only first S-box.
- The output difference activates only on S-box at a time. For the first condition there are 15 such input differences. For the second condition there are $15 \times 16 = 240$ such output differences.

Hence, we tested $240 \times 15 = 3600$ pairs of input output differences. From search result, we can see that 1228 pairs are impossible in seven rounds. All these impossible differences will be possible in eight rounds.

5 Conclusion and Future Works

In this paper we have designed a new 64-bit block cipher namely ULBC, which has less implementation cost relative to other block ciphers. We have used two different methods to calculate differential characteristics and linear approximation. By using *-DDT method we have produced tight bounds of them. 24 rounds are sufficient to resist some Cryptanalytic attack. By using this cipher 64-bits plain-text can be encrypted into 64-bits cipher-text. But in public domain message can be of any length, we have to encrypt messages with variable length and output certain desired bits. For this purpose, we have to select some block cipher modes.

In future, we are going to implement this cipher in different mode and want to see which mode is most compactable with this cipher.

Acknowledgement

Author SG get CSIR scholarship for her reaserch. Author ND and Author MN had no funding agency for this reaserch.

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Mohd BJ, Hayajneh T. Lightweight block ciphers for iot: Energy optimization and survivability techniques. IEEE Acess. 2018;6:35966-35978.

[2] Mohamed K, Pauzi MNM, Ali FHHM, Ariffin S, Zulkipli NHN. Study of S-box properties in block cipher. In 2014 International Conference on Computer, Communications, and Control Technology(14CT). 2014;362-366. IEEE.

[3] Avanzi R. A salad of block ciphers. Cryptology ePrint Archive; 2016.

[4] Daeman J, Rijmen V. The block cipher Rijndael. In First candidate conference (AeS1). 1999;343-348.

[5] Bogdanov A, Knudsen LR, Leander G, Paar C, Poschmann A, Robshaw MJB, Seurin Y, Vikkelsoe C. Present: An ultra-lightweight block cipher. Cryptographic Hardware and Embedded System-CHES 2007: 9th International workshop, Vienna, Austria. Proceedings 9. 2007;450-466. Springer

[6] Banik S, Pandey SK, Peyrin T, Sasaki Y, Sim SM, Todo Y. GIFT: A small present - towards reaching the limit of lightweight encryption. Cryptographic Hardware and Embedded System-CHES 2017: 19th International Conference, Taipei, Taiwan. Proceedings 9. 2017;321-345.

[7] Imdad M, Ramli SN, Mahdin H. An enhanced key schedule algorithm of PRESENT-128 block cipher for random and non-random secret keys. Symmetry, MDPI. 2022;604.

[8] Sun S, Hu L, Weng P, Qiao K, Ma X, Song L. Aumatic security evaluation and (related-key) differential characteristic search: Application to simon, present, lblock, des(1) and other bit-oriented block ciphers. Advances in Cryptology-ASIACRYPT 2014:20th Annual International Conference on the Theory and Applications of Cryptology and Information Security. Kaoshiung, Taiwan, ROC. Proceedings, Part I. 2014;20:158-178. Springer

[9] Abdelkhalek A, Sasaki Y, Todo Y, Tolba M, Youssef AM. Milp modelling for (large) s-boxes to optimize probability of differential characteristics. IACR Transaction on Symmetric Cryptology. 2017;99-12.

[10] Guo J, Jean J, Nikolic I, Quao K, Sasaki Y, Meng Sim S. Invariant Subspace Attack against midori64 and the resistance criteria for s-box designs. Cryptology ePrint Archive; 2016.

[11] Leander G, Minaud B, Sondre Ronjom. A generic approach to invariant subspace attacks: Cryptanalysis of Robin, iSCREAM and Zorro. Advances in Cryptology-Eurocrypt 2015: 34th Annual international conference on the Theory and Applications of Cryptographic techniques, Sofia, Bulgaria. Proceedings Part I. 2015;254-283. Springer

[12] Sasaki Y, Todo Y. New impossible differential search tool from design and cryptanalysis aspects: Revealing structural properties of several ciphers. Advances in Cryptology-EUROCRYPT 2017:36th Annual International Conference on the Theory and Applications of Cryptographic Techniques. Paris, France. Proceedings, Part III. 2017;36:185-215. Springer

[13] Sakiyama K, Sasaki Y, Li Y. Security of block cipher: from algorithm design to hardware implementation. John Wiley & Sons; 2016.

[14] Stinson DR, Maura Paterson. Cryptography: Theory and practice. CRC Press; 2018.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
https://www.sdiarticle5.com/review-history/101670