Full-Round Differential Attack on DoT Block Cipher

Manoj Kumar
Scientific Analysis Group, DRDO, Metcalfe House Complex,
Delhi-110 054, INDIA.
manojkumar@sag.drdo.in

Abstract

The lightweight encryption design DoT was published by Patil et al in 2019. It is based on SPN (substitution permutation network) structure. Its block and key size are 64-bit and 128-bit respectively. In this paper, we analyse the security of DoT against differential attack and present a series of differential distinguishers for full-round DOT. Our analysis proves that DoT we can be distinguished from a random permutation with probability equal to 2^{-62}. Diffusion layer of DoT is a combination of byte shuffling, 8-P permutation, 32-bit word shuffling and circular shift operations. We analyse the security of DoT with and without 8-P permutation in its diffusion layer. Our results indicate that DoT provides better resistance to differential attack without using the 8-P permutation.

Keywords: Block Cipher, Lightweight Block Cipher, Differential Cryptanalysis, Branch-and-bound Algorithm.

1 Introduction

Lightweight cryptography has evolved to provide the security solutions during automatic connections between humans and machines through advanced technologies. Lightweight block ciphers are used to secure the applications like RFID tags, wireless sensor nodes and Internet of things etc [13]. Memory and throughput requirements are considered as important design parameters for these applications. DoT is claimed as a compact and smallest cipher which requires 993 GEs only and it is presented as a significant option for low power and compact memory applications [12]. We analyse the security of DoT against differential attack and present a distinguisher for full rounds.

Differential cryptanalysis was proposed by Biham and Shamir in 1990 [1]. This was the first successful cryptanalytic attempt which reduced the complexity of DES better than exhaustive search. This attack is considered as the basic cryptanalysis method and it is necessary to provide the security proofs against differential attack for new design proposals. A Successful differential attack depends on the existence of high probability trails covering the maximum rounds of the cipher. We apply branch-and-bound based algorithm [6] to search the optimal differential trails in DoT and present 31-round trail with 31 active S-boxes for DoT block cipher.

Block cipher designs are either based on Feistel or Substitution Permutation Network (SPN) structure [2]. Round function in each structure is designed to satisfy the confusion and diffusion properties. DoT is based on SPN structure that processes the full input block through its round function in each round. It applies single 4-bit S-box sixteen times in parallel but its diffusion layer is made up of various components like byte and word shifts, bit permutation and circular shift operations. S-box is used to provide the confusion and diffusion is achieved by combining the bit/byte/word permutations. We also analyse the diffusion layer of DoT and report our results in this paper.

Rest part of the paper is organised in the following way. In section 2, we discuss block ciphers DoT and its key expansion algorithm in brief. In section 3, we apply branch-and-bound based algorithm to search the differential trails and present full-round differential distinguisher for DoT. We analyse the diffusion layer with and without 8-P permutation in section 4.

Notations: The following notations are used throughout the paper:
- P :64-bit input plaintext block
- C :64-bit output ciphertext block
- P_i^{64} :64-bit input to round function
- K :128 bits master key
- UK :128-bit updated key register
- RK_i :64-bit round subkey
- T_1^{64} :64-bit temporary block
- T_L^{32} :Left 32-bit block
- T_R^{32} :Right 32-bit block
- T_1^4, T_2^4 :4-bit temporary nibbles
- $\oplus$:Bitwise exclusive-OR operation
- $\lll n$:Left cyclic shift by n bits
- $\ggg n$:Right cyclic shift by n bits
- $\|$:Concatenation of two n-bit strings
- $[i]_2$:Binary value of integer i

2 Lightweight Encryption Design: DoT

Lightweight block cipher DoT was published by Patil et al at advances in intelligent systems and computing 2019 [12]. It is presented as the compact and smallest design till date for memory requirement and execution time. It consists of a single S-box, shift operations and XOR gates. We provide a brief description of DoT in this section.

2.1 Description

DoT encrypts a 64-bit message block using the 128-bit master key. First, it divides the input plaintext in 64-bit blocks and encrypts each message block to generate the 64-bit ciphertext block. Each input block is processed through the round function 31 times. In each round, there are parallel applications of single 4-bit S-box and it uses nibble/word shuffling, 8-bit permutation and cyclic shift operations.

2.2 Encryption Algorithm

We describe the encryption algorithm using substitution operation S_B, 8-bit permutation P_8, shuffling of i^{th} byte B_i and circular shift operations (Algorithm 1, Fig. 1). Round function applies the substitution operation S_B on 64-bit state value $P_i^{64} \oplus RK_i$ by dividing the 64-bit word into 4-bit nibbles. It applies the 4-bit S-box (Table 1) 16 times in parallel on each nibble. After that, 8-bit permutation P_8 is applied on each byte that outputs a 64-bit value T_1^{64} which is divided into eight bytes (Fig. 2). These bytes are rearranged to produce two 32-bit outputs T_L^{32} and T_R^{32}. Finally, left circular shift by 25 bits is applied on T_L^{32} and right circular shift by 31 bits is applied on T_R^{32} and the output values are concatenated to produce P_{i+1}^{64}. These operations are applied 31 times to get the 64-bit ciphertext. In each round, we need 64-bit round subkey (RK_i) which is obtained from the 128-bit master secret key (K) by using a key expansion algorithm (Algorithm 2) [12].

Algorithm 1: Encryption Algorithm

Input: $P = P_1^{64}$ and RK_i obtained from K
Output: $C = P_{32}^{64} \oplus RK_{32}$
for *i=1 to 31* **do**
  $T_1^{64} = P_8(S_B(P_i^{64} \oplus RK_i))$
  $T_1^{64} \rightarrow (B_1||B_2||B_3||B_4||B_5||B_6||B_7||B_8)$
  $T_L^{32} = (B_1||B_3||B_5||B_7)$
  $T_R^{32} = (B_2||B_4||B_6||B_8)$
  $P_{i+1}^{64} = (T_L^{32} << 25)||(T_R^{32} >> 31)$
end

Algorithm 2: Key Expansion Algorithm

Input: 128-bit master key K
Output: 64-bit round subkeys RK_i
for *i=1 to 31* **do**
  RK_i = 64 leftmost bits of K
  $UK \leftarrow K <<< 13$
  $T_1^4 = (UK >> 124)$ & 0xF
  $T_2^4 = (UK >> 120)$ & 0xF
  $K = S(T_1^4) << 124||S(T_2^4) << 120||((UK << 8) >> 8) \oplus ([i]_2 << 59)$
end

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

Table 1: S-Box

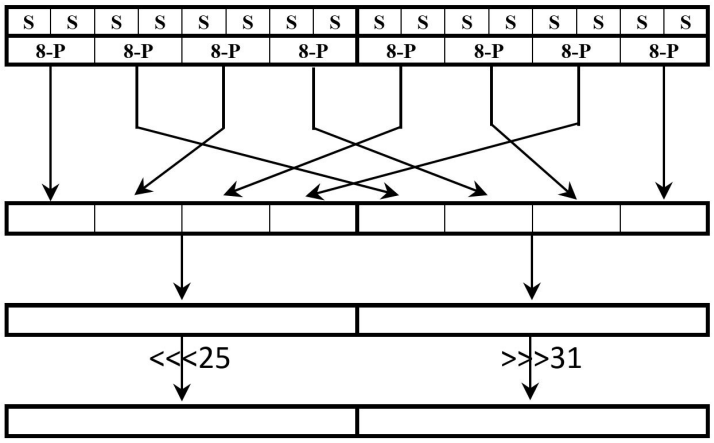

Figure 1: Round Function of DoT

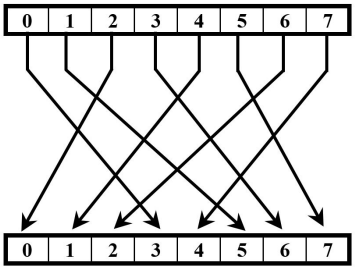

Figure 2: 8-P Permutation

3 Differential Attack on DoT

High probability differential trails are the basis for a successful differential attack. We need automated tools to compute the high probability differential trails covering the maximum number of rounds. There are various automated methods to construct the optimal differential trails e.g. branch-and-bound based algorithm [11] etc. We use branch-and-bound based algorithm to construct the differential distinguisher for DoT. We improve the search algorithm presented by Kumar el al in [10] and apply the improved algorithm to search the differential trails in DoT. We describe the generalized algorithm that takes n-bit block X as input and nibble (S-box) size is m-bit for differential trail search (Algorithm 3). The input block X is divided in $t(= n/m)$ nibbles, then DDT (Difference distribution table) is applied on each nibble to get the possible output differences with probability p_i. Any non-zero input to DDT contributes towards the probability of trail. We start with one non-zero nibble in the input X and track the propagation of this difference. We also tried with two or more non-zero nibbles in the input X and tracked the difference in same way to find out a differential trail with better probability. After trying all possible non-zero values for each nibble, we filter out the trails containing least number of active S-boxes and providing the maximum probability.

Algorithm 3: Branch-and-bound based algorithm for differential trail

Input: X, S-box size (bits) $= m$, $least = A$, $optimal = B$ and $N_{SB} = 0$
Output: Optimal Differential Trails with best probability P_T
for *i=1 to t* **do**
  for *j=1 to* $2^m - 1$ **do**
    $Nibble_i = j; Nibble_{(\neq i)} = 0;$
    $X = (Nibble_1, Nibble_2, \ldots, Nibble_t);$
    for *Round=1 to r* **do**
      Count($\neq$ Non-zero nibbles in X);
      $N_{SB} = N_{SB} + Count;$
      for $k_1 = 1$ *to 15* **do**
        for $k_2 = 1$ *to 15* **do**
          .....
          for $k_t = 1$ *to 15* **do**
            $p_1 = DDT[Nibble_1][k_1];$
            If($p_1 = 0$) continue;
            $p_2 = DDT[Nibble_2][k_2];$
            If($p_2 = 0$) continue;
            .....
            $p_t = DDT[Nibble_t][k_t];$
            If($p_t = 0$) continue;
            $Y = (k_1, k_2, \ldots, k_t);$
            $X = P_{Layer}(Y);$
            $P_{Round} = p_1 * p_2 * ... * p_t$
          end
        end
      end
    end
    $P_T = P_1 * P_2 * ... * P_r$
    if $N_{SB} \leq least$ *and* $P_T \leq optimal$ **then**
      return: Optimal differential trail with probability P_T
    end
  end
end

3.1 Differential Distinguisher for DoT

DoT applies round key addition, 4-bit S-box, permutation on 8 bits, word shifts and left/right rotations in each round [12]. Word shift, 8-bit permutation and rotations are linear operations that always generates a fixed output difference for a particular input difference. We also get the same difference value $\triangle U = U_1 \oplus U_2$ before and after the add round key operation. However, substitute nibble operation does not provide the fix output difference for any input difference due to its non-linearity. Therefore, we need to construct the probabilistic relations for each input and corresponding output differences to S-box. This is tabulated to get a $2^4 \times 2^4$ difference distribution table (DDT) [5] for 4-bit S-box. For any non-zero input ($\triangle_i \neq 0$), we get more than one non-zero outcomes ($\triangle_o \neq 0$). For DoT S-box, there are 256 possible input and output difference pairs ($\triangle_i, \triangle_o$) that are presented in the following table.

$\triangle_o \rightarrow$ $\triangle_i \downarrow$	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	0	0	0	2	2	2	2	2	2	2	2
2	0	0	2	0	0	2	2	2	0	0	2	0	0	2	2	2
3	0	4	2	0	2	0	0	0	2	0	0	0	0	0	2	4
4	0	0	0	4	0	4	0	0	0	2	0	2	0	2	0	2
5	0	0	0	0	4	0	4	0	0	2	0	2	0	2	0	2
6	0	0	2	0	0	2	2	2	0	2	2	2	0	0	2	0
7	0	4	2	0	2	0	0	0	0	0	2	4	2	0	0	0
8	0	0	0	4	0	4	0	0	0	2	0	2	0	2	0	2
9	0	0	0	4	0	0	0	4	2	0	2	0	2	0	2	0
A	0	0	2	0	0	2	2	2	2	0	0	0	2	2	0	2
B	0	4	2	0	2	0	0	0	0	0	2	0	2	4	0	0
C	0	0	0	0	0	0	0	0	4	0	4	0	4	0	4	0
D	0	0	0	4	4	0	4	4	0	0	0	0	0	0	0	0
E	0	0	2	0	0	2	2	2	2	2	0	2	2	0	0	0
F	0	4	2	0	2	0	0	0	2	4	0	0	0	0	2	0

Table 2: Difference Distribution Table of DoT

In table 2, trivial pair (0,0) appears 16 times and any non-trivial pair ($\triangle_i, \triangle_o$) appears 0/2/4 times. We use algorithm 3 to construct the differential trails and search for the trail covering maximum number of rounds with highest probability. It proceeds by dividing the 64-bit input data into sixteen 4-bit nibbles then it perform the trail search for all non-zero values of 4-bit nibble. The trails with least number of active S-boxes and highest probability are filtered out to get the optimal differential trails.

We constructed the 31-round differential distinguisher by extending the 2-round and 6-round trails that produce an output difference equals to the input difference after 2 and 6 rounds respectively. We start with input difference 0000000000000050 which results in the output difference 0000000000000008 after first round with probability 2^{-2}. This difference is the input difference for the second round which generates the difference 0000000000000050 after 2 rounds with probability 2^{-2}. In this way, we get 2-round trail with probability 2^{-4}. We also constructed several 6-round trails with equal values of input and output difference. We constructed total 37 distinct differential distinguihers for full-round DoT by extending these 2-round and 6-round differential trails. Some of the 2-round and 6-round trails are as shown below.

1. 0000000000000050 $\rightarrow$ 0000000000000008 $\rightarrow$ 0000000000000050
2. 0000000000000080 $\rightarrow$ 000000000000000c $\rightarrow$ 0000000000000080

3. 0050000000000000 → 0000000008000000 → 0000005000000000 → 0000000000080000
→ 0000000000005000 → 0800000000000000 → 0050000000000000
4. 0000000000800000 → 000000000000*c*00 → 8000000000000000 → 000*c*000000000000
→ 0000000080000000 → 0000000*c*00000000 → 0000000000800000

We get thirty seven differential trails that serves as full-round differential distinguisher for DoT encryption algorithm with probability equal to 2^{-62}. We compare the lower bound on the number of active S-boxes in any differential trail of DoT with the designers bound (Table 3). This shows that our results provide the significant improvements to the security claims of DoT. One of the full-round differential distinguisher is presented in table 4 with 0000000400000000 as first-round input difference and output difference after 31 rounds as 0000000000500000 with probability 2^{-62}.

Round Index	#Active S-boxes (Lower Bound)	# Active S-boxes (Lower Bound)
2	2	2
6	7	6
30	35	30
Ref.	[12]	This Paper

Table 3: Bounds on number of active S-boxes

Round index	Input Difference $\triangle P_i$	#Active S-box	Prob. $(-log_2 p_i)$	Prob. $(-\Sigma_i log_2 p_i)$
1	00000004 00000000	1	2	2
2	00000000 00500000	1	2	4
3	00000000 00000800	1	2	6
4	50000000 00000000	1	2	8
5	00080000 00000000	1	2	10
6	00000000 50000000	1	2	12
7	00000008 00000000	1	2	14
8	00000000 00500000	1	2	16
9	00000000 00000800	1	2	18
10	50000000 00000000	1	2	20
11	00080000 00000000	1	2	22
12	00000000 50000000	1	2	24
13	00000008 00000000	1	2	26
14	00000000 00500000	1	2	28
15	00000000 00000800	1	2	30
16	50000000 00000000	1	2	32
17	00080000 00000000	1	2	34
18	00000000 50000000	1	2	36
19	00000008 00000000	1	2	38
20	00000000 00500000	1	2	40
21	00000000 00000800	1	2	42
22	50000000 00000000	1	2	44
23	00080000 00000000	1	2	46
24	00000000 50000000	1	2	48
25	00000008 00000000	1	2	50
26	00000000 00500000	1	2	52
27	00000000 00000800	1	2	54
28	50000000 00000000	1	2	56
29	00080000 00000000	1	2	58
30	00000000 50000000	1	2	60
31	00000008 00000000	1	2	62
–	00000000 00500000	-	-	–

Table 4: Full-Round Differential Distinguisher

4 Analysis of Diffusion Layer

We propose an improved design by dropping the 8-P permutation operation of diffusion layer used in DoT and refer it as MDoT (Fig 3). Original version of DoT is referred

as DoT (Fig 1). We compare the differential attack resistance between the two versions of DoT block cipher. We apply our trail search algorithm on both versions of DoT and observe that MDoT provides the better security in comparison to the original DoT block cipher. We observe that any 10-round differential trail of MDoT have at least 10 active S-boxes with probability 2^{-25} while 10-round differential trail of original DoT have 10 active S-boxes with probability 2^{-20}. We have extended the 10-round differential trail of DoT to 31 rounds and presented a full-round differential distinguisher for original DoT. But, this is not possible in case of modified design MDoT. Any 30-round differential trail of MDoT cipher have 30 active S-boxes with probability 2^{-75}. We conclude that it is not possible to construct any full-round differential distinguisher for MDoT block cipher.

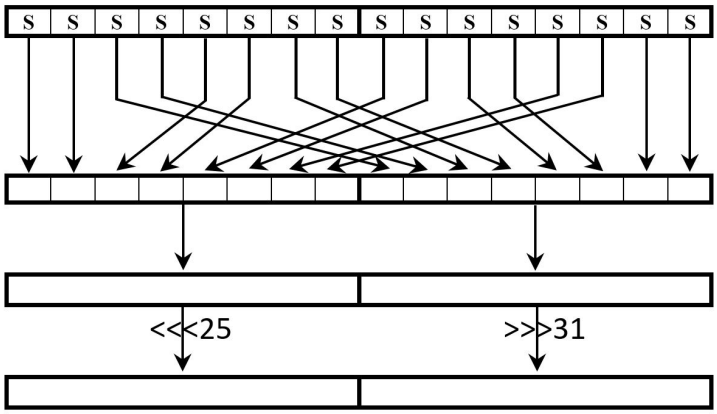

Figure 3: Round Function of MDoT

5 Conclusion

We constructed 31-round differential trails with probability equal to 2^{-62} and presented 37 differential distinguisher for full-round DoT. We can use 30-round differential trails with probability 2^{-60} for key recovery purpose. We also analysed the diffusion layer used in DoT and proposed an improved design named MDoT by removing the 8-P permutation. This analysis shows that MDoT provides the better resistance in comaprison to the original DoT proposal. Our analysis presents the best result for differential attack on DoT block cipher which suggest that 31-round DoT is prone to differential attack. We should either increase the number of rounds or use the modified design MDoT to resist the differential attack.

Acknowledgements

Author would like to thank Director SAG, Ms Anu Khosla for their support to work in this direction. Author is very much grateful to Ms. Pratibha Yadav and Dr. Sucheta Chakrabarti for their continuous guidance and encouragement to carry on this work.

References

[1] Biham, E., Shamir, A., Differential Cryptanalysis of the full 16-round DES, CRYPTO 92, LNCS, Vol. 740, pp. 487-496, Springer, 1992.

[2] Bogdanov, A., Analysis and Design of Block Cipher Constructions, Ph.D. thesis, 2009.

[3] Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C., PRESENT: An Ultra-Lightweight Block Cipher, CHES 2007, LNCS, Vol. 4727, pp. 450-466, Springer, 2007.

[4] Daemen, J., Rijmen, V., The Design of Rijndael, Springer-Verlag, 2002.

[5] Hays, H.M., A Tutorial on Linear and Differential Cryptanalysis, Cryptologia, Vol. 26, No. 3, pp. 188-221, 2002.

[6] Knudsen, L., Robshaw, M.J.B., Block Cipher Companion, Book Springer, ISBN 978-3-642-17341-7, 2011.

[7] Kumar, M., Design and Analysis of Symmetric Cryptographic Primitives, Ph.D. thesis, 2019.

[8] Kumar, M., Pal, S.K., Panigrahi, A., FeW : A Lightweight Block Cipher, Cryptology ePrint Archive, Report 2014/326, 2014.

[9] Kumar, M., Yadav, P., Pal, S.K., Panigrahi, A., Secure and Efficient Diffusion Layers for Block Ciphers, JACSM, 2017.

[10] Kumar, M., Suresh, T. S., Pal, S.K., Panigrahi, A., Optimal Differential Trails in Lightweight Block Ciphers ANU and PICO, Cryptologia, 2019.

[11] Matsui, M., On Correlation between the Order of S-boxes and the Strength of DES, EUROCRYPT 94, LNCS, Vol 950, pp. 366-375, Springer, 1994.

[12] Patil, J., Bansod, G., Kumar, S. K., DoT: A New Ultra-Lightweight SP Network Encryption Design for Resource-Constrained Environment, Advances in Intelligent Systems and Computing 828, pp. 249-257, 2019.

[13] Poschmann, A.Y., Lightweight Cryptography: Cryptographic Engineering for a Pervasive World, Ph.D. thesis, 2009.