J. Korea Soc. Math. Educ. Ser. B: Pure Appl. Math.
Volume 12, Number 2 (May 2005), Pages 143–152

ISSN 1226-0657

MATHEMATICAL ANALYSIS FOR A DYNAMIC CIPHER

YOON-TAE JUNG, EUN-HEE CHOI, AND KWANG-CHEOL RIM

ABSTRACT. We present a new block cipher called DyC. It consists of four sets (procedures) having the different 2^2, 2^2, 2^4, and 2^8 one-to-one correspondence functions as the elements. The round key is used to determine exactly one composite function from the possible 2^{16} composite functions. DyC supports $8 \times n$ bit key size, $16 \times m$ bit block length, and n rounds. We have confirmed that DyC offers security against other well-known advanced cryptanalytic attacks including the slide attacks and interpolation attacks. In this paper, we show several properties of the key schedule of DyC by mathematical analysis.

1. INTRODUCTION

At first, the design of DyC began with a consideration of the concept "key dependent". The term "key dependent" is not defined concretely but used in the several cryptographic primitives(*cf.* Biham [1], Biryukov & Wagner [2], Jung, Kang, Park & Cho [4]). The key schedule of DyC receives $8 \times n$ bit key as an input and outputs n-round keys. The key schedule uses 16 key bit for generating the round key. For each 16-bit round key, encryption of DyC consists of four sets having the different $2^2, 2^2, 2^4$, and 2^8 one-to-one correspondence functions as the elements. The round key of DyC is used to determine exactly one composite function out of the possible 2^{16} composite functions. DyC supports $8 \times m$-bit block length, $8 \times n$-bit key size, and n rounds for positive integers m and n. The security of DyC basically depends on the facts that (1) the key spaces to determine four kinds of one-to-one correspondence functions of four sets are $2^2, 2^2, 2^4$, and 2^8; (2) the key space to determine a composite function corresponding to a round is approximately 2^{16}; (3) the key space of DyC is increased by the number of rounds inductively. Block cipher based on the concept "key dependent" may reveal the weakness for Meet-in-the-Middle

Received by the editors November 30, 2004 and, in revised form, May 30, 2005.
2000 *Mathematics Subject Classification.* 68P25, 94A60.
Key words and phrases. block cipher, dynamic cipher, key dependent, linear cryptanalysis.
The first author was supported by Chosun University Research Funds 2001.

© 2005 Korea Soc. Math. Educ.

attack. The key schedule of DyC offers security against Meet-in-the-Middle attack. The attacks, successful in most Feistel ciphers, are not the case to DyC. We have confirmed that DyC has difficulty finding differential and linear characteristics(*cf.* Biryukov & Wagner [2], Jung, Kang, Park & Cho [4]). Encryption of DyC consists of only logical and bit operations that can be efficiently implemented in software, including the 8-bit processors used in low-end smart cards, 32-processors widely used in PCs, and 64-bit processors. The key schedule of DyC has remarkably short setup time with quite small memory requirements. In this paper, we present the mathematical analysis for one-to-one correspondence functions of DyC, and show several properties of the key schedule of DyC through mathematical analysis.

2. STRUCTURE OF DYC

We begin this section with a definition taking shape the framework for the block ciphers using the concept "key dependent" (*cf.* Jung, Kang, Park & Cho [4]). Here we assume that the key size is $8 \times n$ bit and the plaintext size is $16 \times m$ bit for positive integers m and n.

Definition 1. Dynamic cipher that encrypts a plaintext in the following way:

(1) The encryption consists of the different 2^{16n} one-to-one correspondence functions, each one-to-one correspondence function describes a method to map a given block to another block.

(2) The key schedule uses 16 key bit to generate the n round key, and each round key determines exactly one one-to-one correspondence function on the set of plaintexts.

Definition 2. Let $k_1k_2\cdots k_{8\times l}$ be a given key for $1 \le j \le l$. Then the key schedule of Dynamic cipher generates the 16-bit jth round key

$$k_{8\times(j-1)+1}k_{8\times(j-1)+2}\cdots k_{8\times(j-1)+8}k_{8\times(n-j)+1}k_{8\times(n-j)+2}\cdots k_{8\times(n-j)+8}$$

Also, Dynamic cipher supports various block lengths because the mapping method can be applied to various block lengths. As an example, bitwise XOR between left half and right half of the block can be applied to the blocks with even length.

2.1. Key schedule

The key schedule generates n round keys, and each round key uses 16 key bit derived from $8 \times n$ bit key. Let $k_1k_2\cdots k_{8\times n}$ be the key. The key schedule of DyC

outputs the i-th round key which consists of 10 subkeys

$$\{I_j^i | 1 \le i \le n, 1 \le j \le 10\}$$

where

$$\begin{aligned}
I_1^i &= (k_{8\times(i-1)+1} \quad k_{8\times(i-1)+2}),\\
I_2^i &= (k_{8\times(i-1)+3} \quad k_{8\times(i-1)+4}),\\
I_3^i &= (k_{8\times(i-1)+5} \quad k_{8\times(i-1)+6}),\\
I_4^i &= (k_{8\times(i-1)+7} \quad k_{8\times(i-1)+8}),\\
I_5^i &= (k_{8\times(n-i)+1} \quad k_{8\times(n-i)+2}),\\
I_6^i &= (k_{8\times(n-i)+3}),\\
I_7^i &= (k_{8\times(n-i)+4} \quad k_{8\times(n-i)+5}),\\
I_8^i &= (k_{8\times(n-i)+6}),\\
I_9^i &= (k_{8\times(n-i)+7}),\\
I_{10}^i &= (k_{8\times(n-i)+8}).
\end{aligned}$$

Example 2.1. If the given key is 01100111, then the key schedule generates 16-bit of the first round key 0110011101100111, and the first round key consists of 10 subkeys $I_1^1 = (01), I_2^1 = (10), \cdots, I_{10}^1 = (1)$. If the given key is 01100111000001111001100 11, the the key schedule generates three 16-bit round keys, *i. e.*, 0110011100110011, 0000111100001111, 0011001101100111. Hence $I_3^2 = (11)$.

2.2. Encryption and decryption

In this subsection, we consider $16 \times m$ bit plaintext B. We denote B by $(B_0||B_1||B_2||B_3)$, where B_i is the $2 \times m$-bit subblock for $i = 0, 1, 2, 3$ and $||$ means the catenation of subblocks B_i. The i-th round encryption of DyC consists of four sets (procedures) denoted by E^i, F^i, G^i and H^i. The functions of E^i are determined according to the first subkey $I_1^i = (k_{8\times(i-1)+1} \quad k_{8\times(i-1)+2})$ of i-th round key. We denote them by $e_{00}^i, e_{01}^i, e_{10}^i, e_{11}^i$. For the plaintext $B = (B_0||B_1||B_2||B_3)$, e_{st}^i complements all of the bits of $(2s+t)$-th subblock and the other bits remain. Therefore, E^i has four one-to-one correspondence functions, I_1^i specifies exactly one subblock, and each element of E^i complements $4 \times m$ bits out of $16 \times m$ bits.

Example 2.2. Let $B = 0000000000000000$ be the 16 bit plaintext and let $I_1^i = (10)$. Then $e_{10}^i(B) = 0000000011110000$.

The functions of F^i are determined according to the second subkey

$$I_2^i = (k_{8\times(i-1)+3} k_{8\times(i-1)+4})$$

of i-th round key. We denote them by f_{00}^i, f_{01}^i, f_{10}^i, f_{11}^i. For $B = (B_0||B_1||B_2||B_3)$ the functions of F^i swaps the $4 \times m$ pairs of bits as follows: If $I_1^i = (pq)$ and $I_2^i = (st)$, then the first bit of $(2p+q)$-th subblock of B is exchanged with the first bit of $(2s+t)$-th subblock of B, the second bit of $(2p+q)$-th subblock of B is exchanged with the second bit of $(2s+t+1 \mod 4)$-th subblock of B, 3rd bit of $(2p+q)$-th subblock of B is exchanged with 3rd bit of $(2s+t+2 \mod 4)$-th subblock of B, $\cdots$, and $(4m)$-th bit of $(2p+q)$-th subblock of B is exchanged with $(4m)$-th bit of $(2s+t+4m-1 \mod 4)$-th subblock of B.

Example 2.3. Let $B = 0000111101110011$, $I_1^i = (00)$, and $I_2^i = (00)$. Then, we can obtain

$$f_{00}^i(0000111101110011) = 0111101101010010.$$

If we apply the function e_{00}^i and f_{00}^i to B, then we have

$$f_{00}^i \circ e_{00}^i(B) = 1111111101110011.$$

The elements of F^i swaps $4 \times m$ pairs of bits for spreading complemented bits generated by the elements of E^i so that every subblock has approximately $(|B|/4)$ complemented bits.

Definition 3. For a bitstring $b_1 b_2 \cdots b_l$, it is S-XOR (Sequential eXclusive-OR) that does XOR generating a bitstring $b_1' b_2' \cdots b_l'$ in the following way: $b_2' = b_1 \oplus b_2, b_i' = b_i \oplus b_{i-1}'$ for $2 \le i \le l$, and $b_1' = b_1 \oplus b_l'$. S-XNOR uses XNOR instead of XOR in S-XOR, where XNOR means the negation of XOR.

The functions of G^i are determined according to the 3rd subkey $I_3^i = (k_{8\times(i-1)+5}$ $k_{8\times(i-1)+6})$ and the 4th subkey $I_4^i = (k_{8\times(i-1)+7}$ $k_{8\times(i-1)+8})$ of i-th round key. We denote them by $g_{0000}^i, g_{0001}^i, \cdots, g_{1111}^i$.

Each function g_{pqst}^i of G^i performs S-XOR or S-XNOR beginning with $(2p+q)$-th subblock and determines which operator in S-XOR and S-XNOR to apply and which direction of the operator to choose. We apply the function g_{pqst}^i as follows : If $(st) = (00)$, then g_{pqst}^i apply S-XOR to the first bit of $(2p+q)$-th subblock in the right direction. If $(st) = (01)$, then g_{pqst}^i apply S-XOR to the last bit of $(2p+q)$-th subblock in the left direction. If $(st) = (10)$, then g_{pqst}^i apply S-XNOR to the first

bit of $(2p+q)$-th subblock in the right direction. If $(st) = (11)$, then g^i_{pqst} apply S-XNOR to the last bit of $(2p+q)$-th subblock in the left direction.

Example 2.4. Let $B = 1000110011101111$. Then, $g^i_{0000}(B) = 1111011101001010$ and $g^i_{1011}(B) = 1101000100001111$.

The functions of H^i are determined according to the subkeys $I^i_5, I^i_6, I^i_7, I^i_8, I^i_9$ and I^i_{10}. The subkeys I^i_5, I^i_7 andI^i_9 determine the number of rotations. And the subkeys I^i_6, I^i_8 and I^i_{10} determine the direction of rotations and the bit- units of rotations. In fact, the functions of H^i are a kind of shifts. They are three kinds of shifts. We classify them as follows:

(1) If $I^i_5 = (pq)$ and $I^i_6 = (0)$, then we shift (and rotate) all bits by $(2p+q+1)$-bits in B in the right direction.
If $I^i_5 = (pq)$ and $I^i_6 = (1)$, then we shift all bits by $2p+q+1$-bits in B in the left direction.
We denote these kinds of shifts by $u^i_{000}, \cdots, u^i_{pqt}, \cdots, u^i_{111}$.

(2) If $I^i_7 = (pq)$ and $I^i_8 = (0)$, then we shift all bits by $(2p+q+1)$-bits within only $(B_0||B_1)$ in the right direction and shift all bits by $(2p+q+1)$-bits within only $(B_2||B_3)$ in the right direction. If $I^i_7 = (pq)$ and $I^i_8 = (1)$, then we shift all bits by $2p+q+1$- bits within only $(B_0||B_1)$ in the left direction and shift all bits $2p+q+1$- bits within only $(B_2||B_3)$ in the left direction.
We denote these kinds of shifts by $v^i_{000}, \cdots, v^i_{pqt}, \cdots, v^i_{111}$.

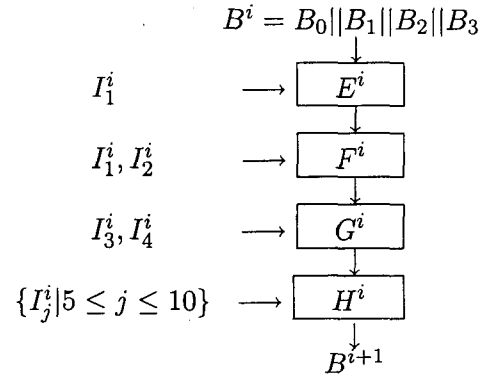
(3) If $I^i_9 = (p)$ and $I^i_{10} = (0)$, then we shift all bits by $p+1$-bits within only each B_i for $i = 0, 1, 2, 3$ in the right direction.
If $I^i_9 = (p)$ and $I^i_{10} = (1)$, then we shift all bits by $p+1$- bits within only each B_i for $i = 0, 1, 2, 3$ in the left direction. We denote these kinds of shifts by $w^i_{00}, \cdots, w^i_{pq}, \cdots, w^i_{11}$.

Hence the functions of H^i performs the rotation(or shift) in three methods.

Example 2.5. Let $B = 1000110011101111$. If the subkey $(I_5||I_6||I_7||I_8||I_9||I_{10})$ is 10101110, then we can consider functions u^i_{101}, v^i_{011} and w^i_{10}. First, since $I^i_5 = (10)$ and $I^i_6 = (1)$, we shift all bits in B by 3 bits in the left direction, so we get 0110011101111100. And, the second, since $I^i_7 = (01)$ and $I^i_8 = (1)$, we shift all bits only within $(B'_0||B'_1)$ by 2 bits in the left direction, and shift all bits only within $(B'_2||B'_3)$ by 2 bits in the left direction, where each B'_i is calculated in the first step. So we get 1001110111110001. Finally, third, since $I^i_9 = (1)$ and $I^i_{10} = (0)$, we shift

all bits by 2 bits only within each B_i'' for $i = 0, 1, 2, 3$ in the right direction, where each B_i'' is calculated in the second step. Therefore, we get 0110011111110100. Thus $w_{10}^i \circ v_{011}^i \circ u_{101}^i(B) = 0110011111110100$.

The following figure shows summarization of i-th round of Dyc.



DyC performs complement-swap-substitution-shift n times. Therefore, DyC is SP, iterated, and Dynamic cipher.

Following property is for the scalability of DyC.

Property 2.1. *DyC satisfies the followings:*

(1) *DyC supports* $16 \times m$ *bit block length and* $8 \times n$ *bit key size;*

(2) *For a given* $8 \times n$ *bit key, the number of rounds is fixed as* n.

Property 2.2 is for the property of one-to-one correspondence functions of DyC.

Property 2.2. *For the one-to-one correspondence functions, DyC satisfies the followings:*

(1) *Every function of DyC, specified by a round key, is one-to-one corresponding function.*

(2) *Any two one-to-one correspondence functions in each set are different one-to-one corresponding functions.*

From the property 2.2, each round of DyC can be considered as the composite function of four kinds of one-to-one correspondence functions specified by a round key. For the ith round block B^i, the (i+1)th round block B^{i+1} is an image of the composite function, and B^i is a preimage of B^{i+1} of the composite function. Therefore, we do not present decryption of DyC.

The design criteria of the encryption/decryption of DyC are the followings:

(1) The executing time should be short.
(2) The memory requirement should be quite small.
(3) Every one-to-one function should be simple.
(4) Plaintext length should be scalable.

3. Mathematical analysis for DyC

Theorem 3.1. *The functions of E^i, F^i, G^i and H^i can be represented by matrix forms.*

Proof. Since the functions of E^i, F^i, G^i and H^i are all linear functions with some constant terms, so they can be represented by matrix forms as follows :

$$\begin{aligned} e^i_{st}(X) &= e_i X + A_i, \\ f^i_{st}(X) &= f_i X, \\ g^i_{pqst}(X) &= g_i X + C_i, \\ u^i_{pqt}(X) &= u_i X, \\ v^i_{pqt}(X) &= v_i X, \\ w^i_{pq}(X) &= w_i X, \end{aligned}$$

where e_i, f_i, g_i, u_i, v_i, and w_i are matrix and A_i and C_i are vector forms. Since the given functions are bijective, the determinants of e_i, f_i, g_i, u_i, v_i, and w_i are not zero. □

Theorem 3.2. *The orders of the functions e^i_{pq} and f^i_{st} are 2's.*

Proof. Since the function e^i_{pq} is the complement on some subblock, the self composite of the given function is the identity function. And since the function f^i_{pq} is to exchange two given elements, the self composite of the given function is also the identity function. □

Proposition 3.3. *If the plaintext length is $m = 2^k$-bit for $1 \leq k \leq 16$, then the order of the functions g^i_{pqst} is $m^2 - 1$, respectively.*

Proof. We checked $(g^i_{pqst})^n$ the following in Remark 3.4 using Matlab program. For example, if the plaintext length 8-bit, then g^i_{0000} is of the form

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix},$$

which order is 63, etc. □

Remark 3.4.

(1) We computed the order of g^i_{pqst} using the following program :

```
function [k,A] = order (n)
E=zeros (n)
for i=1:n;
  E(i,j) =1;
end;
A=zeros(n);
for i=1:n;
    for j=1:n;
      A(i,j)=1;
    end;
end;
A(1,:) = ~ A(1,:) ;
B=A;
k=1;
for i=1:n;
 for j=1:n;
    while B(i,j)~ =E(i,j)
      B=mod (A*B,2);
      k=k+1;
    end;
  end;
end;
```

(2) We conjecture that if the plaintext length is $(m = 2^k)$-bits, then the order of the functions g^i_{pqst} is $m^2 - 1 = (2^k)^2 - 1$ for all k. We expect that more many facts will be checked using another mathematical program.

Remark 3.5.

(1) If the palintext X is of the length 8, 16, 32, 64-bit, respectively, then there is not a plaintext $X(\neq 0)$ such that $g^i_{pqst}(X) = g^i_{abcd}(X)$ for $(pqst) \neq (abcd)$,

$p, q, s, t, a, b, c, d = 0, 1$. We checked the eigenvalues of $g^i_{pqst} \circ (g^i_{abcd})^{-1}$ by Matlab program. For example, if the plaintext length 8-bit, then $g^i_{0000} \circ (g^i_{abcd})^{-1}$ does not have the eigenvalue 1.

(2) There are some nonzero plaintexts X_1, X_2, X_3, X_4, X_5 such that $e^i_{pq}(X_1) = e^i_{st}(X_1)$, $f^i_{pq}(X_2) = f^i_{st}(X_2)$, $u^i_{pqt}(X_3) = u^i_{abc}(X_3)$, $v^i_{pqt}(X_4) = v^i_{abc}(X_4)$ and $w^i_{pq}(X_5) = w^i_{st}(X_5)$, for $(pq) \neq (st), (pqt) \neq (abc)$. Using Matlab program we checked the eigenvalues of $e^i_{pq} \circ (e^i_{st})^{-1}$, $f^i_{pq} \circ (f^i_{st})(X_2)^{-1}$, $u^i_{pqt} \circ (u^i_{abc})^{-1}$, $v^i_{pqt} \circ (v^i_{abc})^{-1}$, and $w^i_{pq} \circ (w^i_{st})^{-1}$, all of which have eigenvalue 1's.

Theorem 3.6. *If the plaintext length is $8m$-bit, the order of functions u^i_{pqt}, v^i_{pqt}, and w^i_{pq} are, respectively,*

$$\frac{8m}{\gcd\{8m, 2p+q+1\}}, \quad \frac{4m}{\gcd\{4m, 2p+q+1\}}, \quad \textit{and} \quad \frac{2m}{\gcd\{2m, 2p+q+1\}},$$

where $\gcd\{a, b\}$ means the greatest common divisor of a and b.

Proof. If the plaintext length is $8m$-bit, the associated matrix u^i_{000} is $8m \times 8m$- matrix obtained by shifting consecutively the row vectors of $8m \times 8m$-identity matrix, that is, the first row of identity matrix is shifted to the second row, ..., the last row of identity matrix is shifted to the first row. Hence $(u^i_{000})^{8m} =$ identity matrix. Since $u^i_{pq0} = (u^i_{000})^{2p+q+1}$, our result holds. For the case u^i_{pq1}, similarly, our result holds. And the associated matrix v^i_{000} is of the form $\begin{pmatrix} A & 0 \\ 0 & A \end{pmatrix}$, where A is $4m \times 4m$ - matrix obtained by shifting consecutively the row vectors of $4m \times 4m$-identity matrix, that is, the first row of identity matrix is shifted to the second row, ..., the last row of identity matrix is shifted to the first row. Hence $(v^i_{000})^{4m} =$ identity matrix. Since $v^i_{pq0} = (v^i_{000})^{2p+q+1}$, our result holds. It is similar with the case v^i_{pq1}. And the associated matrix w^i_{00} is of the form

$$\begin{pmatrix} B & 0 & 0 & 0 \\ 0 & B & 0 & 0 \\ 0 & 0 & B & 0 \\ 0 & 0 & 0 & B \end{pmatrix},$$

where B is $2m \times 2m$-matrix obtained by shifting consecutively the row vectors of $2m \times 2m$-identity matrix, that is, the first row of identity matrix is shifted to the second row, $\cdots$, the last row of identity matrix is shifted to the first row. Hence $w_i^{2m} =$ identity matrix. Since $w^i_{p0} = (w^i_{00})^{p+1}$, our result holds. It is similar with the case w^i_{p1}. □

REFERENCES

1. E. Biham: New types of cryptanalytic attacks using related keys. *J. Cryptology* **7** (1994), no. 4, 229–246. CS **0812.**94012MA
2. A. Biryukov & D. Wagner: Advanced Slide Attacks. *Advances in Cryptology-EUROCRYPT 2000, Lecture Notes in Computer Science 1807* (pp. 589–606), Springer-Verlag, Berlin, 2000. CS **0939.**00052MA
3. T. Jakobsen & L. R. Knuden: The Interpolation Attack on Block Ciphers. *Fast Software Encryption 97, FSE'97, Lecture Notes in Computer Science 1267* (pp. 28–40), Springer-Verlag, Berlin, 1997. CS **0901.**68004MA
4. Y. Jung, S. Kang, S. Park & E. Choi: Analysis for a Block Cipher Based on a New Framework *Proceedings of the ICIM'01, 2001 International Conference on Integration of Multimedia Comtents* (pp. 167–171), Chosun University, 2001.

(Y.-T. JUNG) DEPARTMENT OF MATHEMATICS, CHOSUN UNIVERSITY, 375 SEOSEOK-DONG, DONG-GU, KWANGJU 501-759, KOREA
Email address: ytajung@chosun.ac.kr

(E.-H. CHOI) DEPARTMENT OF MATHEMATICS, CHOSUN UNIVERSITY, 375 SEOSEOK-DONG, DONG-GU, KWANGJU 501-759, KOREA
Email address: euro1@hanmir.com

(K.-C. RIM) DEPARTMENT OF MATHEMATICS, CHOSUN UNIVERSITY, 375 SEOSEOK-DONG, DONG-GU, KWANGJU 501-759, KOREA
Email address: rim1201@hanmail.net