

# **A NEW TYPE OF CIPHER: DICING\_CS**

Li An-Ping

Beijing 100085, P.R.China

apli0001@sina.com

**Abstract:** In this paper, we will propose a new type of cipher named DICING\_CS, which come from our previous a synchronous stream cipher DICING. It applies a stream of subkeys and a encryption form of block ciphers, so, it can be viewed a combinative of stream cipher and block cipher. Hence, the new type of cipher has fast speed like a stream cipher and no need MAC.

**Keywords:** stream cipher, block cipher, LFSR, projector, finite field.

## 1. Introduction

In a synchronous stream cipher, the ciphertext is generally made by bitwise adding (XOR) the plaintext with a binary sequence called keystream. Clearly, in this encryption form the plaintext is easy to be falsified by other people. As a result, a synchronous stream cipher usually is equipped a MAC ( message authentication code ) to protect the message from to be tampered. In our algorithm DICING [1], one of candidates of eSTREAM ( The ECRYPT STREAM Ciphers Project ), the combining function had mainly applied keyed-Sboxes, which are often used in the block ciphers, we realize that it is possible to make a combinative of stream cipher and block cipher (CSB mode), so will be able to omit MAC in this way. In the proposal cipher, the component  $u_t$  of DICING will be applied as a role of a stream of subkeys, and the encryption means are mainly keyed-Sboxes, like one ordinary block cipher. The components are almost same to the ones in DICING, for the completeness, which are repeated in this paper.

In the proposal cipher, we will apply the LFSR-like components called projector (Pr.). A projector consists of an element  $\sigma_t$  called state from some finite field  $GF(2^m)$  and an updating rule. The rule of updating states is that multiplying  $\sigma_t$  with  $x^k$ ,  $k$  is an integer, namely,

$$\sigma_{t+1} = x^k \cdot \sigma_t. \quad (1.1)$$

The finite fields used in here are  $GF(2^m)$ ,  $m = 128, 127, \text{ or } 126$ . In other word, the operation *shift* in LFSR now is replaced by multiplying  $x^k$  in the field  $GF(2^m)$ .

As same as DICING, the key sizes in DICING-CSB can be 128 bits or 256 bits, and the size of initial value may be taken as large as 256 bits, and the size of output is 128 bits.

In this paper the finite field  $GF(2)$  is simply denoted as  $\mathbb{F}$ , and  $\mathbb{F}[x]$  is the polynomial ring of unknown  $x$  over the field  $\mathbb{F}$ . The symbols  $\oplus$ ,  $\otimes$  will represent the bitwise addition *XOR*, bitwise *and*, that is the operation  $\&$  in  $C$ , and symbols  $\gg$ ,  $\ll$ ,  $|$  and  $\sim$  stand for the operations *right-shift*, *left-shift*, *concatenate* and *complement* respectively.

Suppose that  $\zeta$  is a binary string, denoted by  $\zeta[i]_{bit}$  and  $\zeta[i, j]_{bit}$  the  $i$ -th bit and the segment from  $i$ -th bit to  $j$ -th bit respectively, and there are the similar expressions  $\zeta[i]_{byte}$ ,  $\zeta[i, j]_{byte}$  and  $\zeta[i]_{word}$ ,  $\zeta[i, j]_{word}$  measured in bytes and 32-bits words respectively, and if the meaning is explicit from the context, the low-index *bit*, *byte* and *word* will be omitted.

## 2. Construction

We will use two projectors  $\Gamma_1$  and  $\Gamma_2$ , the first one acts a controller to control the updating of the second one, which will be used to form a stream of subkeys, or runkeys.

Denoted by  $\alpha_i$  and  $\omega_i$  the states of  $\Gamma_1$  and  $\Gamma_2$  in time  $t$  respectively, which are based on the finite fields  $\mathbb{E}_1$  and  $\mathbb{E}_2$ ,  $\mathbb{E}_i = \mathbb{F}[x]/p_i(x)$ ,  $i = 1, 2$ ,  $p_1(x)$  and  $p_2(x)$  are two primitive polynomials with degree 127 and 128 respectively, which expressions are given in the List 1. The state  $\alpha_i$  satisfy the simple recurrence equation

$$\alpha_{i+1} = x^8 \cdot \alpha_i, \quad i = 0, 1, 2, \dots \quad (2.1)$$

The integer of the last eight bits of  $\alpha_i$  is called the dice  $D_i$ , denoted by  $d = (D_i \gg 4) + 1$ ,

the states  $\omega_i$  will be updated as

$$\omega_{i+1} = x^d \cdot \omega_i, \quad \text{for } t \geq 0. \quad (2.2)$$

Besides, we use a memorizes  $u_i$  to assemble  $\omega_i$ ,

$$u_i = u_{i-1} \oplus \omega_i, \quad \text{for } t > 0, \quad (2.3)$$

The initial values  $\alpha_0, \omega_0$ , and  $u_0$  will be specified in the later.

Suppose that  $\mathbb{K}$  is a finite field  $GF(2^8)$ ,  $\mathbb{K} = \mathbb{F}[x]/p(x)$ ,  $p(x)$  is an irreducible polynomial of degree eight, which expression is given in the List 1. We define S-box  $S_0(x)$  as

$$S_0(x) = 5 \cdot (x \oplus 3)^{127}, \quad x \in \mathbb{K}. \quad (2.4)$$

We also adopt the representation  $S_0(\zeta)$  for a bytes string  $\zeta$  to represent that S-box  $S_0$  substitute each byte of the string  $\zeta$ .

The startup includes two subprocesses *keysetup* and *ivsetup*, where the basic materials as the secret key and key-size will be input and the internal states will be initialized. Besides, in the *keysetup* we will make a key-defined two S-boxes  $S_1(x)$  and  $S_2(x)$  from  $S_0(x)$  and a diffusion transformation  $L$ . The process is as following.

For a string  $\rho$  of 8 bytes, we define a  $8 \times 8$  matrix  $M_\rho$ :

$$M_\rho = T_u \cdot J \cdot T_l. \quad (2.5)$$

where  $T_u = (a_{i,j})_{8 \times 8}$  and  $T_l = (b_{i,j})_{8 \times 8}$  are the upper-triangular matrix and the lower-triangular matrix respectively,

$$a_{i,j} = \begin{cases} \rho[8i+j]_{bit} & \text{if } i < j, \\ 1 & \text{if } i = j, \\ 0 & \text{if } i > j, \end{cases} \quad b_{i,j} = \begin{cases} \rho[8i+j]_{bit} & \text{if } i > j, \\ 1 & \text{if } i = j, \\ 0 & \text{if } i < j, \end{cases} \quad (2.6)$$

and  $J$  is a key-defined permutation matrix, for the simplicity, here take  $J = 1$ .

Suppose that  $K$  is the secret key, let  $K_c = K[0,23]_{byte} \oplus K[8,31]_{byte}$  if  $|K| = 256$ , else

$K_c = K[0,15] \mid (K[0,7] \oplus K[8,15])$ ,  $\lambda_i = K_c[(i-1) \times 8, 8i-1]_{byte}$ ,  $i = 1, 2, 3$ , and define three affine transformations on  $\mathbb{K}$ ,

$$A(x) = M_{\lambda_1}(x), \quad B(x) = M_{\lambda_2}(x), \quad C(x) = M_{\lambda_3}(x), \quad x \in \mathbb{K}, \quad (2.7)$$

and a transformation  $L$  on  $\mathbb{K}^4$ ,

$$L = \begin{pmatrix} A & B & A & A \oplus B \\ B & A & A \oplus B & A \\ A & A \oplus B & A & B \\ A \oplus B & A & B & A \end{pmatrix}. \quad (2.8)$$

Denoted by  $v_i = \bigoplus_{0 \leq k < 8} \lambda_i[k]_{byte}$ ,  $i = 1, 2, 3$ , and define two new S-boxes

$$S_1(x) = S_0(x \oplus v_1) \oplus v_2, \quad S_2(x) = C(S_0(x \oplus v_2) \oplus v_3), \quad x \in \mathbb{K}. \quad (2.9)$$

Suppose that  $\zeta$  is a string of  $n$  bytes, if  $n = 4k$  we also view it as a string of  $k$  words, and

write  $L(\zeta)$  to represent that  $L$  takes on the each word of  $\zeta$ . Simply, we denote

$$Q(\zeta) = L \cdot S_1(\zeta). \quad (2.10)$$

In the *ivsetup*, the second step of the startup, the internal states will be initialized with the secret key and the initial value.

For a 32-bytes string  $\zeta$  we define a bytes permutation  $\phi: \zeta^\phi = \phi(\zeta)$ ,  $\zeta^\phi[i] = \zeta[4i \bmod 31]$ ,

for  $0 \leq i < 31$ , and  $\zeta^\phi[31] = \zeta[31]$ . Let  $\tilde{K} = K$  if  $|K| = 256$  else  $\tilde{K} = K \mid (\sim K)$ , denoted

by  $\tilde{K}_0 = \tilde{K}$ ,  $\tilde{K}_i = \tilde{K}[8i, 31]_{byte} \mid \tilde{K}[0, 8i-1]_{byte}$ ,  $i = 1, 2, 3$ , define the functions recurrently

$$F(\zeta) = Q(\phi(\zeta)), \quad F_0(\zeta) = F(\zeta) \oplus \tilde{K}_0, \quad F_i(\zeta) = F(F_{i-1}(\zeta)) \oplus \tilde{K}_i, \quad i = 1, 2, 3. \quad (2.11)$$

Suppose that  $IV$  is the initial value of 32-bytes,  $e$  is the base of natural logarithm and  $c$  the integral part of  $e \cdot 57!$ , and  $\xi_i$ ,  $0 \leq i \leq 3$ , are four 32-bytes strings defined as

$$\xi_0 = F_3(IV \oplus c), \quad \xi_i = F_3(\xi_{i-1} \oplus c), \quad i = 1, 2. \quad (2.12)$$

In the encryption we will employ an array of 16 bytes  $\eta$ . The internal states are initialized respectively as following

$$\eta = \xi_0[0,15] \oplus \xi_0[16,31], \quad u_0 = \xi_1[0,15], \quad \alpha_0 = \xi_1[128,254]_{bit}, \quad \omega_0 = \xi_2[0,15]. \quad (2.13)$$

If  $\xi_2[0,15] = 0$ , the states  $\omega_0$  will be re-set as

$$(\omega_0, \tau_0) = \xi_2[16,31]. \quad (2.14)$$

*Note.* For a secret key, there is at most one  $IV$  such that  $\xi_2 = 0$ .

In the proposal cipher DICING\_CSB, the sequence  $\{u_t\}$  will play a flow of subkeys. After initialization, the process enters the recurrence part of encryption/decryption, in which including the sub-process of updating states, namely, making the stream of subkeys  $\{u_t\}$ . Denoted by  $\{x_t\}_{t>0}$  and  $\{y_t\}_{t>0}$  the sequences of plaintext and ciphertext respectively, the encryption function is defined as

$$y_t = \text{Encrypt}(x_t) = S_2(Q(x_t \oplus u_t) \oplus Q(\eta)) \oplus u_t. \quad (2.15)$$

We have summarized the whole process in a sketch as Fig. 1.

#### List of the Primitive Polynomials used

Polynomials	Expression
$p(x)$	$x^8 + x^6 + x^5 + x + 1$
$p_1(x)$	$x^{127} + (x^{96} + x^{64} + 1)(x^5 + 1)$
$p_2(x)$	$x^{128} + (x^{96} + x^{67} + x^{32} + 1)(x^3 + 1)$

List 1

#### The Sketch of Encryption Process

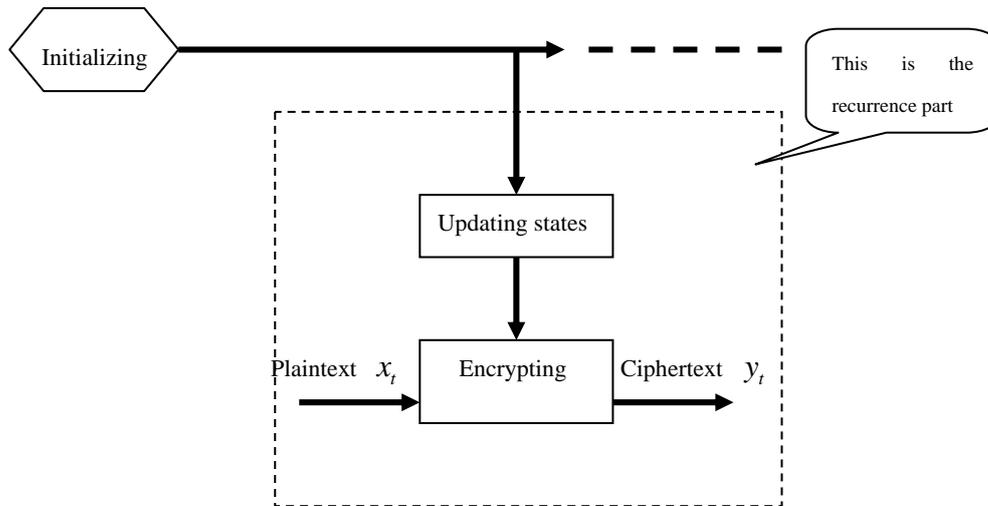


Fig.1

### 3. Security analysis

The analysis for DICING\_CSB as a stream cipher will be similar to the one for DICING, refer to see paper [1]. Besides, as a block cipher, the encryption mode of DICING\_CSB is not as usual iterative one, so the traditional analyses for the block ciphers of iterative mode will not be feasible. Although the proposal cipher is stronger than additive stream ciphers in plaintext-recovery attacks, we have found there is such attack for DICING\_CSB, if a IV's value would be repeatedly used more than  $2^{16}$  times for a secret key  $K$ . So, it is suggested that the usage of a  $IV$  is best one time one value. If intend to apply a  $IV$  many times, then in encryption function should be added a more round in order to enlarge the range of diffusion, as a cost, the encrypting rate will be raised about 2 cycles/byte.

It maybe should be mentioned that we have reduced two Pr.'s from DICING for we think that in this encryption form the requirement for the period of the sequence  $\{u_i\}$  may be relaxed, in this place, the period of the sequence  $\{u_i\}$  is no less than  $(17 \cdot 2^{126} - 1)(2^{128} - 1)$ .

### 4. Implementation

In the platform of 32-bit Windows OS and Intel ® Celeron 2.66G, 64-bit processor, Borland C++ 5.0, the performance of DICING\_CSB is as following

## Report of Performance

Encryption		Decryption	
Sub-processes	Time	Sub-processes	Time
Keysetup	9890 cycles	Keysetup	16400 cycles
IVsetup	2870 cycles	IVsetup	2920 cycles
Encrypting rate	10.3 cycles/byte	Decrypting rate	10.3 cycles/byte

### List 2

*Remark:* There is an alternate updating rule for the states  $\alpha_t$  and  $\omega_t$  as following:

$$\alpha_{t+1} = x^{16} \cdot \alpha_t, \quad \text{for } t > 0. \quad (2.1')$$

Denoted by  $d_i = 1 + (\alpha_{t+1}[i]_{\text{byte}} \& 15)$ ,  $0 \leq i < 16$ , the states  $\omega_t$  are updated as

$$\omega_{16t+i+1} = x^{d_i} \cdot \omega_{16t+i}, \quad 0 \leq i < 16, \quad \text{for } t \geq 0. \quad (2.2')$$

With the updating rules above, the encrypting/decrypting rate will be as fast as *8.8 cycles/byte* in the case of larger size of message. We call the rule (2.1') and (2.2') as *loting*.

Besides, with the processors as Pentium-m, Pentium 4 or AMD-64, the implementation will be faster about 20~30%.

## 5. Conclusion

The proposal cipher can be viewed as a combinative of a stream cipher and a block cipher. It assimilates the good qualities of stream ciphers in the speed and block ciphers in the secure. It is able to serve as a synchronous stream cipher or a block cipher, and there will not be need to equip a MAC when it is applied as a synchronous stream cipher. While it is applied as block cipher, it will still require a IV to initialize the internal states, however, this requirement is easy to be simply satisfied, for example, the name or the date of files may be taken as the IV' values.

## References

- [1] A.P. Li, A New Stream Cipher: DICING, now available at [eSTREAM - The ECRYPT Stream Cipher Project - Phase 2](#)