A NEW STREAM CIPHER: DICING

Li An-Ping

Beijing 100085, P.R.China
apli0001@sina.com

Abstract: In this paper, we will propose a new synchronous stream cipher named DICING, which can be taken as a clock-controlled one but with a new mechanism of altering steps. With the simple construction, DICING has satisfactory performance, faster than AES about two times. For the security, there have not been found weakness for the known attacks, the key sizes can be 128 bits and 256 bits respectively.

Keywords: stream cipher, LFSR, projector, finite field, correlation attack, algebraic attack, distinguishing attack.

1. Introduction

In a synchronous stream cipher, the ciphertext is generally made by bitwise adding (XOR) the plaintext with a binary sequence called keystream. Hence the principle task for a synchronous stream cipher is to produce a secure keystream. For the real cases that there is the possibility that the cipher is abused or the plaintext of some ciphertext are known by some people, thus the keystream will become visible for them, the analysis for this case is called the plaintext-known analysis. Therefore, a secure keystream should satisfy two basic conditions: The first is that the original key can not be recovered from the keystream, the second is that the contents of the bits in the keystream should be unpredictable for an adversary, in other words, for the adversaries the keystream should look like a random one, i.e. pseudo-random. Clearly, if the keystream sequence is periodic and the period is short, then it will be predictable, thus the keystream should have enough large period. It is known that the technique of the linear feedback shift registers (LFSR) is able to generate the larger periods of sequences, so LFSRs are often used in the stream ciphers as the component parts. However, LFSR also has an obvious weakness that the each bit in a LFSR's sequence is linearly related to the initial state, and so this implies that the initial state is easy deduced from some of the later bits in the sequence, the famous Berlekamp-Massey's algorithm [18,21] is such a example of the algorithms. In the almost of known attacks such as correlation attacks, algebraic attacks and distinguishing attacks, etc. just exploited the weakness of LFSR. So, LFSR-based stream ciphers should interfere the linear relations in the bits of the LFSRs. Clearly, the clock-controlled methods comes from this consideration, see [12,21].

On the other hand, in respect to the implementation, the operation of the ordinary LFSRs is in bits, that is, is based on the finite field $GF(2)$, not convenient for software implement, so the later stream ciphers such as BLUETOOTH [1], SOBER [14,15], SNOW [7] and SCREAM [13] have been developed to the finite field $GF(2^k),\ k = 8, 16,\ or\ 32$.

In this paper, we will present a new synchronous stream cipher called DICING. It satisfies the security requirement, with a simple construct and has a satisfactory performance.
The cipher DICING may be taken as a clock-controlled one, but with a new mechanism of altering steps. It consists of a controller and a combiner. In the proposal cipher, we will substitute the LFSR for the LFSR-*like* called projector (Pr.). A projector consists of an element σ_t called state from some finite field $GF(2^m)$ and an updating rule. The rule of updating states is that multiplying σ_t with x^k, k is an integer, namely,

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

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

The key sizes in DICING can be 128 bits or 256 bits, and the size of initial value is assumed same as the key size, and the size of output of DICING is 128 bits.

In the section 2, we will give a detail description for the construction, the section 3 is security analysis, the section 4 is a report of software implementation, the section 5 are some considerations in the design, in the section 6 are some possible variances for the proposal cipher.

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 >>, <<, $|$ and ~ stand for the operations *right-shift, left-shift, concatenate* and *complement* respectively

Suppose that ζ 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 words respectively, and if the meaning is explicit from the context, the low-index *bit, byte* and *word* will be omitted. In this paper, the size of a word is 32 bits.

2. Construction

As the general stream ciphers, the proposal cipher is also encrypt the plaintext and decrypt the ciphertext by adding bitwise a binary string called keystream, namely,

$$Ciphertext = Plain\,text \oplus Keystream \qquad (2.1)$$

The keystream generator contains two main parts, a controller **H** and a combiner **C**. The controller **H** is made from two projectors Γ_1, Γ_2 and two counters D'_t, D''_t which are also called dices.

Denoted by α_t and β_t the states of Γ_1 and Γ_2 in time t respectively, which come from the finite fields $\mathbb{E}_1$ and $\mathbb{E}_2$ respectively, $\mathbb{E}_1 = \mathbb{F}[x]/p_1(x)$ and $\mathbb{E}_2 = \mathbb{F}[x]/p_2(x)$, $p_1(x)$ and $p_2(x)$ are the primitive polynomials with degree 127 and 126 respectively, which expression are given in the List 1. They satisfy the simple recurrence equations

$$\alpha_{i+1} = x^8 \cdot \alpha_i, \quad \beta_{i+1} = x^8 \cdot \beta_i, \qquad i = 0, 1, 2, \ldots. \qquad (2.2)$$

In other words, α_t and β_t move a byte in a cycling.

The dices D_t' and D_t'' are two integers to record the last eight bits of α_t and β_t respectively. The combiner **C** also contains two projectors Γ_3 and Γ_4, which are based on the two finite fields $\mathbb{E}_3$ and $\mathbb{E}_4$ respectively, $\mathbb{E}_3 = \mathbb{F}[x]/p_3(x)$, $\mathbb{E}_4 = \mathbb{F}[x]/p_4(x)$, $p_3(x)$ and $p_4(x)$ are primitive polynomials of degree 128 given in the List 1. Denoted by ω_t and τ_t the states of Γ_3 and Γ_4 in the time t respectively, $\omega_t \in \mathbb{E}_3$ and $\tau_t \in \mathbb{E}_4$. Denoted by $D_t = (D_t' \oplus D_t'')$, $a = D_t \,\&15$, $b = (D_t) >> 4$, then

$$\omega_{t+1} = x^a \cdot \omega_t \ , \quad \tau_{t+1} = x^b \cdot \tau_t \, . \tag{2.3}$$

Besides, we use two memorizes u_t and v_t to assemble ω_t and τ_t respectively,

$$u_t = u_{t-1} \oplus \omega_t, \quad v_t = v_{t-1} \oplus \tau_t, \qquad for\ t > 0\,, \tag{2.4}$$

where $u_{-1} = v_{-1} = 0$.

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

$$S_0(x) = 5 \cdot (x \oplus 3)^{127}, \quad x \in \mathbb{K}\,. \tag{2.5}$$

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

The initialization process is as following. Suppose that K is the secret key, and IV is the initial value. Let

$$K_I = K \oplus IV, \qquad K' = \begin{cases} K_I & if\ |K| = 256 \\ K_I \,|\, (\sim K_I) & if\ |K| = 128, \end{cases} \tag{2.6}$$

Let c_i be the constants, which are equal to the integer part of the logarithm of the *i*-th prime number with multiplying properly $2^w, w = 32,\ 31\ or\ 30, 1 \le i \le 8$, and $c = (c_1, c_2, \ldots, c_8)$. Define

$$K_{ICS} = S_0(K' \oplus c)\,. \tag{2.7}$$

$$\alpha_0 = K_{ICS}[0,126]_{bit}, \ \beta_0 = K_{ICS}[128,253]_{bit}. \tag{2.8}$$

Moreover, denoted by $s = \bigoplus_{0 \le i < 32} K_{ICS}[i]$, let σ be a bytes string of length 32, $\sigma[i] = s, \ 0 \le i < 32$. Define

$$K_{II} = S_0(K_{ICS} \oplus \sigma \oplus (\sim c)), \tag{2.9}$$

and

$$\omega_0 = K_{II}[0,127]_{bit}, \quad \tau_0 = K_{II}[128,255]_{bit}. \tag{2.10}$$

Follow the initializing, there is a self-cycling sub-process of 64 loops, in which only the states are updated but without outputs. In this sub process, the states of the first 32 cycles are used to make up two affine transformations A and B, which are used to renew the S-box and set up a mixed transformation L_1 on $\mathbb{K}^4$ respectively. The procedure is in the below: For a string ρ of 8 bytes, we define a vector V_ρ of 8 bits and a 8×8 matrix M_ρ: $V_\rho[i] = \rho[8i+i]_{bit}, 0 \le i < 8,$ and

$$M_\rho = T_u \cdot T_l. \tag{2.11}$$

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,

$$a_{i,j} = \begin{cases} \rho[8i+j]_{bit} & if \ i < j, \\ 1 & if \ i = j, \\ 0 & if \ i > j, \end{cases} \qquad b_{i,j} = \begin{cases} \rho[8i+j]_{bit} & if \ i > j, \\ 1 & if \ i = j, \\ 0 & if \ i < j. \end{cases}$$

Let λ_i be the strings of length 16 bytes, which is iteratively defined as

$$\lambda_0 = \omega_0 \oplus \tau_0, \quad \lambda_i = \lambda_{i-1} \otimes (\alpha_i \oplus \beta_i) \oplus (\omega_i \oplus \tau_i), \quad 1 \le i \le 32. \tag{2.12}$$

Denoted by $\lambda' = \lambda_{32}[0,7]_{byte}, \lambda'' = \lambda_{32}[8,15]_{byte}$, and define two affine transformations

$$A(x) = M_{\lambda'}(x) \oplus V_{\lambda'}, \quad B(x) = M_{\lambda''} \cdot M_{\lambda'}^{-1}(x \oplus V_{\lambda'}) \oplus V_{\lambda''}, \quad x \in \mathbb{F}^8. \tag{2.13}$$

In the 32-th loop of the self-cycling, we define a new S-box $S(x)$ and a transformation L_1 on $\mathbb{K}^4$,

$$S(x) = A \cdot S_0(x), \qquad L_1 = \begin{pmatrix} 1 & B & 1 & 1\oplus B \\ B & 1 & 1\oplus B & 1 \\ 1 & 1\oplus B & 1 & B \\ 1\oplus B & 1 & B & 1 \end{pmatrix}. \tag{2.14}$$

Suppose that ζ is an array of 16 bytes, then ζ can be arranged as a 4×4 matrix, $\zeta = (a_{i,j})_{4\times 4}$, $a_{i,j} = \zeta[4i+j]_{byte}$, $0 \le i, j < 4$. So, denoted by $L_1(\zeta)$ the transformation L_1 takes on the each row of ζ. Moreover, let $R(\zeta)$ be the rotation operation, which rotates the j-*th* column j bytes, i.e. $R(\zeta[4i+j]) = \zeta[4i'+j], i' = i + j \bmod(4)$, $0 \le i, j < 4$.

After self-cycling, the process enters the recurrence part of generating keystream, each cycle includes two sub-processes of updating and combining. In the updating, all the states are updated from the time $t-1$ to the time t. Let $\gamma_0 = \omega_{32} \oplus \tau_{32}$, the combining functions $C_0(\zeta)$, $C(\zeta, \vartheta)$ are defined as

$$C_0(\zeta) = L_1(S(\zeta)), \qquad C(\zeta,\vartheta) = C_0(R(C_0(\zeta)\oplus\vartheta))\oplus\gamma_0 . \tag{2.15}$$

where ϑ and ζ are two arrays of length 16 bytes. The combing process is controlled by the case $D'_t : D''_t$, which is formulized as in the following, z_t is denoted as the keystream,

$$z_{t+1} = \begin{cases} C(u_{t+1}, v_{t+1}), & if\ D'_t > D''_t, \\ C(v_{t+1}, u_{t+1}), & if\ D'_t < D''_t, \\ \varnothing, & if\ D'_t = D''_t, \end{cases} \tag{2.16}$$

where symbol $\varnothing$ represents no keystream out. We have summarized the whole process in a sketch as Fig. 1.

List of the Primitive Polynomials used

Polynomials	Expression
$p(x)$	$x^8 + x^6 + x^5 + x + 1$
$p_1(x)$	$x^{127} + (x^{89} + x^{41} + 1)(x^3 + 1)$
$p_2(x)$	$x^{126} + (x^{83} + x^{35} + 1)(x^7 + 1)$
$p_3(x)$	$x^{128} + (x^{90} + x^{57} + x^{35} + 1)(x^7 + x^2 + 1)$
$p_4(x)$	$x^{128} + (x^{97} + x^{58} + x^{27} + 1)(x^7 + x + 1)$

List 1

The Sketch of Encryption Process

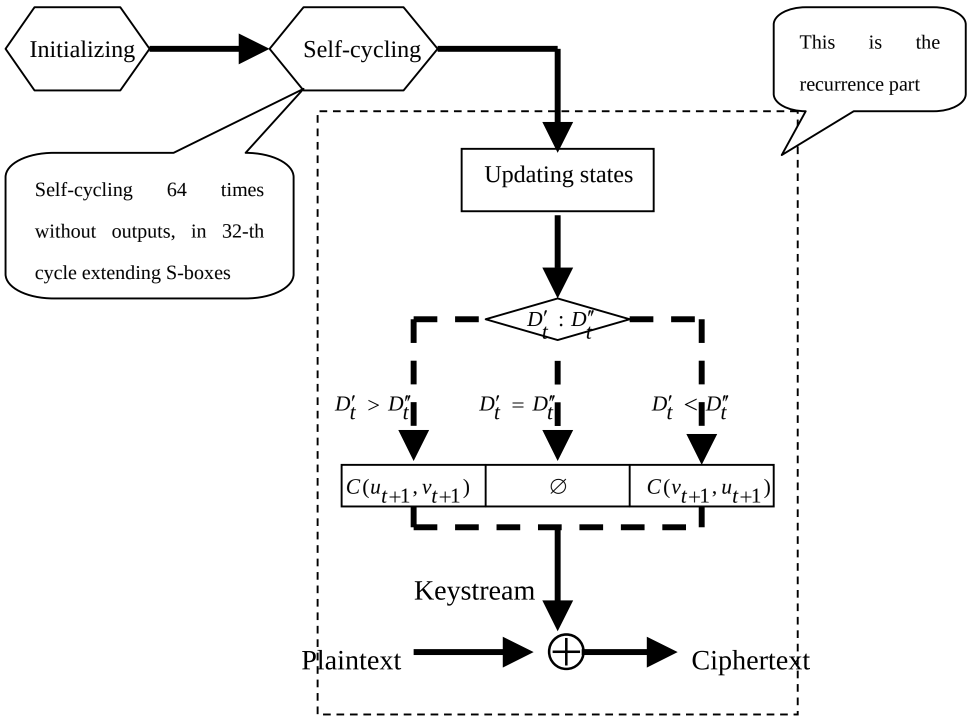

Fig.1

3. Security Analysis

In the design, we always take the security as the aspect of the first important. In the beginning of this section, we will show some results about the periods and distributions for proposal stream cipher, and then give an investigation with respect to standard cryptanalytic attacks, finally provide some results of statistic tests.

Period and Distribution

Clearly, the periodicity of a sequence implies it is predictable after a period, so with a larger period is a requisite for the keystream. Denoted by $Period(\pi_t)$ as the period of a sequence π_t.

Proposition 1

$$Period\,(D_t) = (2^{126}-1)(2^{127}-1), \tag{3.1}$$

$$Period\,(\omega_t) = Period\,(\tau_t) = (2^{126}-1)(2^{127}-1)(2^{128}-1)/15 \tag{3.2}$$

Proof. Note that polynomials $p_1(x)$ and $p_2(x)$ are primitive, and the order of x in the fields $\mathbb{E}_1$ and $\mathbb{E}_2$ are $2^{127}-1$ and $2^{126}-1$ respectively, hence

$$Period\,(\alpha_t) = 2^{127}-1, \quad Period\,(\beta_t) = 2^{126}-1, \tag{3.3}$$

and equation (3.1) is followed for $(2^{126}-1, 2^{127}-1)=1$.

Write $\omega_i = \omega_{i-1}\cdot x^{k_i}$, and let $n=(2^{127}-1)(2^{126}-1)$, $m=\sum_{0<i\le n} k_i$, it is easy to calculate that for each integer $c,\ 1\le c<16$, the occurrence times of c in the sum above is

$$2^{123}\cdot 2^{122}\cdot 14 + (2^{123}-1)\cdot 2^{122} + (2^{122}-1)\cdot 2^{123} = 2^{249}-2^{123}-2^{122}.$$

Thus,

$$m=(2^{249}-2^{123}-2^{122})\cdot 120 \tag{3.4}$$

and

$$\omega_n = \omega_0\cdot x^m = \omega_0\cdot x^{-2^{124}\cdot 5^2\cdot 3} \tag{3.5}$$

In the field $\mathbb{E}_3$ or $\mathbb{E}_4$, the order of x is equal to $2^{128}-1$, and $(2^{128}-1, 2^{124}\cdot 5^2\cdot 3)=15$, the formula (3.2) is followed. □

Note that $u_t \oplus u_{t-1} = \omega_t,\ v_t\oplus v_{t-1}=\tau_t,$ so we have

Corollary 1

$$Period\,(u_t), Period\,(v_t) \ge (2^{126}-1)(2^{127}-1)(2^{128}-1)/15. \tag{3.6}$$

In order to have some knowledge about the distributions of the sequences $\omega_t, \tau_t, u_t,$ and v_t, we show the following results.

Proposition 2 Suppose that $\mathbb{E}=\mathbb{F}[x]/q(x)$ is a finite field $GF(2^m)$, $q(x)$ is a primitive polynomial of degree m, and s is a positive integer, let $g(x)$ be the generating function $g(x)=\sum_{0\le k<s} x^k$, $g^n(x)=\sum_{0\le i<2^m-1} c_i(n)x^i$, then for each integer i, $0\le i<2^m-1$.

$$\lim_{n\to\infty}(c_i(n)/s^n) = 1/(2^m-1) \tag{3.7}$$

Proof. Let $c_i(n)/s^n = p_i(n)$, then $\sum_i p_i(n) = 1$. Denoted by $p'(n) = \min_i \{p_i(n)\}$, $p''(n) = \max_i \{p_i(n)\}$. It is easy to know that the sequence $\{p'(n)\}$ is non-decreasing and the sequence $\{p''(n)\}$ is non-increasing. Suppose that $\lim_n p'(n) = \mu$, j and k are two integers such that $p_j(n) = p'(n)$, $p_k(n) = p''(n)$, without loss generality, we can assume that $\lim_n p'(n) = \lim_n p_j(n) = \mu$. Let d be the least non-negative integer such that $k - j \equiv d \mod (2^m - 1)$, and $\lceil d/s \rceil = d_0$, then it has

$$p_j(n+d_0) - p_j(n) \geq \frac{p_k(n) - p_j(n)}{d+s} \geq \frac{1/(2^m-1) - \mu}{d+s}.$$

Let $n \to \propto$, it follows that

$$\mu = \frac{1}{2^m - 1}. \qquad \square$$

We know that *m*-sequences have good statistic properties in randomness [11,19], so we introduce the following conception, for an event $once$, denoted by $\hat{P}(once)$ the probability of the event $once$ occurs under the assumption the dices D'_t and D''_t behave randomly. Then the Proposition 2 can be rewritten as following, for any integer $i, 0 \leq i < 2^{128} - 1$,

$$\lim_{n\to\propto} \hat{P}(\omega_t = x^i) = \frac{1}{2^{128}-1}, \ \lim_{n\to\propto} \hat{P}(\tau_t = x^i) = \frac{1}{2^{128}-1}. \qquad (3.8)$$

Similarly, we have

Proposition 3 For $\forall \mu \in \mathbb{E}_3, or\, \mathbb{E}_4$, it has

$$\lim_{t\to\propto} \hat{P}(u_t = \mu) = \frac{1}{2^{128}}, \quad \lim_{t\to\propto} \hat{P}(v_t = \mu) = \frac{1}{2^{128}}, \qquad (3.9)$$

moreover, for any finite subset $J \subset \mathbb{Z}$, denoted by $U_t^J = \bigoplus_{i\in J} u_{i+t}$, $V_t^J = \bigoplus_{i\in J} v_{i+t}$, if $U_0^J \neq 0, V_0^J \neq 0$, then

$$\lim_{t\to\propto} \hat{P}(U_t^J = \mu) = \frac{1}{2^{128}}, \quad \lim_{t\to\propto} \hat{P}(V_t^J = \mu) = \frac{1}{2^{128}}. \qquad (3.10)$$

Proof. The proof is similar to the one of Proposition 2, but note that the any element in

$\mathbb{E}_3, or\, \mathbb{E}_4$ is a polynomial with degree less than 128, and so can be represented as a combination of $\{u_{t+i}\}_{i=0}^{127}$, the detail proof is omitted. □

With larger periods and good statistic properties, LFSRs are usually applied in the stream ciphers as main components. However, on the other hand, the linear relations between the bits of a LFSR's sequence is the fatal weakness which incur the various attacks, such as correlation attacks, algebraic attacks, distinguishing attacks, etc. In the cipher DICING, the relations between the states are dynamic, so that the attacks that draw support from the correlations are foiled in some degree. In the next, we give a discussion in respect to the main ones of the known attacks.

Correlation Attack

Correlation attack is one of main analyses for the stream ciphers, and there have been a lot of researches on this topic since the earlier work T. Siegenthaler[25] and W. Meier, O. Staffelbach [18]. See [3,10,21,23,24]. The cipher *Deffe* is a famous example in the stream ciphers that are encountered this kind of attacks [22,25].

Suppose that x is a binary segment of length l, let

$$\delta(x) = \bigoplus_{i=0}^{l-1} x[i] \tag{3.11}$$

It is obvious that $\delta(x) = 0, or 1$, depending on that the number of $1's$ bits in the segment x is even or odd. Furthermore, suppose that $f(x)$ is a function from $\mathbb{F}^n$ to $\mathbb{F}^m$, for $a \in \mathbb{F}^n$ and $b \in \mathbb{F}^m$, $a \neq 0$. We write

$$L(a,b,f(x)) = \delta(a \,\&\, f(x)) \oplus \delta(b \,\&\, x). \tag{3.12}$$

Clearly, $L(a,b,f(x)) = 0, or 1$ as the variable x varies, namely, equal to 0 for some $x's$ and equal to 1 for the other $x's$. We call the equations

$$L(a,b,f(x)) = 0, \text{ or } L(a,b,f(x)) = 1 \tag{3.13}$$

the linear approximations of function $f(x)$ with coefficients a and b, which are probabilistic equations. Clearly, if one of the equations in (3.13) is certainly known true or wrong by some way, then it will reveal a relation between the input x and the values of function $f(x)$ in some bits, and so a relation among the contents of the input x will be found, for the values of function $f(x)$ are known in a plaintext-known attack, thus this case should be avoided in a stream cipher. The case that the equations in (3.13) have greater probabilities is this kind of case, since it is easier to find true value by the samples test in statistics. This means that for a good cipher the chance of

$L(a,b,f(x)) = 0, or 1$ should be balanced for all possible a and b. We define

$$d(f,a,b) = \sum_x L(a,b,f(x)), \tag{3.14}$$

$$\Lambda(f,a,b) = \left| (2d(f,a,b) - 2^m)/2^m \right|, \tag{3.15}$$

$$\Delta(f) = \max_{a,b}\{\Lambda(f,a,b)\}, \tag{3.16}$$

$$\Delta_0(f) = \max_{a\neq 0}\{\Lambda(f,a,0)\}, \tag{3.17}$$

in (3.14) the summarize is the integer addition, which is just the times of $L(a,b,f(x)) = 1$, and $\Lambda(f,a,b)$ is the bias between the times of equations $L(a,b,f(x)) = 1$ and $L(a,b,f(x)) = 0$ as x run over the domain., $\Delta(f)$ is the maximum bias as a and b run over $\mathbb{F}^n$ and $\mathbb{F}^m$ respectively, and $\Delta_0(f)$ is a special case of $\Delta(f)$ in $b = 0$.

We know that the keystream can be written as $f(x_t)$, where $f(x_t)$ is a nonlinear function and x_t usually come from some LFSR's. As each bit of LFSR is linearly related to the initial state, that is, a linear combination of the initial bits $\{x_0[i]\}_0^{l-1}$ of sequence x_t, this means that every linear approximation as (3.13) will return to an equation about the initial bits $\{x_0[i]\}_0^{l-1}$ by, for example, Berlekamp-Massey algorithm. So if there are l or more linear approximations as (3.13) which are known absolutely true or with a rather higher probability, then the initial state will be deduced by solving the resulting linear system about $\{x_0[i]\}_0^{l-1}$. The main idea of correlation attack is to find the l linear approximations with higher probabilities. The first step in the attack is to look for sufficient correlations satisfied by x_t. Secondly, from these correlations, set up the corresponding parity checks, in which some are equal to 0, the others are equal to 1.Then select the l linear approximations with highest probabilities by maximum likelihood rule.

It is known that there are two ways to find these correlations, one is by squaring and shifting a correlation polynomial iteratively, and another is by selecting the polynomial multiples of the connection polynomial [10].

By squaring and shifting, it is easy to know that the number m of the correlations that one x_t satisfy is about

$$m \approx t \cdot \log_2(N/2k), \tag{3.18}$$

where the parameters N, k and t are the length of the known keystream, the length of a

correlation and the number of nonzero terms in a correlation polynomial [20].
From formula (3.18), we can know the number of correlations that one x satisfying will be limited, for example, in the case of the key space is 256 bits, and $t \le 64,$ then

$$m \le 2^{14}. \qquad (3.19)$$

This implies that the attack will be incapable when the bias of the parity check sequence $\le 1/2^7.$

The output size of the proposal cipher is 128 bits, so it is difficult to find $d(C,a,b)$ and $\Delta(C)$ for the computer capability limited. Nevertheless, we can provide an estimate for the upper bound about the bias. For the S-box $S(x)$ used in DICING, it has

$$\Delta(S) = 1/2^3 \qquad (3.20)$$

Moreover, in the combining function $C(\zeta, \vartheta)$, there are at least five S-boxes are activated, so we obtained the following estimation

$$\Delta(C) \le 1/2^{15}. \qquad (3.21)$$

So, the correlation attack in this case is not feasible for DICING.
As to the second way, selecting polynomial multiples, it is known that to find N multiples with the weight (number of nonzero monomials) no higher than w, the required computations is

$$Cost \ge (N \cdot 2^r)^{w/(w-1)}, \qquad (3.22)$$

where the parameter r is the degree of the connection polynomial, cf. [9].
On the other hand, we know that the bias of the parity check formed by w linear approximations with bias Δ is equal to Δ^w, so, by the theory of hypothesis testing, about $O(\Delta^{-2w})$ tests of parity checks are needed. From (3.21), we have known that $\Delta \le 1/2^{15}$, and in the formula (3.22) let $N = \Delta^{-2w} = 2^{30w},\ r = 128,$ then the required computations will be greater than

$$2^{(30w+128)w/(w-1)} > 2^{327},$$

thereby, this kind of correlation attack is also impossible for the cipher DICING.

Algebraic Attack.
In the above discussion for the correlation attack, we have seen that the stream ciphers should avoid the linear relations between the output and the inside components. Thus, the functions are generally used in the filters or the combiners are nonlinear, and the Boolean functions in each bit are not linear functions. Algebraic attack is such an attack for these ciphers that the orders of Boolean functions are not enough high, cf. [5]. In this analysis, the main idea is that taking every possible monomial in the Boolean function as a new variable, so the original algebraic equations become the linear equations but the number of the variables increases. If the size of the input is

m bits, then the number of the monomials of order no greater than k is $\sum_{0\le i\le k} C_m^i$. Thus, if the order of the Boolean functions is large, then the number of the variables will increase very fast in the linearization. Hence, the Boolean functions used in the stream ciphers should have higher order. However, on the other hand, it is known that by multiplying some multiplies can reduce the order of Boolean functions such that the order of the result function is no more than $\lceil (m+1)/2 \rceil$, see [5].

In DICING, the S-box $S_0(x) = 5\cdot(x\oplus 3)^{127}$, it can be written as

$$S_0(x) = (f_0, f_1, \ldots, f_7)\,, \tag{3.23}$$

where $f_i(x)$, $0 \le i \le 7$, are the Boolean functions of order seven. As we have used the S-boxes two times in the combining function $C(u_t, v_t)$, so, the number of new variables after the linearization will be about $2\sum_{0\le i\le 49} C_{128}^i \approx 2^{116}$, In order to set up 2^{116} linear equations over the field $\mathbb{F}$, 2^{109} output blocks are needed. We have seen that in DICING the relations between state ω_t (or τ_t) and state ω_{t+1} (or τ_{t+1}) are not known, the successful probability of a guess the relation from the time t to time $t+1$ is no more than $1/2^4$, and so the successful probability of a guess the relation between u_t and u_{t+k} is no more than $1/2^{4\times k}$. Therefore, algebraic attack for DICING is impossible.

It maybe should mention that here we have not counted the efficiency of the key-defining method used in S-box $S(x)$ and the transformation L_1 .

Distinguishing Attack

To guarantee the good randomness of the keystream, it is required that the keystream should not be distinguished from a truly random sequence with computations less than the exhaustive search. This implies that the keystream should be very balanced for statistics. It is shown by the practice that this requirement is somehow severe for the design of stream ciphers, a number of the known stream ciphers such as SNOW, SOBER and SCREAM etc. are encountered this kind of attack, see [4,8,17]. The main tool used in this attack is the theory of hypothesis testing in statistics. There are two ways to take this attack, one is directly to the distribution of output values of keystream, which is usually for the stream ciphers with smaller output sizes, the another is through linear approximations and parity checks. The following is a simple description for this kind of attack. Assume that some linear approximations $L(a,b,f(x))$ with bias $1/2^s$, and the states x_i satisfy the correlation

$$\bigoplus_{i\in J} x_i = 0, or\ 1 \tag{3.24}$$

where $J \subset \mathbb{Z}$ is a finite subset of the non-negative integers. Then we have

$$\bigoplus_{i\in J} L(a,b, f(x_i) = \bigoplus_{i\in J} \delta(a \,\&\, f(x_i)) = 0, or\ 1. \tag{3.25}$$

Denoted by $y(t) = \bigoplus_{i\in J} f(x_{t+i})$, the cardinality $|J| = w$, then it is easy to know that the sequence $y(t)$ is of the distribution with the bias about $1/2^{ws}$, that is,

$$\Delta_0(y(t)) \approx 1/2^{ws}. \tag{3.26}$$

By the theory of hypothesis testing, through about $O(2^{2ws})$ sample tests will distinguish this distribution from a random one with a significant level.

In order to analysis the cipher DICING, let a, b_1 and b_2 be the arrays of length 16 bytes, write $b = (b_1, b_2)$. From Proposition 2 and 3, we know that the distribution of the sequences u_t and v_t are nearly uniform, and it is clear the values of the function $C_0(u)$ is uniformly distributed, so if $b_1 = 0$, or $b_2 = 0$, then it is easy to know that

$$d(C(u,v),a,b) \le \frac{1}{2^{128}}. \tag{3.27}$$

This means that if the linear approximation $L(C(u,v),a,b)$ is applied to a distinguishing attack, it should be $b_1 \ne 0$, $b_2 \ne 0$. Consequently, if a subset $J \subset \mathbb{Z}$ is applied to form a parity check, then it should be that

$$\bigoplus_{i\in J} u_i = 0 \quad \text{and} \quad \bigoplus_{i\in J} v_i = 0. \tag{3.28}$$

This case seems very scarce for that the sequences u_t and v_t are nearly independent of each other. We call a subset $J \subset \mathbb{Z}$ as a correlation set if J satisfies the equations (3.30). Define $\| J \| = \max\{|b-a| \mid a,b \in J\}$, we conjecture that there is no identical correlation set J with $\| J \| < Period(u_t)$, where term identical means independent of the key K. On the other hand, suppose $J \subset \mathbb{Z}$ is a correlation set, denoted by $y(t) = \bigoplus_{i\in J} z_{t+i}$, then

$$\Delta_0(y(t)) \le 1/2^{15\cdot|J|},$$

and so

$$|J| \le 8. \tag{3.29}$$

Time-memory trade-off attacks

This attack has the aid of a pre-computed table that list the correspondences between the states and keystream. Clearly, this attack only success in the cases that the size of state is small or there are the relations between a small part of states and the keystream, and these relations are independent of the other part of the states. In DICING, the size of states is near to 2^{509} bits, and the states u_t and v_t have similar status in the process of generating keystream, moreover, with the transformations L_1, rotation R and S-box $S(x)$ such that the content of each bit will effect all the other bits in at most two cycles, so there are no the correspondences between some of small isolated parts of the states and the keystream.

Guess-and-Determine attacks

In this attack, at first guess the contents of some variables, which usually are the bits of the key or the states or the intermediary variables, and then deduce the contents of the rest variables by some conventional ways. If total computations in this procedure are less than the exhaustive search, then the attack is successful. The ciphers with bias construction or small states size are easy baffled by this attack. For the same reason as above, it seems no flaws in the proposal cipher for this attack.

Inversion Attacks

With the mask γ_0, and the S-box $S(x)$ and the transformation L_1 are key-defined, so, which are not easy taken off unless take 2^{64} tries for each, thus it will have few results in this way.

Some tests in statistics

We have made some tests about the statistic property of DICING. One test is in respect to the bias of linear approximations, for $a = b_1 = b_2 = 2^d, d = 0,1,\ldots,127,$ with 2^{30} outputs, the maximum bias $\leq 1/2^{13.6}$. Another test is for the distributions of the bit segments of the keystream, we have calculated the frequencies of segments of length 10 bits with 2^{30} outputs, the standard deviation of the frequencies $\leq 1/2^{18.5}$, indicates that it is near uniformly distributed

4. Implementation

The substitution $S(x)$ and the transformation L_1 can be combined into four 8×32 table look-ups $S_i(x)$, $1 \leq i \leq 4$, and $S_i(x)$ acts on the *i*-th byte of each row respectively, $1 \leq i \leq 4$. The method once was applied in AES [6].

In the platform of AMD$1.8G$, 32-bit processor, Borland C++ 5.0, the performance of DICING is presented in the following List 2

Report of Performance

Sub-processes	Time or Rate
Initialization	918 *cycles*
Self-cycling	24880 *cycles*
Encrypt/Decrypt Rate	28.2 *cycles/byte* or 452 *cycles/block*

List 2

From the list above we have seen that the DICING is faster than AES about two times. It is likely that there will be an improvement for the rate with an optimal code, and anticipatable that the cipher DICING will run much better in the cases of 64-processor or hardware.

5. Some Considerations in the Design

In the proposal cipher we introduced the called projectors replacing of LFSRs for that it seems more flexible for applications, which can be regarded as a combination of a LFSR and a FSM, finite states machine. For example, in this paper, the projectors Γ_1 and Γ_2 play the role of LFSR, and the projectors Γ_3 and Γ_4 resemble FSM.

The mechanism of applying two dices to control the process like the clock-controlled one is to interfere the linear relations in regular LFSRs or the Projectors.

The purpose of applying a self-cycling sub-process is that first to have the data in the initial states mixed enough, second to have the states start to output far from the initial states. This treatment had once occurred in the existed ciphers before, e.g. SNOW [7].

As ordinarily, S-boxes are used to get higher order of non-linearity, and linear transformations are employed for the diffusion of the data.

The key-defining procedure employed in (2.11) ~ (2.14) is to interfere the linear approximating, which was once applied in our block cipher designs. Besides, it has that advantage that it is not easy to be taken off. The affine transformation B in fact may be even arbitrary, not need restricted in non-singular.

6. Some possible variances in DICING

In the following are some possible variances in the design.

1) The constant mask γ_0 can be changed as a variable mask γ_t, e.g. $\gamma_t = \omega_t \oplus \tau_t$.

2) The Pr. Γ_2 may be reduced, and correspondingly in the updating of the Pr. Γ_1 moves two

bytes in a cycle instead of one byte, that only affect the period of the sequences, but it will have enough security at least for the key size 128 bits.

3) The key-defining procedure may be leaved out, and simply let $A = 1, B = 2, or\ 4, etc.$

Though we think that this will deduct some strength from the original one, but it is noted that in the security analysis above, we almost have not relied on the advantage of this procedure, therefore, the simplified will still satisfy the security requirement. But this only decreases the startup time.

References

[1] Bluetooth SIG, Bluetooth specification, version 1.0.A

[2] T. Beth, F.C. Piper, The stop-and-go generator, *Advances in Cryptology– Proceedings of EUROCRYPT 84* (LNCS 209), 88–92, 1985.

[3] V. Chepyzhov, T. Johansson and B. Smeets, A simple algorithm for fast correlation attacks on stream ciphers, *Fast Software Encryptionm FSE'2000*, LNCS Springer-Verlag,2000

[4] D. Coppersmith, S. Halevi, and C. Jutla. Cryptanalysis of stream ciphers with linear masking. In *CRYPTO'02*, *Lecture Notes in Computer Science*. Springer-Verlag, 2002.

[5] N. Courtois, W. Meier, Algebraic attack on stream ciphers with linear feedback, In *Advances in Cryptology—EUROCRYPT '2003*, LNCS 2656, 346-359, Springer-Verlag, 2003.

[6] J. Daemen, V. Rijmen, AES Proposal: Rijndael, 1998.

[7] P. Ekdahl and T. Johansson, SNOW— a new stream cipher. Submitted to NESSIE. Available on-line from http://www.it.lth.se/cryptology/snow/

[8] ------, Distinguishing attacks on SOBER-t16 and t32. In *Fast Software Encryption FSE' 2002* (LNCS 2365), 210-224, 2002.

[9] G. Dj Golic, Computation of low-weight parity-check polynomials, *Electronic Letters*, vol 32(21), Oct(1996), 1981-1982.

[10] G. Dj Golic, M. Salmasizadeh, Ed Dawson, Fast Correlation Attack on the Summation Generator, *J. Cryptology* (2000)13: 245-262.

[11] S.W. Golomb, Shift Register Sequences, Holden-Day, San Francisco, 1967. Reprinted by Aegean Park Press, 1982.

[12] C.G. Gunther, Alternating step generators controlled by de Bruijn sequences, *Advances in Cryptology–EUROCRYPT '87* (LNCS 304), 5–14, 1988.

[13] S. Halevi, D.Coppersmith, and C. Jutla, Scream: a software-efficient stream cipher, In *Fast Software Encryption FSE' 2002* (LNCS 2365), 195-209, 2002.

[14] P. Hawkes, G. Rose, Primitive specification and supporting documentation for SOBER-t16, submission to NESSIE. In *Proceeding of the First Open NESSIE Workshop*, 13-14 November 2000, Heverlee, Belgium.

[15] ------, Primitive specification and supporting documentation for SOBER-t32, submission to NESSIE. In *Proceeding of the First Open NESSIE Workshop*, 13-14 November 2000, Heverlee, Belgium. See:

[16] ------, Guess and Determine on SNOW (NESSIE), pdf

[17] T. Johansson, A. Maximov, A linear Distinguishing Attack on Scream, (ISIT 2003) pdf

[18] J.L. Massey, Shift-register synthesis and BCH decoding, *IEEE Transactions on Information Theory*, 15 (1969), 122–127.

[19] R.J. Mceliece, Finite Fields for Computer Scientists and Engineeers, Kluwer Academic Publishers, Boston, 1987.

[20] W. Meier and O. Staffelbach, Fast correlation attacks on certain stream ciphers, *Journal of Cryptology*,1(3) (1989), 159–176.

[21] ------, Correlation properties of combiners with memory in stream ciphers, *Journal of Cryptology*, 5 (1992), 67–86.

[22] A. Menezes, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptograpgy, CRC Press,1997.

[23] R.A. Rueppel, Analysis and Design of Stream Ciphers, Springer-Verlag, Berlin, 1986.

[24] T. Siegenthaler, Correlation-immunity of nonlinear combining functions for cryptographic applications, *IEEE Transactions on Information Theory*, 30 (1984), 776–780.

[25] ------, Decrypting a class of stream ciphers using ciphertext only, *IEEE Transactions on Computers*, 34 (1985), 81–85.