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 128bits and 256bits 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. In case that the cipher is abused or the plaintext of some ciphertext are known by some people, and so the keystream will become visible for them, the analysis for this case is called the plaintext-known analysis, 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 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, the clock-controlled methods comes from this consideration.

The proposal 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 with 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 may be taken as large as 256 bits, and the size of output of DICING 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 >>, <<, $|$ 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 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

As 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

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

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 $D_t = (D_t' \oplus D_t'')$, $a = 1 + (D_t \,\&15)$, $b = 1 + (D_t >> 4)$, they satisfy

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

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

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

The initial values ω_0, τ_0, u_0 and v_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

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

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 string ζ.

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 the S-box $S(x)$ from $S_0(x)$ and a diffusion transformation L. The process is as following.

For a string ρ of 8 bytes, we define an 8-bits vector V_ρ and a 8×8 matrix M_ρ:

$$V_\rho[i]=\rho[8i+i]_{bit}, 0\le i<8, \qquad M_\rho=T_u\cdot J\cdot T_l. \tag{2.6}$$

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,

$$a_{i,j}=\begin{cases}\rho[8i+j]_{bit} & if\ i<j,\\ 1 & if\ i=j,\\ 0 & if\ i>j,\end{cases} \qquad b_{i,j}=\begin{cases}\rho[8i+j]_{bit} & if\ i>j,\\ 1 & if\ i=j,\\ 0 & if\ i<j,\end{cases} \tag{2.7}$$

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

Suppose that K is the secret key, let $\lambda=K[0,15]_{byte}\oplus K[16,31]_{byte}$, if $|K|=256$, else $\lambda=K[0,15]_{byte}$, and $\lambda'=\lambda[0,7]_{byte}$, $\lambda''=\lambda[8,15]_{byte}$,define two affine transformations on $\mathbb{K}$

$$A(x)=M_{\lambda'}(x), \qquad B(x)=M_{\lambda''}(x), \qquad x\in\mathbb{K}. \tag{2.8}$$

Denoted by $V_1=V_{\lambda'}\oplus V_{\lambda''}$, and $V_2=V_{\lambda'}\oplus ROTL8(V_{\lambda''},1)$, and then define a new S-box $S(x)$ and a transformation L on $\mathbb{K}^4$,

$$S(x)=S_0(x\oplus V_2)\oplus V_1, \qquad L=\begin{pmatrix} A & B & A & A\oplus B\\ B & A & A\oplus B & A\\ A & A\oplus B & A & B\\ A\oplus B & A & B & A\end{pmatrix}. \tag{2.9}$$

Suppose that ζ 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 ζ . Simply, we denote

$$Q(\zeta) = L \cdot S(\zeta)\,. \tag{2.10}$$

In the $ivsetup$, the second step of the startup, the internal states will be initialized with the secret key and the initial value. In the generating keystream we will employ one mask of 16 bytes, which are denoted by η .

For a 32-bytes string ζ we define a bytes permutation ϕ: $\zeta^{\phi} = \phi(\zeta)$, $\zeta^{\phi}[i] = \zeta[4i \bmod 31]$, for $0 \le i < 31$, and $\zeta^{\phi}[31] = \zeta[31]$. Let $\hat{K} = K$ if $\left|K\right| = 256$ else $\hat{K} = K \,|\, (\sim K)$, denoted by $\breve{K} = (\sim \hat{K}[16,31]_{byte}) \,|\, (\sim \hat{K}[0,15]_{byte})$. We define the functions

$$F(\zeta) = Q(\phi(\zeta)), \quad G(\zeta) = F(F(F(\zeta) \oplus \hat{K}) \oplus \breve{K}). \tag{2.11}$$

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 ξ_i, $0 \le i \le 3$, are four 32-bytes strings defined as

$$\xi_0 = G(IV \oplus c), \quad \xi_i = G(\xi_{i-1} \oplus c), \quad i = 1, 2, 3. \tag{2.12}$$

The internal states are initialized respectively as following

$$\eta = \xi_0[0,15] \oplus \xi_0[16,31], \quad (u_0, v_0) = \xi_1, \quad (\alpha_0, \beta_0) = \xi_2', \quad (\omega_0, \tau_0) = \xi_3, \tag{2.13}$$

where $\xi_2' = \xi_2[0,126] \,|\, \xi_2[128,253]$, i.e. $\alpha_0 = \xi_2[0,126]$, $\beta_0 = \xi_2[128,253]$.

If $\xi_3 = 0$, the states ω_0 and τ_0 will be re-set as

$$(\omega_0, \tau_0) = \hat{K}\,. \tag{2.14}$$

Note. For a secret key, there is at most one IV such that $\xi_3 = 0$.

After initializing, 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 as stated in (2.2) ~ (2.4). Suppose that u and v are two 16-bytes strings, which are also viewed as 4×4 matrices of bytes in the ordinary way. Denoted by M^T the transposition of a matrix M , the combining function is defined as

$$C(u,v) = Q((Q(u) \oplus v)^T) \oplus \eta. \tag{2.15}$$

Denoted by z_t the keystream in the time t , $(t > 0)$, then

$$z_t = C(u_t, v_t)\,. \tag{2.16}$$

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

List of the Primitive Polynomials used

Polynomials	Expression
$p(x)$	$x^8+x^6+x^5+x+1$
$p_1(x)$	$x^{127}+(x^{89}+x^{41}+1)(x^3+1)$
$p_2(x)$	$x^{126}+(x^{83}+x^{35}+1)(x^7+1)$
$p_3(x)$	$x^{128}+(x^{96}+x^{67}+x^{32}+1)(x^3+1)$
$p_4(x)$	$x^{128}+(x^{96}+x^{64}+x^{37}+1)(x^7+x^5+1)$

List 1

The Sketch of Encryption Process

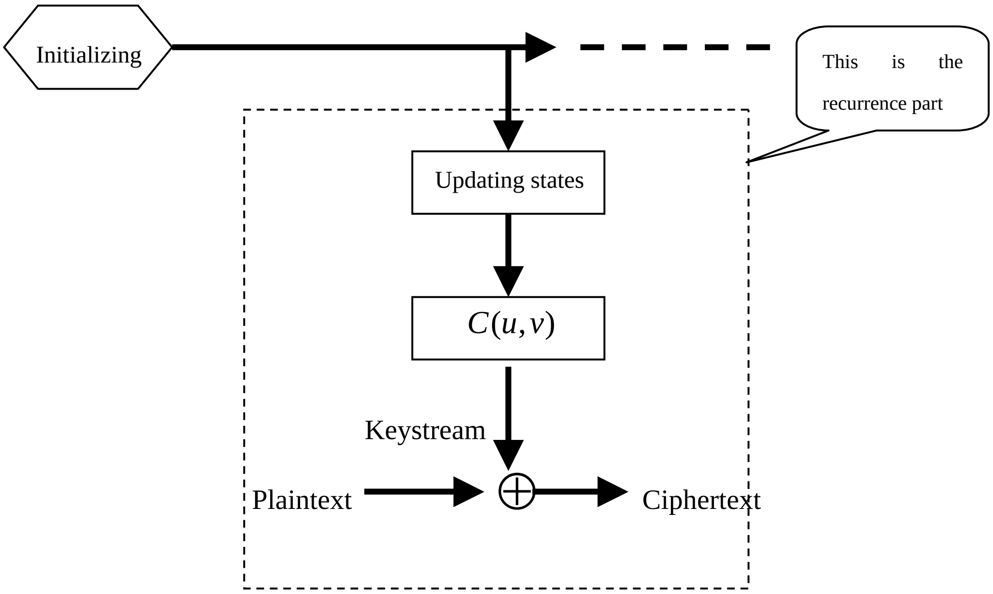

Fig.1

3. Security Analysis

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

Period and Distribution

Denote $\pi(z_t)$ as the period of a sequence z_t .

Proposition 1

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

$$\pi(\omega_t) = \pi(\tau_t) = (2^{126}-1)(2^{127}-1)(2^{128}-1)/3 \qquad (3.2)$$

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

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

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 $c,\ 1 < c \le 16$, the occurrence times of c in the sum above is $2^{249}-2^{123}-2^{122}$ and integer 1 occurs one more times than the number. Thus,

$$m = (2^{249}-2^{123}-2^{122}) \cdot 136 + 1, \qquad (3.4)$$

and

$$\omega_n = \omega_0 \cdot x^m = \omega_0 \cdot x^{1-2^{124}\cdot 5\cdot 17} \qquad (3.5)$$

In the field $\mathbb{E}_3$ or $\mathbb{E}_4$, the order of x is equal to $2^{128}-1$, and $(2^{128}-1, 2^{124}\cdot 5\cdot 17-1) = 3$, 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

$$\pi(u_t), \pi(v_t) \ge (2^{126}-1)(2^{127}-1)(2^{128}-1)/3. \qquad (3.6)$$

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, $0 \le i < 2^m - 1$.

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

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

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

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

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

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 $i, 0 \le i < 2^{128} - 1$,

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

Similarly, we have

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

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

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

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

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. □

In the next, we give a discussion in respect to the main ones of the known attacks.

Correlation Attack

For a binary segment x of length l, denote by

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

Suppose that $f(x)$ is a function from $\mathbb{F}^n$ to $\mathbb{F}^m$, for $a\in\mathbb{F}^n$ and $b\in\mathbb{F}^m$, $a\neq 0$, let

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

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, and define

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

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

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

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

If the variable x is from some LFSR, we know that a linear approximation will return to an equation about the initial bits $\{x_0[i]\}_0^{l-1}$. The main idea of correlation attack is to find the l linear approximations with higher probabilities by statistic means called parity checks.

Clearly, to form such an attack should have enough many correlations. There are two ways known to find these correlations. One way is by squaring and shifting a correlation polynomial iteratively. It is not difficult to know that the number 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 [5].

In the case of the key space is 256 bits, and $t \le 64,$ from formula (3.18) it has

$$m \le 2^{14}. \tag{3.19}$$

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

The output size of DICING 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

$$\Delta(S) = 1/2^3 \tag{3.20}$$

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

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

So, the correlation attack in this case is not feasible for DICING.

The second way is to select 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

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

where the parameter r is the degree of the connection polynomial, cf. [2].

On the other hand, we know that the bias of the parity check formed by w linear approximations with bias Δ is about 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.

The main idea of algebraic attack 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 $\sum_{0\le i\le k} C_m^i$, to see [1].

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

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

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.

Distinguishing Attack

To guarantee a good randomness, it is required that the keystream should not be distinguished from a truly random sequence with computations less than the exhaustive search. A usual one of this kind of attacks is as the following. 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 $T(\zeta)$ 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 $b_1\ne0$, $b_2\ne0$. Consequently, if a subset $J\subset\mathbb{Z}$ is applied to form a parity check, then it should be that

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

We call a subset $J\subset\mathbb{Z}$ as a correlation set if J satisfies the equations (3.28). Define $\|J\|=\max\{|b-a|\mid a,b\in J\}$, we conjecture that there is no identical correlation set J with $\|J\|<\pi(u_t)$, 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

With the transformations Q, transposition T 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

For the same reason as above, it seems no flaws in the proposal cipher for this attack.

Inversion Attacks

With the mask η and the S-box $S(x)$ and the transformation L are key-defined, which are not easy taken off unless take 2^{64} tries for each, thus we think that the inversion attack will be difficult to feasible.

Chosen-IV Attacks

The initialization and combining functions also have protected DICING against the chosen-IV attacks, that is, the attacks in the transverse direction.

Collision Attacks

The places should be paid attension to collision attacks are the initialization and the updating of internal states. In the initialization of DICING, the function $G(\zeta)$ is injective if the secret key K is fixed, so ξ_0 's will be different for the different IV and so ξ_i 's will be different, $0 \le i \le 3$. Moreover, in the beginning of theis section we have shown a explicit bounds of the periods of the internal states ω_t, τ_t, u_t and v_t, thus there will no be chance of collision attacks to DICING.

Timing Attacks

In DICING, the process of updating states is dependent on the values of dice D, which is variable, so maybe there will be some difference in implementation time for some different values of dice D. But we know that the distribution of these values of dice D are very balanced, hence the difference will be made up in a series of updating processes. Besides, with our reference code, we also have not found a remarkable timing gap in the initialization process .

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} blocks of outputs, the maximum bias $\le 1/2^{13.5}$. 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} blocks of outputs, the standard deviation of the frequencies $\le 1/2^{23.5}$, indicates that it is very balanced in the distribution.

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 is as following

Report of Performance

Sub-processes	Time or Rate
Keysetup	6850 *cycles*
IVsetup	1800 *cycles*
Keystream Rate	13.5 *cycles/byte* or 216 *cycles/block*

List 2

The presented algorithm DICING now is one of candidates for eSTREAM [4], but here with a little difference from [4] in (2.8) and (2.9), the vector V_1 has been moved into S-box $S(x)$ from the transformation $A(x)$, for the detail please refer to the papers[3], [4] and [6] .

5. The reduced versions for DICING

The algorithm DICING presented above is a very conservative one, the users who wish to strive for a faster rate may adopt the reduced version as following, simply take the combining function

$$C(u,v) = Q(u) \oplus v, \tag{5.1}$$

and omit the Pr. β_t. In this way, the keystream rate may be decreased about $3\, cycles / byte$. We think the reduced one will also have sufficient security.

In the case there are no attacks for conditional branching, the combining function may be taken as

$$C(u,v) = \begin{cases} Q(u) \oplus v & if \ \alpha[0] = 0, \\ Q(v) \oplus u & if \ \alpha[0] = 1. \end{cases} \tag{5.2}$$

The later keystream will have larger period and will be more difficult for mathematical cryptanalysis.

References

[1] 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.

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

[3] A.P. Li, A New Stream Cipher: DICING, now available at http://www.ecrypt.eu.org/stream/dicing.html

[4] A.P. Li, ------An update for phase2 of eSTREAM, now available at eSTREAM - The ECRYPT Stream Cipher Project - Phase 2

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

[6] Gilles Piret, Practical Attacks on one Version of DICING. Available at http://www.ecrypt.eu.org/stream/papersdir/051.pdf