Loiss: A Byte-Oriented Stream Cipher *

Dengguo Feng, Xiutao Feng, Wentao Zhang, Xiubin Fan and Chuankun Wu

State Key Laboratory of Information Security, Institute of Software,
Chinese Academy of Sciences, Beijing, 100190, China.
{feng, fengxt, zhangwt, fxb, ckwu}@is.iscas.ac.cn

Abstract. This paper presents a byte-oriented stream cipher – Loiss, which takes a 128-bit initial key and a 128-bit initial vector as inputs, and outputs a key stream of bytes. The algorithm is based on a linear feedback shift register, and uses a structure called BOMM in the filter generator, which has good property on resisting against algebraic attacks, linear distinguishing attacks and fast correlation attacks. In order for BOMM to be balanced, the S-boxes in BOMM must be orthomorphic permutations. To further improve the capability in resisting against those attacks, the S-boxes in BOMM must also possess some good cryptographic properties, for example, high algebraic immunity, high nonlinearity, and so on. However current researches on orthomorphic permutations pay little attention on their cryptographic properties, and we believe that Loiss not only enriches applications of orthomorphic permutations in cryptography, but also motivates the research on a variety of cryptographic properties of orthomorphic permutations.

Keywords: stream ciphers, Loiss, BOMM, orthomorphic permutations

1 Introduction

Stream ciphers are widely used in secure network communications to protect communication data. A stream cipher algorithm is usually composed of a pseudorandom sequence generator and a plaintext mask function. The pseudorandom sequence generator first generates key streams under the control of an initial seed key, and then the plaintext mask function masks plaintexts with the above generated key streams and obtains the corresponding ciphertexts. Usually the mask function is the exclusive-OR operation.

Traditional stream cipher algorithms are mostly bit-oriented. With the rapid development of communication techniques, communication bands become wider, and requirements on data throughput become higher. The traditional bit-oriented stream ciphers can hardly be designed to meet the requirements of communication applications in nowadays, specially in software implementations. For a more efficient use of modern computer hardware, some word-oriented (8/32-bit

* This work was supported by the Natural Science Foundation of China (Grant No. 60833008 and 60902024)

word) stream ciphers have been proposed, for example, 3GPP the third generation communication encryption standard SNOW 3G [1], and many algorithms recommended by Europe eSTREAM project [2].

A common design model of stream ciphers is to use a linear feedback shift register (LFSR) together with a nonlinear filter generator. The outputs of the LFSR go to the nonlinear filter generator for further process before the final key stream is produced. In this work we present a novel byte-oriented stream cipher – Loiss, which uses the above described model, and takes a 128-bit initial key and a 128-bit initial vector as inputs, and outputs a key stream of bytes. The Loiss algorithm has an LFSR, and uses a structure called byte-oriented mixer with memories (BOMM) as a part of the nonlinear filter generator. BOMM itself contains some memory units and uses S-boxes as building blocks. The design of the BOMM component adopts the idea of the stream cipher RC4 [3], and adds the properties of confusion and accumulation. BOMM has good capability in resisting against a number of common attacks on stream ciphers, including algebraic attack, linear distinguishing attack, and fast correlation attack. In order for BOMM to be balanced in the statistical sense, the S-boxes in BOMM must be orthomorphic permutations. What's more, to further improve the capability in resisting against those attacks, the S-boxes in BOMM must also possess some good cryptographic properties, for example, high algebraic immunity, high nonlinearity, and so on. Unfortunately, little research results about these properties can be found from public literatures, and researches on orthomorphic permutations have been mainly about their construction and counting [5, 6]. In the design of Loiss, extensive computing assistance together with some fundamental theoretical analysis are used. We believe that the Loiss algorithm will not only enriches applications of orthomorphic permutations in cryptography [7, 8], but also motivate the research on cryptographic properties of orthomorphic permutations.

The rest of the paper is organized as follows: In section 2, the Loiss algorithm is described in detail, and in section 3, some basic properties of Loiss are introduced, and in-depth security analysis is given in section 4, and finally basic analysis on software and hardware implementation cost is given in section 5.

2 Description of Loiss

As stated above, the Loiss algorithm is a byte-oriented stream cipher, which generates a sequence of bytes (the key stream), one byte at a time. The execution of the Loiss is under the control of a 128-bit initial key and a 128-bit initial vector.

Loiss is logically composed of three parts: LFSR, the nonlinear function F and BOMM, see Figure 1.

2.1 The LFSR

Underlying finite field Let F_2 be the binary field with elements 0 and 1 and $F_2[x]$ be the polynomial ring over F_2. The field F_{2^8} with 2^8 elements is defined

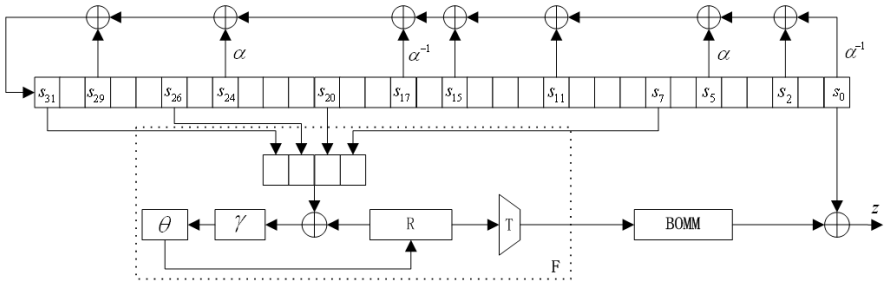

Fig. 1. The structure of Loiss

by a primitive polynomial $\pi(x) = x^8 + x^7 + x^5 + x^3 + 1$ over F_2 as the quotient $F_2[x]/(\pi(x))$. Let α be a root of the polynomial $\pi(x)$ in F_{2^8}, i.e., $\pi(\alpha) = 0$. Then $1, \alpha, \alpha^2, \cdots, \alpha^7$ form a basis of F_{2^8} and any element x in F_{2^8} can be uniquely written as

$$x = x_0 + x_1\alpha + \cdots + x_7\alpha^7,$$

where $x_i \in F_2$, $0 \le i \le 7$. Further the element x is represented by an 8-bit string or an 8-bit integer according to the following bijection mappings from F_{2^8} to the set $\{0,1\}^8$ or $\{0, 1, 2, \cdots, 2^8 - 1\}$:

$$x = \sum_{i=0}^{7} x_i\alpha^i \mapsto \quad x_7 \parallel x_6 \parallel \cdots \parallel x_0$$

or

$$x = \sum_{i=0}^{7} x_i\alpha^i \mapsto \quad \sum_{i=0}^{7} x_i 2^i,$$

where $\parallel$ denotes the concatenation of two bit strings. In this sense any element in F_{2^8} can be represented by an 8-bit string or an integer between 0 and 255.

Definition of the LFSR The LFSR in the Loiss algorithm is defined over the field F_{2^8} and and contains 32 byte registers, denote them as s_i, where $0 \le i \le 31$. The characteristic polynomial $f(x)$ of LFSR is defined as below:

$$f(x) = x^{32} + x^{29} + \alpha x^{24} + \alpha^{-1}x^{17} + x^{15} + x^{11} + \alpha x^5 + x^2 + \alpha^{-1}. \qquad (1)$$

Let $(s_0^{(t)}, s_1^{(t)}, s_2^{(t)}, \cdots, s_{31}^{(t)})$ be the state of LFSR at time t ($t \ge 0$). Then the state $(s_0^{(t+1)}, s_1^{(t+1)}, s_2^{(t+1)}, \cdots, s_{31}^{(t+1)})$ at time $t+1$ satisfies

$$\begin{aligned} s_{31}^{(t+1)} &= s_{29}^{(t)} \oplus \alpha s_{24}^{(t)} \oplus \alpha^{-1} s_{17}^{(t)} \oplus s_{15}^{(t)} \oplus s_{11}^{(t)} \oplus \alpha s_5^{(t)} \oplus s_2^{(t)} \oplus \alpha^{-1} s_0^{(t)}, \\ s_i^{(t+1)} &= s_{i+1}^{(t)}, \quad i = 0, 1, 2, \cdots, 30. \end{aligned}$$

2.2 The nonlinear function F

The nonlinear function F (the dotted rectangle in figure 1) is a compressing function from 32 bits to 8 bits, which contains a 32-bit memory unit R. The function F takes the values of the registers $s_{31}, s_{26}, s_{20}, s_7$ of LFSR as inputs, and outputs a byte w, see Figure 2.

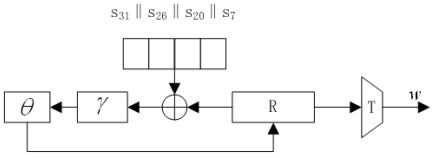

Fig. 2. The nonlinear function F

Let $s_{31}^{(t)}$, $s_{26}^{(t)}$, $s_{20}^{(t)}$ and $s_7^{(t)}$ be the values of the registers s_{31}, s_{26}, s_{20} and s_7 respectively at time t, and w be the output of F. Denote by $R^{(t)}$ and $R^{(t+1)}$ the values of the memory unit R at time t and $t+1$ respectively. Then we have

$$\begin{aligned} w &= T(R^{(t)}), \\ X &= s_{31}^{(t)} \parallel s_{26}^{(t)} \parallel s_{20}^{(t)} \parallel s_7^{(t)}, \\ R^{(t+1)} &= \theta(\gamma(X \oplus R^{(t)})), \end{aligned}$$

where $T(\cdot)$ is a truncation function which truncates the leftmost 8 bits from $R^{(t)}$ as output; γ is obtained by paralleling 4 S-boxes S_1 of size 8×8, that is,

$$\gamma(x_1 \parallel x_2 \parallel x_3 \parallel x_4) = S_1(x_1) \parallel S_1(x_2) \parallel S_1(x_3) \parallel S_1(x_4),$$

where x_i is a byte, $0 \le i \le 3$ and S_1 is defined in Table 4 (see appendix A); θ is a linear transformation on 32-bit strings, and is the same as the one used in the block cipher SMS4 [8], which is defined as

$$\theta(x) = x \oplus (x \lll 2) \oplus (x \lll 10) \oplus (x \lll 18) \oplus (x \lll 24), \tag{2}$$

where $\lll$ denotes the left cyclic shift on 32-bit strings.

2.3 The BOMM structure

BOMM is a transformation from 8 bits to 8 bits, and contains 16 byte memory units, denote them as y_i, $0 \le i \le 15$, see Figure 3.

Let w and v be the input and the output of BOMM respectively. Denote by $y_i^{(t)}$ and $y_i^{(t+1)}$ be the values of the memory units y_i at time t and $t+1$

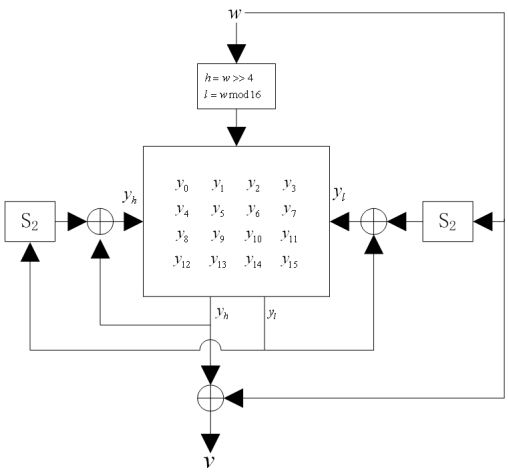

Fig. 3. The structure of BOMM

respectively, where $i = 0, 1, \cdots, 15$. Then BOMM works as follows:

$$\begin{aligned}
h &= w \gg 4, \\
l &= w \bmod 16, \\
v &= y_h^{(t)} \oplus w, \\
y_l^{(t+1)} &= y_l^{(t)} \oplus S_2(w), \\
y_h^{(t+1)} &= \begin{cases} y_h^{(t)} \oplus S_2(y_l^{(t+1)}), & \text{if } h \neq l, \\ y_l^{(t+1)} \oplus S_2(y_l^{(t+1)}), & \text{if } h = l, \end{cases} \\
y_i^{(t+1)} &= y_i^{(t)}, \quad \text{for } i = 0, 1, \cdots, 15 \text{ and } i \neq h, l,
\end{aligned}$$

where $\gg$ denotes the right shift operator, and S_2 is an S-box of size 8×8, see Table 5 in appendix A.

2.4 Initialization of Loiss

The initialization process of Loiss can be divided into two stages:

In the first stage, it loads a 128-bit initial key and a 128-bit initial vector into the memory units of LFSR and BOMM as well, and then set the initial value of the 32-bit memory unit R of F to be zero, i.e., $R^{(0)} = 0$.

Set

$$\begin{aligned}
\mathrm{IK} &= \mathrm{IK}_0 \parallel \mathrm{IK}_1 \parallel \cdots \parallel \mathrm{IK}_{15}, \\
\mathrm{IV} &= \mathrm{IV}_0 \parallel \mathrm{IV}_1 \parallel \cdots \parallel \mathrm{IV}_{15},
\end{aligned}$$

where both IK_i and IV_i are bytes, $0 \leq i \leq 15$.

Let the initial states of LFSR and BOMM be $(s_0^{(0)}, s_1^{(0)}, \cdots, s_{31}^{(0)})$ and $(y_0^{(0)}, y_1^{(0)}, \cdots, y_{15}^{(0)})$ respectively. Then for $0 \leq i \leq 15$, we have

$$\begin{aligned} s_i^{(0)} &= \mathrm{IK}_i, \\ s_{i+16}^{(0)} &= \mathrm{IK}_i \oplus \mathrm{IV}_i, \\ y_i^{(0)} &= \mathrm{IV}_i. \end{aligned}$$

In the second stage, Loiss runs 64 times and the output of BOMM takes part in the feedback calculation of LFSR, see Figure 4.

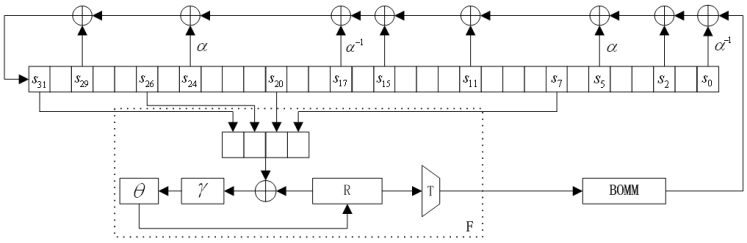

Fig. 4. The initialization of Loiss

2.5 Key stream generation

After the initialization, Loiss starts to produce key stream. Loiss produces one byte of key stream when it runs one time. Let z_t be the output of Loiss at time t. Then

$$z^{(t)} = s_0^{(t)} \oplus v^{(t)}, \tag{3}$$

where $s_0^{(t)}$ and $v^{(t)}$ are the value of the register s_0 of LFSR and the output of BOMM respectively at time t.

3 Some basic properties of the components in Loiss

3.1 Properties of LFSR

The LFSR of the Loiss algorithm is defined over the finite field F_{2^8}, and its characteristic polynomial $f(x)$ (see equation (1) for definition) is a primitive polynomial over F_{2^8} of degree 32. Thus non-zero sequences over F_{2^8} generated by $f(x)$ are m-sequences, and their periods are $2^{256} - 1$.

Let $\underline{a} = (a_0, a_1, \cdots, a_t, \cdots)$ be a non-zero sequence over F_{2^8} generated by $f(x)$. Note that $a_t \in F_{2^8}$ can be expressed as follows:

$$a_t = a_{t,7}\alpha^7 + a_{t,6}\alpha^6 + \cdots + s_{t,1}\alpha + a_{t,0}$$

for $t \geq 0$, where $1, \alpha, \cdots, \alpha^7$ is a basis of the finite field F_{2^8}. We call the sequence $\underline{a}_i = (a_{0,i}, a_{1,i}, \cdots, a_{t,i}, \cdots)$ $(0 \leq i \leq 7)$ derived from $\underline{a}$ as *the i-th coordinate sequence*. Then it is known [9] that each coordinate sequence of sequences generated by $f(x)$ is an m-sequence over the binary field F_2, and its characteristic polynomial is a primitive polynomial over F_2 of degree 256. And it is easy to verify that the characteristic polynomial of each coordinate sequence of non-zero sequences generated by $f(x)$ has a weight (the number of non-zero coefficients) of 131.

In addition, when choosing the polynomial $f(x)$, for a better resistance against linear distinguishing attack and fast correlation attack [10, 11], we avoid as much as possible that $f(x)$ has a multiple with low degree and low weight whose all non-zero coefficients are one.

3.2 Properties of F

The nonlinear function F contains a 32-bit memory unit R and uses S-boxes as building blocks. F itself has good resistance against linear distinguishing attack and fast correlation attack. Simple computation reveals that the following properties about F hold.

Property 1. *The algebraic degree, nonlinearity, differential uniformity and algebraic immunity of the S-box S_1 are 7, 112, 4 and 2 respectively.*

Property 2. *When we view θ as a transformation over the vector space $(F_{2^8})^4$, its linear branch number is equal to 5.*

The bias of a linear approximation true with probability p is defined as $\varepsilon = p - 1/2$. Then the following two properties are also easy to prove.

Property 3. *The bias of arbitrary linear approximations of 2-round F is zero.*

Property 4. *The number of active S-boxes of arbitrary linear approximations of 3-round F is at least 5.*

3.3 Properties of BOMM

The S-box S_2.

Property 5. *The algebraic degree, nonlinearity, differential uniformity and algebraic immunity of the S-box S_2 are 5, 112, 16 and 2 respectively.*

Definition 1. *Let $p(x)$ be a mapping over the finite field F_{2^n} with 2^n elements, where n is a positive integer. Then $p(x)$ is called an orthomorphic permutation if both $p(x)$ and $p(x) \oplus x$ are permutations over F_{2^n}.*

Property 6. *The S-box S_2 is an orthomorphic permutation over F_{2^8}.*

The balance of BOMM. Let two random variables X and Y be independent and uniformly distributed, then for simplicity of description, X and Y are called IUD random variables.

Definition 2. *Suppose that the variables $y_0^{(t)}$, $y_1^{(t)}$, $\cdots$, $y_{15}^{(t)}$ and the input w of BOMM are pairwise IUD random variables over F_{2^8} at time t. Denote by v the output of BOMM at time t. Then BOMM is called to be* balanced *if for arbitrary element $a \in F_{2^8}$ and $0 \le i \le 15$, we have*

$$\Pr(v = a) = \Pr\left(y_i^{(t+1)} = a\right) = \frac{1}{256}.$$

Property 7. *BOMM is balanced if and only if the S-box S_2 is an orthomorphic permutation over F_{2^8}.*

Proof. We first prove the sufficiency. Let $h = w \gg 4$ and $l = w \bmod 16$. Then both h and l are IUD variables over the set $Z_{2^4} = \{0, 1, 2, \cdots, 15\}$. Since $v = y_h^{(t)} \oplus w$, and $y_h^{(t)}$ and w are IUD variables, it is easy to see that $\Pr(v = a) = \frac{1}{256}$ for arbitrary given $a \in F_{2^8}$. Now we consider the updates of the memory units y_k. Note that only two units y_l and y_h are updated at time t, thus we only need to prove that

$$\Pr\left(y_l^{(t+1)} = a\right) = \Pr\left(y_h^{(t+1)} = a\right) = \frac{1}{256}$$

for arbitrary given $a \in F_{2^8}$. Below we consider two cases.

1. $h \neq l$.
   Since $y_l^{(t+1)} = y_l^{(t)} \oplus S_2(w)$, note that $y_l^{(t)}$ and w are independent and S_2 is a permutation, thus we have
   $$\begin{aligned}\Pr\left(y_l^{(t+1)} = a \mid h \neq l\right) &= \Pr\left(y_l^{(t)} \oplus S_2(w) = a\right) \\ &= \sum_{b \in F_{2^8}} \Pr\left(y_l^{(t)} = b\right) \Pr\left(S_2(w) = a \oplus b\right) \\ &= 256 \cdot \frac{1}{256} \cdot \frac{1}{256} = \frac{1}{256}.\end{aligned}$$
   Similarly, since $y_h^{(t+1)} = y_h^{(t)} \oplus S_2(y_l^{(t+1)})$, note that $y_h^{(t)}$ and $y_l^{(t+1)}$ are also independent, we can obtain $\Pr\left(y_h^{(t+1)} = a\right) = \frac{1}{256}$ in the same way.
2. $h = l$.
   Only one unit $y_h = y_l$ is updated in this case. Note that
   $$y_h^{(t+1)} = y_h^{(t)} \oplus S_2(w) \oplus S_2(y_h^{(t)} \oplus S_2(w))$$
   and S_2 is an orthomorphic permutation, thus we have
   $$\begin{aligned}\Pr\left(y_h^{(t+1)} = a \mid h = l\right) &= \Pr\left(y_h^{(t)} \oplus S_2(w) \oplus S_2(y_h^{(t)} \oplus S_2(w)) = a\right) \\ &= \sum_{b \in F_{2^8}} \Pr\left(y_h^{(t)} \oplus S_2(w) = b\right) \Pr\left(S_2(b) \oplus b = a\right) \\ &= 256 \cdot \frac{1}{256} \cdot \frac{1}{256} = \frac{1}{256}.\end{aligned}$$

Second we prove the necessity. At above we have proven that

$$\Pr\left(y_h^{(t+1)} = a \mid h \neq l\right) = \frac{1}{256}.$$

By the balance property of BOMM, we have $\Pr\left(y_h^{(t+1)} = a \mid h = l\right) = \frac{1}{256}$. Since

$$\begin{aligned}
&\Pr\left(y_h^{(t+1)} = a \mid h = l\right) = \Pr\left(y_h^{(t)} \oplus S_2(w) \oplus S_2(y_h^{(t)} \oplus S_2(w)) = a\right) \\
&= \sum_{b \in F_{2^8}} \Pr\left(y_h^{(t)} \oplus S_2(w) = b\right) \Pr\left(S_2(b) \oplus b = a\right) \\
&= \frac{1}{256} \sum_{b \in F_{2^8}} \Pr\left(S_2(b) \oplus b = a\right),
\end{aligned}$$

thus we have $\sum_{b \in F_{2^8}} \Pr\left(S_2(b) \oplus b = a\right) = 1$ for any $a \in F_{2^8}$. It follows that the equation $S_2(x) \oplus x = a$ has exactly one solution for any $a \in F_{2^8}$, that is, $S_2(x) \oplus x$ is a permutation. So S_2 is an orthomorphic permutation. ■

4 Security analysis

4.1 Guess and determine attack

The guess and determine attack is a known plaintext attack for state recovery [12, 13]. Its main idea is that: an attacker first guesses the value of a portion of inner states of the target algorithm, then it takes a little cost to deduce all the rest of the inner states making use of the guessed portion of the inner states and a few known key stream bits.

As for Loiss, below we construct a guess and determine attack.

Definition 3. *Let $w^{(t+i)}$ be the outputs of the nonlinear function F at the seven successive times starting from time t, $0 \leq i \leq 6$. Then when the following conditions are met, we call it an event* **A***:*

1. $h^{(t)} = l^{(t)}$*, where* $h^{(t)} = w^{(t)} \gg 4$ *and* $l^{(t)} = w^{(t)} \bmod 16$*;*
2. $w^{(t)} = w^{(t+1)} = \cdots = w^{(t+6)}$.

When the event **A** occurs at time t, the attacker guesses the values of $h^{(t)}$, $s_0^{(t)}$, $s_{12}^{(t)}$, $s_{14}^{(t)}$, $s_{15}^{(t)}$, $s_{25}^{(t)}$ and the values of the rightmost three bytes of each $R^{(t+i)}$, $0 \leq i \leq 5$, then recovers the values of all the rest inner states of LFSR and F. After all inner states of LFSR and F are recovered, the attacker runs Loiss for about another 128 ($= 2^7$) times and then can recover the values of all memory units of BOMM. Since the probability that the event **A** occurs is 2^{-52} and the attacker has to guess 188-bit inner states in the guessing stage, so the time complexity of the above attack method is $O(2^{247})$ and its data complexity is $O(2^{52})$.

4.2 Linear distinguishing attack

The linear distinguishing attack is a common attack on stream ciphers [14, 15]. Its basic idea is that: an attacker first constructs a linear distinguisher by means of linear approximations of the nonlinear part of an algorithm, and the linear distinguisher only depends on the key stream. When the bias of constructed linear distinguisher is significant, the attacker can distinguish key streams generated by the algorithm from a true random bit stream by means of the above distinguisher.

Now we consider the linear distinguishing attack on Loiss. The nonlinear part of Loiss includes F and BOMM. First we consider linear approximations of F. Assume that the inputs of F and the value of the memory unit R are IDU random variables within a short successive time interval. Since the linear branch number of the transformation θ is 5, thus the bias of arbitrary linear approximations of 2-round F is zero. Below we construct a linear approximation of 3-round F:

$$a \cdot (w^{(t)} \oplus s_{31}^{(t)}) \oplus c \cdot X^{(t+1)} \oplus b \cdot w^{(t+2)} = 0, \tag{4}$$

where a and b are 8-bit strings, c is a 32-bit string, and "$\cdot$" denotes the inner product of bit strings. The rationale of the linear approximation (4) is demonstrated in Figure 5.

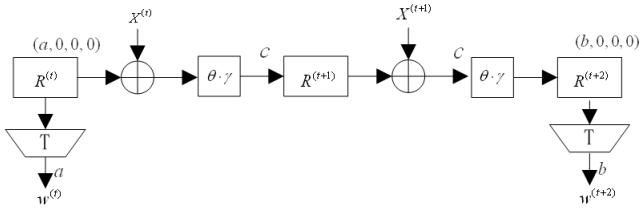

Fig. 5. Linear approximations of 3-round F

Going through all possible values of a, b and c, we can obtain that the maximum bias of the linear approximation (4) is $2^{-16.4}$.

Second we consider linear approximations of BOMM. Below we only consider the inputs $w^{(t)}, w^{(t+2)}$ and the outputs $v^{(t)}, v^{(t+2)}$ of BOMM at times t and $t+2$.

Definition 4. *For the inputs $w^{(t)}, w^{(t+2)}$ of BOMM at times t and $t+2$, if the equalities $h^{(t)} = l^{(t)} = h^{(t+2)} = h$ hold, where $h^{(t)} = w^{(t)} \gg 4$, $l^{(t)} = w^{(t)} \bmod 16$, $h^{(t+2)} = w^{(t+2)} \gg 4$, and the memory unit y_h is not updated at time $t+1$, then we call it an event* **B**.

By definition, the probability that the event **B** occurs is $\Pr(B) = 2^{-8}\left(\frac{15}{16}\right)^2$. When the event **B** occurs, we have

$$v^{(t)} \oplus v^{(t+2)} \oplus w^{(t)} \oplus w^{(t+2)} = S_2(w^{(t)}) \oplus S_2(v^{(t)} \oplus w^{(t)} \oplus S_2(w^{(t)})). \tag{5}$$

And we can construct the following linear approximation of equation (5):

$$a \cdot w^{(t)} \oplus b \cdot w^{(t+2)} = d \cdot v^{(t)} \oplus e \cdot v^{(t+2)}, \tag{6}$$

where a, b, d and e are 8-bit strings.

Going through all possible values of a, b, d and e, we have that the maximum bias of the linear approximation (6) is $2^{-15.2}$.

Combining the linear approximations (4) and (6), and going through all possible values of a, b, c, d and e, by the Piling-up Lemma [16] we obtain that the maximum bias of linear approximations of the nonlinear part of Loiss is $2^{-30.6}$, at the same time the bias of linear approximations of both F and BOMM reach the maximum possible value, that is, $2^{-16.4}$ and $2^{-15.2}$ respectively.

Suppose that an attacker has got a trinomial multiple of the characteristic polynomial $f(x)$ of LFSR with low degree whose all non-zero coefficients are one. Then by means of the above linear approximations, the attacker can construct a only key stream linear distinguisher with the maximum bias, whose bias is about $2^{3-1}(2^{-30.6})^3 = 2^{-89.8}$. By means of this linear distinguisher the attacker needs about 2^{180}-bit key stream to distinguish key streams generated by Loiss from a true random bit stream [16]. In fact, it is yet an open problem whether the attacker can obtain such a trinomial multiple with low degree whose all non-zero coefficients are one, which might not exist. Thus the above data complexity is in an optimistic case. This shows that Loiss has a good resistance against linear distinguishing attacks.

4.3 Algebraic attacks

The algebraic attack is a powerful attack to cryptosystems [17–19]. Its main idea is that: a cryptographic system can be viewed as an algebraic equation system, and then the problem about breaking the cryptographic system can be converted into the problem of solving the corresponding algebraic equation system.

As for Loiss, we will consider how to establish such an algebraic equation system and give a simple estimation on the time complexity of solving it.

First we consider F whose nonlinear part is only the S-box S_1. When F runs one time, the S-box S_1 will be called for four times. Since the algebraic immunity of S_1 is 2, and at most 39 linearly independent quadratic equations on the inputs and outputs of S_1 can be established, thus at most $39 \times 4 = 156$ linearly independent quadratic equations can be established when F runs for one time.

Second we consider BOMM. Here we introduce choosing functions $h_i(w)$ and $l_i(w)$, where $h_i(w)$ and $l_i(w)$ are boolean functions from F_{2^8} to F_2 and defined as follows:

$$h_i(w) = \begin{cases} 1, \text{ if } w \gg 4 = i, \\ 0, \text{ otherwise,} \end{cases}$$

$$l_i(w) = \begin{cases} 1, \text{ if } w \bmod 16 = i, \\ 0, \text{ otherwise,} \end{cases}$$

where $0 \leq i \leq 15$, and the intermediate variables $yl^{(t)}$, $yh^{(t)}$, $Sl^{(t)}$ and $Sh^{(t)}$ satifying

$$yl^{(t)} = y_0^{(t)} \cdot l_0(w^{(t)}) \oplus y_1^{(t)} \cdot l_1(w^{(t)}) \oplus \cdots \oplus y_{15}^{(t)} \cdot l_{15}(w^{(t)}), \tag{7}$$

$$yh^{(t)} = y_0^{(t)} \cdot h_0(w^{(t)}) \oplus y_1^{(t)} \cdot h_1(w^{(t)}) \oplus \cdots \oplus y_{15}^{(t)} \cdot h_{15}(w^{(t)}), \tag{8}$$

$$Sl^{(t)} = S_2(w^{(t)}), \tag{9}$$

$$Sh^{(t)} = S_2(yl^{(t)} \oplus Sl^{(t)}). \tag{10}$$

Then we can establish the update function of BOMM as follows:

$$y_i^{(t+1)} = y_i^{(t)} \oplus Sl^{(t)} l_i(w^{(t)}) \oplus Sh^{(t)} h_i(w^{(t)}), 0 \leq i \leq 15. \tag{11}$$

Note that the boolean functions $h_i(w)$ and $l_i(w)$ have an algebraic degree of 4, thus the algebraic degree of equations (7), (8) and (11) are 5. Therefore when BOMM updates for one time, we can obtain 144 equations of degree 5. In addition, Note that each time when BOMM is updated, the S-box S_2 is invoked for two times, see equations (9) and (10), since the algebraic immunity of S_2 is 2, and at most 36 linearly independent quadratic equations can be established, thus we can obtain 72 linearly independent quadratic equations. Finally by the output equation $v^{(t)} = yh^{(t)} \oplus w^{(t)}$ of BOMM, we can obtain 8 linear equations. Totally during one time update of BOMM we obtain 72 linearly independent quadratic equations and 144 equations of degree 5, and introduce 152 ($= 3 \times 8 + 16 \times 8$) independent intermediate bit variables.

Assume that the inner states of Loiss at time t are IDU random variables. Since Loiss has 416 bits of inner states (LFSR:256+F:32+BOMM:128), an attacker needs at least 52 (=416/8) byte of key stream to possibly establish an over-defined algebraic equation system. Similarly to the above process, we can establish an entire algebraic equation system for Loiss with 52 key stream bytes, which contains 9800 variables (including introduced intermediate variables) and 18980 equations. The highest degree of the entire algebraic equation system is 5.

When applying the normal linearization to solve the above algebraic equation system, the number of linear equations is much less than that of variables because the linearization process introduces many more intermediate variables. Therefore the normal linearization method cannot be directly used to solve it. Here we use the XL method [20] to estimate the time complexity of solving the above algebraic equation system, and obtain that its time complexity is about $2^{2420.88}$, which is much higher than the time complexity of a brute force attack. Therefore Loiss has good resistance against algebraic attacks.

4.4 Time-memory-and-data attack

The time-memory-and-data attack is a basic method in computer science [21]. Its main idea is that: reduce the cost of space by sacrifice the cost in time, or vice versa. In analyzing stream ciphers, the data is also taken into consideration.

More precisely, denote by D, T and M respectively the number of data (plaintext-ciphertext pairs) an attacker can get, the time complexity of the attack, and the size of memory required to perform an attack. Then D, T and M satisfy $TM^2D^2 = N^2$ and $N > T \geq D^2$, where N denote the size of the space of unknowns that the attack targets to recover. Normally we assume the number of data got by an attacker reaches the upper bound, that is, $T = D^2$, then we have $TM = N$.

With respect to the time-memory-and-data attack, since Loiss contains a total of 416 bits of unknowns, thus $N = 2^{416}$. It follows that $TM = 2^{416}$. This shows that at least one of T and M is no less than 2^{208}. So Loiss have good resistance against the time-memory-and-data attack.

5 Evaluations on software and hardware implementations

5.1 On software implementation

Since Loiss's basic operators are byte-oriented, thus a software implementation can make it very fast, and the software implementation only needs small memory and small size of code, and hence the algorithm can be used in resource limited environments, e.g., in smart cards. Compared to the well-known byte-oriented block cipher AES [4], the encryption speed of Loiss is almost the same as that of 128-AES (whose keys has a length of 128) in the counter mode. Below we give a simple performance evaluation on software implementations on a 32-bit common PC according to the Intel 486 32-bit instruction set, and compare it with that of SNOW 3G, see Table 1.

Table 1. Performance of software implementation of parts of Loiss (Unit: Cycles)

Algorithm Name	LFSR	F	BOMM	Initialization	Generate single key
Loiss	13	21	14	3223	48
SNOW 3G	14	31		1508	46

Note: From the above table it's seen that the speed of generating single key by Loiss is almost the same as that of SNOW 3G. But since SNOW 3G generates a 32-bit word one time, thus totally the speed of generating keystream by SNOW 3G is three times faster than the one by Loiss in common length.

5.2 On hardware implementation

There are different approaches in hardware implementation. As an approach of hardware implementation of Loiss, we give a rough estimation on the size of electric circuits needed in implementing each part of Loiss, where the number of

gates can further be optimized, see Table 2. In addition, we also give a simple comparison of the sizes of electric circuits in hardware implementations of both Loiss and SNOW 3G, see Table 3.

Table 2. Size of electric circuits in hardware implementation of Loiss

Units	Num. of units	Num. of Gates
8-bit register	48	$80 \times 48 = 3850$
32-bit register	1	$320 \times 1 = 320$
multiplication by α and α^{-1}	4	$10 \times 4 = 40$
S-boxes	6	$500 \times 6 = 3000$
8-bit XOR	12	$22 \times 12 = 264$
32-bit XOR	5	$86 \times 5 = 430$
two-choose-one logic	5	$8 \times 5 = 40$
total	7934	

Table 3. Comparison of sizes of electric circuits in hardware implementation of Loiss and SNOW 3G

	Loiss	SNOW 3G
Size	7934	10908

6 Conclusions

In this paper we present a byte-oriented stream cipher Loiss. The Loiss algorithm has good performance of software and hardware implementations, and is suitable for a variety of software and hardware implementation requirements. Loiss has good properties in resisting against many known attacks, including guess and determine attack, linear distinguishing attack, algebraic attack, time-memory-and-data attack, and fast correlation attack, and can offer the 128-bit-level security. In the design of Loiss, an orthomorphic permutation is used which is a necessity to ensure the balance of the BOMM component, which is expected to motivate the research on the cryptographic properties of orthomorphic permutations.

Acknowledgement

During the design of Loiss, a large number of graduate students from the State Key Laboratory of Information Security, Chinese Academy of Sciences, have made significant contributions. Those students are highly appreciated.

References

1. ETSI/SAGE, SNOW 3G Specification, Specification of the 3GPP Confidentiality and Integrity Algorithms UEA2 & UIA2, Document 2, September 2006.
2. eSTREAM, ECRYPT Stream Cipher Project. http://www.ecrypt.eu.org /stream.
3. R.L. Rivest, The RC4 encryption algorithm, RSA Data Security, Inc., March 1992.
4. FIPS PUB 197, the official AES standard, http://csrc.nist.gov/publications /fips/fips197/fips-197.pdf.
5. L. Mittenthal, Block substitutions using orthomorphic mappings, Advances in Applied Mathematics, 16(1), pp.59-71, 1995.
6. S.W Lv, X.B Fan, Z.S Wang, J.L Xu and J. Zhang, Completing mappings and their appliactions, University of Sciences and Technology of China Press, 2008.
7. S. Vaudenay, On the Lai-Massey scheme, ASIACRYPT'99, LNCS1716, pp.8-19, 1999.
8. Chinese State Bureau of Cryptography Administration, Cryptographic algorithms SMS4 used in wireless LAN products, http://www.oscca.gov.cn/Doc/6/News_1106.htm.
9. S.W. Golomb and G. Gong, Signal design for good correlation for wireless communication, cryptography and radar, Cambridge University Press, 2004.
10. K. Zeng and H. Huang, On the linear syndrome method in cryptanalysis, EUROCRYPT'88, pp.469-478, 1990.
11. T. Siegenthaler, Correlation-immunity of nonlinear combining functions for cryptographic applications, IEEE Transaction on Information Theory, IT-30, pp.776-780, 1984.
12. C. Canniere, Guess and Determine Attack on SNOW, NESSIE Public Document, NES/DOC/KUL/WP5/011/a, 2001.
13. P. Hawkes and G.G. Rose, Guess and Determine Attacks on SNOW, Qualcomm Australia, SAC 2002, LNCS 2595, pp.37-46, 2002.
14. D. Watanabe, A. Biryukov, C. Canniere, A distinguishing attack of SNOW 2.0 with linear masking method, In M.Matsui and R.Zuccherato eds. Selected Areas in Cryptography 2003, Lecture Notes in Computer Science 3006, pp.222-233, Springer-Verlag Berlin Heidelberg 2004.
15. D. Coppersmith, S. Halevi, C. Jutla, Cryptanalysis of stream ciphers with linear masking, In M.Yung ed. CRYPTO 2002, LNCS 2442, pp.515-532, Springer-Verlag Berlin Heidelberg 2002.
16. M. Matsui, Linear cryptanalysis of the fast data encipherment algorithm, In: Advances in Cryptology, EUROCRYPT'93 proceeding, pp.386-397, 1994.
17. N.T. Courtois and W. Meier, Algebraic Attacks on Stream Ciphers with Linear Feedback, In Advances in Cryptology-EUROCRYPT 2003, LNCS 2656, pp.346-359, 2003.

18. S. Ronjom, T. Helleseth, Attacking the Filter Generator over GF(2^m), in Workshop Record of SASC 2007: The State of the Art of Stream Ciphers, eSTREAM report 2007/011 (2007).
19. W. Meier, E. Pasalic, and C. Carlet, Algebraic attacks and decomposition of Boolean functions, In Advances in Cryptology-EUROCRYPT2004, LNCS 3027, pp.474 - 491, 2004.
20. C. Diem, The XL-Algorithm and a Conjecture from Commutative Algebra, In Pil Joong Lee, editor, Advances in Cryptology-ASIACRYPT2004, LNCS 3329, pp.323-337, 2004.
21. M.E. Hellman, A cryptanalytic time-memory tradeoff, IEEE Transactions on Information Theory, vol.26, pp.401-406, 1980.

Appendix A: The S-boxes S_1 and S_2

For an S-box S of size 8×8 which can be S_1 or S_2, let $x \in F_{2^8}$ and $h = x \gg 4$ and $l = x \bmod 16$. Then $S(x)$ is the element at the intersection of the h-th row and the l-th column in Tables 4 or 5. For example, $S_1(0x3A) = 0xBF$.

Note: Data in Table 4 and 5 are expressed in the hexadecimal format.

Table 4. The S-box S_1

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	55	C2	63	71	3B	C8	47	86	9F	3C	DA	5B	29	AA	FD	77
1	8C	C5	94	0C	A6	1A	13	00	E3	A8	16	72	40	F9	F8	42
2	44	26	68	96	81	D9	45	3E	10	76	C6	A7	8B	39	43	E1
3	3A	B5	56	2A	C0	6D	B3	05	22	66	BF	DC	0B	FA	62	48
4	DD	20	11	06	36	C9	C1	CF	F6	27	52	BB	69	F5	D4	87
5	7F	84	4C	D2	9C	57	A4	BC	4F	9A	DF	FE	D6	8D	7A	EB
6	2B	53	D8	5C	A1	14	17	FB	23	D5	7D	30	67	73	08	09
7	EE	B7	70	3F	61	B2	19	8E	4E	E5	4B	93	8F	5D	DB	A9
8	AD	F1	AE	2E	CB	0D	FC	F4	2D	46	6E	1D	97	E8	D1	E9
9	4D	37	A5	75	5E	83	9E	AB	82	9D	B9	1C	E0	CD	49	89
A	01	B6	BD	58	24	A2	5F	38	78	99	15	90	50	B8	95	E4
B	D0	91	C7	CE	ED	0F	B4	6F	A0	CC	F0	02	4A	79	C3	DE
C	A3	EF	EA	51	E6	6B	18	EC	1B	2C	80	F7	74	E7	FF	21
D	5A	6A	54	1E	41	31	92	35	C4	33	07	0A	BA	7E	0E	34
E	88	B1	98	7C	F3	3D	60	6C	7B	CA	D3	1F	32	65	04	28
F	64	BE	85	9B	2F	59	8A	D7	B0	25	AC	AF	12	03	E2	F2

Table 5. The S-box S_2

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	61	97	FF	E9	66	56	F1	F3	54	72	CC	4D	85	52	7A	70
1	D0	2E	4C	58	BE	88	7F	5A	2F	1B	47	AF	9B	D5	BF	81
2	C3	4E	86	2D	6A	9C	CE	20	2B	53	6D	FD	3C	BC	33	22
3	F7	59	C9	63	6E	8D	DD	F2	E3	1A	75	DA	13	1D	68	42
4	A4	3F	B7	46	90	12	73	EB	FA	F6	09	40	A5	E0	B4	B1
5	51	8E	06	34	7D	DF	99	6F	AA	0B	80	95	25	EA	87	CD
6	DC	0C	43	FB	A7	BD	9E	FC	EE	9F	74	B6	CF	EF	16	0F
7	78	D1	92	64	D6	84	48	41	08	60	5D	2A	B8	4F	E2	69
8	01	C1	31	5F	62	49	B2	93	00	CB	04	18	07	71	17	E4
9	AC	8B	B0	7E	F8	44	5B	AD	98	A0	27	4B	3A	B5	F0	83
A	F9	14	E7	23	77	D2	10	AE	B3	36	30	3B	1C	03	82	38
B	0E	7B	50	A6	1F	7C	CA	C2	02	2C	A9	8A	39	15	F4	D9
C	A3	55	32	96	C8	8C	C0	05	67	1E	EC	19	29	89	F5	21
D	37	BB	E1	57	A2	C7	E6	8F	AB	91	35	28	D3	D7	79	BA
E	A1	6C	B9	DE	A8	5E	FE	6B	C5	ED	65	9A	45	C6	C4	9D
F	94	24	0D	0A	E5	76	3D	E8	26	5C	D4	4A	D8	11	DB	3E