NOLTA, IEICE

Anything really related to chaos??? except non linearity???

Paper

Further improving security of Vector Stream Cipher

Atsushi Iwasaki [1 a)] *and Ken Umeno* [2]

[1] *Fukuoka Institute of Technology*
Wajiro-higashi, Higashiku, Fukuoka 811-0295, Japan

[2] *Graduate school of Informatics, Kyoto University*
Yoshida-honmachi, Sakyo-ku, Kyoto city 606-8501, Japan

[a)] *a-iwasaki@fit.ac.jp*

Received October 10, 2016; Revised January 17, 2017; Published July 1, 2017

Abstract: Vector Stream Cipher (VSC) is a stream cipher which consists of permutation polynomial over a ring of modulo 2^w. The algorithm for generating key stream is very simple and the encryption is very fast. Some theoretical attacks for VSC have been reported so far since the invention of VSC in 2004. Then, the authors proposed some improvements and developed "Vector Stream Cipher 2.0 (VSC 2.0)" to be immune against the theoretical attacks. In this paper, we propose further improvement of VSC 2.0 to publish as a new chaos cipher "Vector Stream Cipher 2.1 (VSC2.1)". VSC 2.1 is faster and more secure than VSC 2.0. Our result suggests that permutation polynomials over a ring of modulo 2^w are useful for cryptography.

Key Words: stream cipher, chaos, pseudorandom number, permutation polynomial

1. Introduction

Vector Stream Cipher (VSC) is a stream cipher which was developed by one of the author at Communication Research Laboratory (now called the National Institute of Information and Communications Technology) in 2004 [1]. Although stream cipher is usually constructed by shift register and other parts based on finite field theory, VSC is constructed by permutation polynomials over a ring of modulo 2^w [2] and so is classified in chaos cipher. For digital computers and digital signal processors, a modulo 2^w operation is practically negligible. Therefore, the values of permutation polynomials over a ring of modulo 2^w can be calculated very fast, and so VSC is very fast. The property of the polynomials is also useful for light implementation of VSC.

On the other hand, the security of VSC has been investigated by several researchers [3–6], and some security problems against for the theoretical attacks were reported though any practical attack breaking VSC has not been reported so far [4–6]. (In this paper, "theoretical attack" means an attack, which currently needs too much computation and so cannot practically break cipher with today's computers. Thus, a cipher which is proven to be immune against theoretical attacks can be said to have a certain provable security, while the success of a theoretical attack leads to a potential risk to be attacked with future's computers. On the other hand, "practical attack" means an attack which can practically break cipher with today's computers.) In order to avoid the problems, the

authors proposed in 2016 some improving of VSC and developed a new algorithm "Vector Stream Cipher 2.0 (VSC 2.0)" [7]. Although VSC 2.0 is more secure than the original VSC, something to be optimized is left. In this paper, we propose further improving of VSC as a result of optimization and set it a new cipher called "Vector Stream Cipher 2.1".

Permutation polynomials over a ring of modulo 2^w have not been applied for cryptography and not been studied except some applications and research [2, 8–12]. Our purpose of this research is not only to develop a new cipher but also to explore possibility of the permutation polynomials and chaos in the field of cryptography.

This paper is constructed as follows. In sections 2 and 3, we introduce the algorithm of the original VSC and VSC 2.0, respectively. In section 4, we propose improvements for VSC and VSC 2.0 and describe the optimized design as a new cipher "Vector Stream Cipher 2.1". In section 5, we perform some experiments to investigate the security of the methods. Finally, we conclude this paper.

2. Vector Stream Cipher

In this section, we introduce Vector Stream Cipher 128 (VSC128) [1] which is one kind of original VSC. VSC128 requires 128-bit secret key and 128-bit initial vector. The encryption algorithm is as follows:

1. Assume that A, B, C, D, X, Y, Z and W are 32-bit integer variables. Assign a secret key to A, B, C and D, and an initial vector to X, Y, Z and W.

2. Repeat the following operation 8 times. (In this paper, we call the operation "round".)

   (a) Assume that a, b, c, d, x, y, z and w are 32-bit integer variables. Calculate the values of a, b, c, d, x, y, z and w as follows.

$$\begin{aligned}
a &= A - (A \mod 4) + 1 \mod 2^{32},\\
b &= B - (B \mod 4) + 1 \mod 2^{32},\\
c &= C - (C \mod 4) + 1 \mod 2^{32},\\
d &= D - (D \mod 4) + 1 \mod 2^{32},\\
x &= X - (X \mod 4) + 1 \mod 2^{32},\\
y &= Y - (Y \mod 4) + 1 \mod 2^{32},\\
z &= Z - (Z \mod 4) + 1 \mod 2^{32},\\
w &= W - (W \mod 4) + 1 \mod 2^{32}.
\end{aligned}$$

   In this paper, we regard "mod" as modulus operator.

   (b) Assume that A', B', C', D', X', Y', Z' and W' are 32-bit integer variables. Calculate the values of A', B', C', D', X', Y', Z' and W' as follows.

$$\begin{aligned}
A' &= A(2A + y) \mod 2^{32},\\
B' &= B(2B + x) \mod 2^{32},\\
C' &= C(2C + z) \mod 2^{32},\\
D' &= D(2D + w) \mod 2^{32},\\
X' &= X(2X + c) \mod 2^{32},\\
Y' &= Y(2Y + d) \mod 2^{32},\\
Z' &= Z(2Z + a) \mod 2^{32},\\
W' &= W(2W + b) \mod 2^{32}.
\end{aligned}$$

   (c) Regard $(A', B', C', D', X', Y', Z', W')$ as a 256-bit sequence, and perform 5-bit left rotational shift. After that, copy the sequence to (A, B, C, D, X, Y, Z, W). Writing mathematically,

$$A = (A' << 5) \oplus (B' >> 27) \mod 2^{32}, \quad B = (B' << 5) \oplus (C' >> 27) \mod 2^{32},$$
$$C = (C' << 5) \oplus (D' >> 27) \mod 2^{32}, \quad D = (D' << 5) \oplus (X' >> 27) \mod 2^{32},$$
$$X = (X' << 5) \oplus (Y' >> 27) \mod 2^{32}, \quad Y = (Y' << 5) \oplus (Z' >> 27) \mod 2^{32},$$
$$Z = (Z' << 5) \oplus (W' >> 27) \mod 2^{32}, \quad W = (W' << 5) \oplus (A' >> 27) \mod 2^{32}.$$

Here, "<<" means simple bit shift.

3. Assume that $D1$, $D2$, $D3$ and $D4$ are 32-bit plaintexts and $E1$, $E2$, $E3$ and $E4$ are the corresponding ciphertexts respectively. Then, calculate the values of $E1$, $E2$, $E3$ and $E4$ as follows.

$$E1 = D1 \oplus X,$$
$$E2 = D2 \oplus Y,$$
$$E3 = D3 \oplus Z,$$
$$E4 = D4 \oplus W.$$

4. Repeat step 2 and 3 until all the given plaintexts are encrypted.

3. VSC 2.0

There are some security problems of VSC128. We think that the following three points are the most important problems.

- The maximum linear characteristic probability of VSC128 is 2^{-115}. Therefore, distinguishing attack with linear masking is practical [4, 5].
- It is reported that the output sequence (key-stream) of VSC128 have a statistical deviation if the initial vector is chosen among specific vectors, and so the distinguishing attack is realized if an attacker can chose an initial vector intentionally [6].
- The round of VSC128 is not a bijection. Then, the effective key length of VSC128 is smaller than 128bit.

To solve the above problems, we proposed "Vector Stream Cipher 2.0 (VSC 2.0)" [7], which has the following three improvements.

- Changing the iteration number of the round from 8 to 9. Herewith, the maximum linear characteristicprobability changes from 2^{-115} to 2^{-129}, and so a distinguishing attack with linear masking becomes not practical.
- Adding a preprocessing which replaces a given initial vector with another value like a hash value. Herewith, an attacker cannot choose an initial vector intentionally.
- Introducing a new rule "Keep the value of the variable D is even". Herewith, by the following Theorem 1, the round becomes a bijection. In order to keep the new rule, the key length and step 2(c) of the VSC128 algorithm change.

Theorem 1. Consider a map $g : (\mathbb{Z}/2^n\mathbb{Z})^m \to (\mathbb{Z}/2^n\mathbb{Z})^m$, which is described as

$$g(A_0, A_1, \cdots, A_{m-1}) = (A'_0, A'_1, \cdots, A'_{m-1}),$$
$$A'_i = A_i \left(2A_i + a\left(A_{(i+1 \mod m)}\right)\right) \mod 2^n \quad (^\forall i \in \mathbb{Z}/m\mathbb{Z}),$$

where $A_1, \cdots, A_m$ and $A'_1, \cdots, A'_m$ are elements of $\mathbb{Z}/2^n\mathbb{Z}$ and $a(A_i) = A_i - (A_i \mod 4) + 1$ ($^\forall i \in \mathbb{Z}/m\mathbb{Z}$). Assume that O_n is a subset of $\mathbb{Z}/2^n\mathbb{Z}$, which is constructed of odd numbers in $\mathbb{Z}/2^n\mathbb{Z}$. Then, if we restrict the domain of g to $(\mathbb{Z}/2^n\mathbb{Z})^m$ except $(O_n)^m$, g become a bijective map on $(\mathbb{Z}/2^n\mathbb{Z})^m \backslash (O_n)^m$.

If you would like to know the proof of Theorem 1, see [7].

The algorithm of VSC 2.0 which is based on the above is as follows:

1. Assume that A, B, C, D, X, Y, Z and W are 32-bit integer variables. Set A=0xfedcba98, B=0x01234567, C=0x89abcdef and D=0x76543210 and assign an initial vector to X, Y, Z and W. (In actuality, we can choose values of A, B, C and D more freely. Since the specification must be fixed, the values, A=0xfedcba98, B=0x01234567, C=0x89abcdef and D=0x76543210, are for the moment chosen.)

2. Repeat the following operation 30 times. (The operation is the "round" of VSC 2.0.)

   (a) Assume that a, b, c, d, x, y, z and w are 32-bit integer variables. Calculate the values of a, b, c, d, x, y, z and w as follows.

$$\begin{aligned}
a &= A - (A \mod 4) + 1 \mod 2^{32},\\
b &= B - (B \mod 4) + 1 \mod 2^{32},\\
c &= C - (C \mod 4) + 1 \mod 2^{32},\\
d &= D - (D \mod 4) + 1 \mod 2^{32},\\
x &= X - (X \mod 4) + 1 \mod 2^{32},\\
y &= Y - (Y \mod 4) + 1 \mod 2^{32},\\
z &= Z - (Z \mod 4) + 1 \mod 2^{32},\\
w &= W - (W \mod 4) + 1 \mod 2^{32}.
\end{aligned}$$

   (b) Assume that A', B', C', D', X', Y', Z' and W' are 32-bit integer variables. Calculate the values of A', B', C', D', X', Y', Z' and W' as follows.

$$\begin{aligned}
A' &= A(2A + y) \mod 2^{32},\\
B' &= B(2B + x) \mod 2^{32},\\
C' &= C(2C + z) \mod 2^{32},\\
D' &= D(2D + w) \mod 2^{32},\\
X' &= X(2X + c) \mod 2^{32},\\
Y' &= Y(2Y + d) \mod 2^{32},\\
Z' &= Z(2Z + a) \mod 2^{32},\\
W' &= W(2W + b) \mod 2^{32}.
\end{aligned}$$

   (c) Regard $(A', B', C', D', X', Y', Z', W')$ as a 256-bitssequence, and perform 5-bit left rotational shift. After that, copy the sequence to (A, B, C, D, X, Y, Z, W). After that, 1-bit left rotational shift for low-ranking 6-bit of D. Writing mathematically,

$$\begin{aligned}
&A = (A' << 5) \oplus (B' >> 27) \mod 2^{32}, \quad B = (B' << 5) \oplus (C' >> 27) \mod 2^{32},\\
&C = (C' << 5) \oplus (D' >> 27) \mod 2^{32}, \quad D = (D' << 5) \oplus ((X' >> 27) << 1)) \mod 2^{32},\\
&X = (X' << 5) \oplus (Y' >> 27) \mod 2^{32}, \quad Y = (Y' << 5) \oplus (Z' >> 27) \mod 2^{32},\\
&Z = (Z' << 5) \oplus (W' >> 27) \mod 2^{32}, \quad W = (W' << 5) \oplus (A' >> 27) \mod 2^{32}.
\end{aligned}$$

3. Assign a secret key to A, B, C and D except the least significant bit of D. Set the least significant bit of D to 0.

4. Perform the round 9 times.

5. Assume that $D1$, $D2$, $D3$ and $D4$ are 32-bit plaintexts and $E1$, $E2$, $E3$ and $E4$ are the corresponding ciphertexts respectively. Then, calculate the values of $E1$, $E2$, $E3$ and $E4$ as follows.

$$\begin{aligned}
E1 &= D1 \oplus X,\\
E2 &= D2 \oplus Y,\\
E3 &= D3 \oplus Z,\\
E4 &= D4 \oplus W.
\end{aligned}$$

6. Repeat step 4 and 5 until all the given plaintexts are encrypted.

4. Further improvements of VSC 2.0

Although the security of VSC 2.0 is better than the original VSC, something to be considered for optimization of cipher design is left. In particular, we think that the following two things should be considered for further improvements.

- The key length of VSC 2.0 is 127bit. Thus, the key space is the half of the original VSC128.
- The step 2(c) of VSC 2.0 algorithm is slower than that of VSC128.

Both of them are caused by what makes the round a bijection. To improve them, we introduce the following theorem.

Theorem 2. Assume that $g : (\mathbb{Z}/2^n\mathbb{Z})^m \to (\mathbb{Z}/2^n\mathbb{Z})^m$ is described as

$$\begin{aligned} &g(A_0, A_1, \cdots, A_{m-1}) = (A'_0, A'_1, \cdots, A'_{m-1}),\\ &A'_i = A_i \left(2A_i + 4A_{(i+1 \mod m)} + 1\right) \mod 2^n \quad (^\forall i \in \mathbb{Z}/m\mathbb{Z}). \end{aligned}$$

Here, $A_1, \cdots, A_m$ and $A'_1, \cdots, A'_m$ are elements of $\mathbb{Z}/2^n\mathbb{Z}$. Then, g is a bijection.

Proof. It is enough to prove injectivity. Assume that $(A_0, A_1, \cdots, A_{m-1})$ and $(\tilde{A_0}, \tilde{A_1}, \cdots, \tilde{A_{m-1}})$ are elements of $(\mathbb{Z}/2^n\mathbb{Z})^m$ satisfying

$$(A_0, A_1, \cdots, A_{m-1}) \neq (\tilde{A_0}, \tilde{A_1}, \cdots, \tilde{A_{m-1}}).$$

There are non-negative numbers s_i $(i \in \mathbb{Z}/m\mathbb{Z})$ satisfying

$$\begin{aligned} &s_i \leq n \quad (^\forall i \in \mathbb{Z}/m\mathbb{Z}),\\ &(s_0, s_1, \cdots, s_{m-1}) \neq (n, n, \cdots, n),\\ &\tilde{A_i} = A_i + (2p_i - 1)2^{s_i} \mod 2^n \quad (^\forall i \in \mathbb{Z}/m\mathbb{Z}), \end{aligned}$$

where p_i are natural numbers. Assume that

$$\begin{aligned} (A'_0, A'_1, \cdots, A'_{m-1}) =& g(A_0, A_1, \cdots, A_{,-1}),\\ (\tilde{A_0}', \tilde{A_1}', \cdots, \tilde{A}'_{m-1}) =& g(\tilde{A_0}, \tilde{A_1}, \cdots, \tilde{A}_{m-1}). \end{aligned}$$

Then, there exists $k \in \mathbb{Z}/m\mathbb{Z}$ such that $s_k \leq s_i$ $(^\forall i \in \mathbb{Z}/m\mathbb{Z})$. Obviously, $s_k < n$. To simplify notation, we define new symbols A_m, $\tilde{A_m}$, s_m and p_m as A_0, $\tilde{A_0}$, s_0 and p_0, respectively. Then,

$$\begin{aligned} &\tilde{A_k}'\\ &= \tilde{A_k}\{2\tilde{A_k} + 4\tilde{A}_{k+1} + 1\} \mod 2^n\\ &= \{A_k + (2p_k - 1)2^{s_k}\}\{2A_k + (2p_k - 1)\cdot 2^{s_k+1} + 4A_{k+1} + (2p_{k+1} - 1)\cdot 2^{s_{k+1}+2} + 1\} \mod 2^n\\ &= A'_k + (2p_k - 1)\cdot 2^{s_k} + r \cdot 2^{s_k+1} + r' \cdot 2^{s_{k+1}+2} \mod 2^n. \end{aligned}$$

Here, r and r' are natural numbers. Since $s_k \leq s_{k+1}$ and $s_k < n$, $\tilde{A_k}' \neq A'_k$. From the above, g is a bijection. □

By using Theorem 2, we can further improve VSC 2.0. We propose a new cipher "Vector Stream Cipher 2.1 (VSC 2.1)". The algorithm proposed here as VSC 2.1 is as follows:

1. Assume that A, B, C, D, X, Y, Z and W are 32-bit integer variables. Set A=0xfedcba98, B=0x01234567, C=0x89abcdef and D=0x76543210 and assign an initial vector to X, Y, Z and W.
2. Repeat the following operation 30 times. (The operation is the "round" of VSC 2.1.)

(a) Assume that a, b, c, d, x, y, z and w are 32-bit integer variables. Calculate the values of a, b, c, d, x, y, z and w as follows.

$$\begin{aligned}
a &= 4A + 1 \mod 2^{32},\\
b &= 4B + 1 \mod 2^{32},\\
c &= 4C + 1 \mod 2^{32},\\
d &= 4D + 1 \mod 2^{32},\\
x &= 4X + 1 \mod 2^{32},\\
y &= 4Y + 1 \mod 2^{32},\\
z &= 4Z + 1 \mod 2^{32},\\
w &= 4W + 1 \mod 2^{32}.
\end{aligned}$$

(b) Assume that A', B', C', D', X', Y', Z' and W' are 32-bit integer variables. Calculate the values of A', B', C', D', X', Y', Z' and W' as follows.

$$\begin{aligned}
A' &= A(2A + y) \mod 2^{32},\\
B' &= B(2B + x) \mod 2^{32},\\
C' &= C(2C + z) \mod 2^{32},\\
D' &= D(2D + w) \mod 2^{32},\\
X' &= X(2X + c) \mod 2^{32},\\
Y' &= Y(2Y + d) \mod 2^{32},\\
Z' &= Z(2Z + a) \mod 2^{32},\\
W' &= W(2W + b) \mod 2^{32}.
\end{aligned}$$

(c) Regard $(A', B', C', D', X', Y', Z', W')$ as a 256-bit sequence, and perform 5-bit left rotational shift. After that, copy the sequence to (A, B, C, D, X, Y, Z, W). Writing mathematically,

$$\begin{aligned}
&A = (A' << 5) \oplus (B' >> 27) \mod 2^{32}, \quad B = (B' << 5) \oplus (C' >> 27) \mod 2^{32},\\
&C = (C' << 5) \oplus (D' >> 27) \mod 2^{32}, \quad D = (D' << 5) \oplus (X' >> 27)) \mod 2^{32},\\
&X = (X' << 5) \oplus (Y' >> 27) \mod 2^{32}, \quad Y = (Y' << 5) \oplus (Z' >> 27) \mod 2^{32},\\
&Z = (Z' << 5) \oplus (W' >> 27) \mod 2^{32}, \quad W = (W' << 5) \oplus (A' >> 27) \mod 2^{32}.
\end{aligned}$$

3. Assign a secret key to A, B, C and D.

4. Perform the round 9 times.

5. Assume that $D1$, $D2$, $D3$ and $D4$ are 32-bit plaintexts and $E1$, $E2$, $E3$ and $E4$ are the corresponding ciphertexts respectively. Then, calculate the values of $E1$, $E2$, $E3$ and $E4$ as follows.

$$\begin{aligned}
E1 &= D1 \oplus X,\\
E2 &= D2 \oplus Y,\\
E3 &= D3 \oplus Z,\\
E4 &= D4 \oplus W.
\end{aligned}$$

6. Repeat step 4 and 5 until all the given plaintexts are encrypted.

By Theorem 2, the round of VSC 2.1 is a bijection. The maximum linear probability of VSC 2.1 is the same as that of VSC 2.0. The key length of VSC 2.1 is 128bit, it is longer than that of VSC 2.0. VSC 2.1 is expected to be faster than VSC 2.0 because step 2(c) is more simple than that of VSC 2.0.

5. Experiments

In this section, we perform some experiments for VSC 2.1.

5.1 Speed

We measure the speeds of performing VSC128, VSC 2.0, VSC 2.1 and AES-128. The environment in which we measure is shown in Table I. As results, we got Table II. VSC2.1 is slightly slower than the original VSC128, but faster than VSC 2.0.

Table I. Environment on which we measure speed.

CPU	1.3 GHz Intel Core i5
Memory	4 GB 1600 MHz DDR3
OS	OS X 10.9.5 (13F34)
Compiler	gcc 4.2.1
Optimization option	-Ofast

Table II. Encryption speeds.

Algorithm	Speed(Mbps)
VSC128	1202.254889
VSC 2.0	1039.222464
VSC 2.1	1113.193866
AES-128 ECB	366.901621

5.2 Property of the preprocessing

We investigate properties of the preprocessing of VSC 2.0 and VSC 2.1. Step 2 of VSC 2.0 algorithm and that of VSC 2.1 algorithm are preprocessing, respectively. The detail of the experiment is as follows:

1. Select an input randomly. (We call the input I_1.)
2. Select a bit of I_1 and reverse the bit. (We call the value I_2.)
3. Apply the preprocessing to I_1 and I_2. (We call the outputs I_1' and I_2' respectively.)
4. Measure the Hamming distance between I_1' and I_2'.
5. Repeat step 1-4 1000000 times. Calculate the average of the Hamming distance which are measure at step 3.

As a result, we got Table III. Since the output length of the both preprocessing are 128bit, the result shows that the preprocessings have a good property.

Table III.

Algorithm	Average Hamming distance
VSC 2.0	64.000107
VSC 2.1	63.995965

5.3 Randomness of key stream

We performed randomness test described by NIST SP800-22 [13] for key streams generated by VSC128, VSC 2.0 and VSC 2.1. The test was performed for 11 sets. Each set is constructed of 1000 sequences. (Exceptionally, the sets 10 and 11 are constructed of 255 sequences respectively. A sequence of the set 10 is generated with an initial condition (key and initial vector) whose one bit is “1” and the

others are “0”. VSC 2.0 requires the least bit of D is “0”. Then, the set 10 is constructed of only 255 sequences. A sequence of the set 11 is generated with an initial condition whose two bits are “0” and the others are “1”. One of the two is the least bit of D. Then, set 11 is also constructed of only 255 sequences.) Each sequence is constructed of 1000000 bits, which are generated by VSC128, VSC 2.0 or VSC 2.1 with a secret key and an initial vector. The secret key and the initial vector are chosen randomly, but random pattern is dependent on a set. Table IV shows the result. The randomness test is constructed of 188 test items. Even if sequences are exactly random, there are cases that the sequence does not pass all test items. Therefore, the result show that there are no problem about randomness of the key stream of VSC128, VSC 2.0 and VSC 2.1.

Table IV. Results of randomness tests.

Set No.	Numbers of test items which the set passed		
	VSC128	VSC 2.0	VSC 2.1
1	188	188	187
2	188	187	188
3	188	186	188
4	187	188	188
5	187	188	188
6	188	187	188
7	188	187	188
8	188	188	187
9	188	188	188
10	188	188	188
11	188	188	188

6. Conclusion

We proposed further improving of VSC as a certain optimization of cipher design. Our proposed VSC 2.1 is *faster* than VSC 2.0. We think that VSC 2.1 is *more secure* than VSC 2.0 because of the following two reasons. First is that the key length of VSC 2.1 is longer than that of VSC 2.0. Second is that any theoretical attacks which were reported as workable attacks for the original VSC128 are not workable for VSC 2.1. In particular, VSC 2.1 has the *provable security* for the distinguishing attack with linear masking. Thus, VSC 2.1 is a precious example of chaotic cipher which uses permutation polynomials over a ring of modulo 2^w, and suggests that the permutation polynomials over a ring of modulo 2^w and chaos theory designing unpredictability are particularly useful for cryptography.

References

[1] K. Umeno, S. Kim, and A. Hasegawa, “128bit VSC specification,” http://www.chaosware.com/vsc128.pdf, 2004.

[2] R.L. Rivest, “Permutation polynomials modulo 2^w,” *Finite Fields and their Applications*, vol. 7, pp. 287–292, 2001.

[3] H. Tanaka, K. Nemoto, T. Miki, and M. Sato, “Security evaluation of 128bit VSC,” *Technical Report of IEICE*, ISEC2003-144, pp. 179–184, 2004.

[4] M. Nakamura, T. Nosaka, and T. Kaneko, “A Study on the linear property of basic function in VSC128,” *Proc. ESS Conf. IEICE*, p. 126, 2004.

[5] T. Nosaka, M. Nakamura, and T. Kaneko, “A study on linear cryptanalysis of VSC128,” *Proc. ESS Conf. IEICE*, p. 127, 2004.

[6] Y. Tunoo, T. Saito, K. Myao, T. Suzaki, and T. Kawabata, “Distinguishing attack with chosen initial vector against VSC128,” SASC Workshop, pp. 212–219, 2004.

[7] A. Iwasaki and K. Umeno, “Improving security of Vector Stream Cipher,” *Nonlinear Theory and Its Applications, IEICE*, vol. 7, pp. 30–37, 2016.

[8] D.E. Knuth, "The Art of Computer Programming," vol. 2, Addison-Wesley, Upper Saddle River, 1981.

[9] R. Coveyou, Random Number Generation Is Too Important to Be Left to Chance, Studies in Applied Mathematics, III, pp. 70–111, 1970.

[10] R.L. Rivest, M.J.B. Robshaw, R. Sidney, and Y.L. Yin, "The RC6 block cipher," https://people.csail.mit.edu/rivest/pubs/RRSY98.pdf.

[11] A. Iwasaki and K. Umeno, "One-stroke polynomials over a ring of modulo 2^w," *JSIAM Letters*, vol. 9, pp. 5–8, 2017.

[12] A. Iwasaki and K. Umeno, "Three Theorems on odd degree Chebyshev polynomials and more generalized permutation polynomials over a ring of module 2^w," arXiv:1602.08238v2, 2016.

[13] NIST, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," Special Publication 800–22 Revision 1a, http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf, 2010.