

名古屋工業大学学術機関リポジトリ
Nagoya Institute of Technology Repository
<http://repo.lib.nitech.ac.jp>

Title	Internal-State Reconstruction of a Stream Cipher RC4
Author(s)	Shiraishi, Yoshiaki; Ohigashi, Toshihiro; Morii, Masakatsu
Citation	IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, E86-A(10): 2636-2638
Issue Date	2003-10-01
URL	http://repo.lib.nitech.ac.jp/handle/123456789/6759
Rights	Copyright(c)2003 IEICE http://search.ieice.org/index.html
Type	Journal Article
Textversion	publisher

- ・名古屋工業大学学術機関リポジトリは、名古屋工業大学内で生産された学術情報を電子的に収集・保存・発信するシステムです。
- ・論文の著作権は、著者または出版社が保持しています。著作権法で定める権利制限規定を超える利用については、著作権者に許諾を得てください。
- ・Textversionに「Author」と記載された論文は、著者原稿となります。実際の出版社版とは、レイアウト、字句校正レベルの異同がある場合もあります。
- ・Nagoya Institute of Technology Repository Sytem is built to collect, archive and offer electronically the academic information produced by Nagoya Institute of Technology.
- ・The copyright and related rights of the article are held by authors or publishers. The copyright owners' consents must be required to use it over the curtailment of copyrights.
- ・Textversion "Author " means the article is author's version. Author version may have some difference in layouts and wordings form publisher version.

Internal-State Reconstruction of a Stream Cipher RC4

Yoshiaki SHIRAISHI[†], *Regular Member*, Toshihiro OHIGASHI^{††}, *Student Member*,
and Masakatu MORII^{††}, *Regular Member*

SUMMARY Knudsen et al. proposed an efficient method based on a tree-search algorithm with recursive process for reconstructing the internal state of RC4 stream cipher. However, the method becomes infeasible for word size $n > 5$ because its time complexity to reconstruct the internal state is too large. This letter proposes a more efficient method than theirs. Our method can reconstruct the internal state by using the pre-known internal-state entries, which are fewer than their method.

key words: stream cipher, RC4, internal-state reconstruction

1. Introduction

RC4 is one of key-stream generators in stream ciphers [9]. Many key-stream generators consist of a number of possibly clocked linear feedback shift registers (LFSRs) combined by a function [7], [8]. In contrast, RC4 takes a design approach that is quite different from that of LFSR-based stream ciphers. RC4 is widely used in commercial products and standards, e.g. Secure Sockets Layer standard (SSL) 3.0 [3].

One of the research results about RC4 is the secret key reconstruction attack [1]. The attack can recover a secret key by using roughly 5,000,000 packets in the old version of Wired Equivalent Privacy (WEP) encryption. Other research results about RC4 are as follows: distinguishing attack [2], [4], [6], internal-state reconstruction [2], [5], output-prediction attack [2], etc.

Knudsen et al. have proposed an internal-state reconstruction method [5]. The method is based on a tree-search algorithm with recursive process, which causes increased computational complexity. To decrease computational complexity, we must have a part of internal-state entries to succeed in reconstruction, beforehand. However, the method becomes infeasible for word size $n > 5$. We must know at least 100 internal-state entries to succeed in the method with the word size $n = 8$. In this letter, we propose a method which is more efficient than Knudsen et al.'s method. We evaluate that efficiency by comparing the number

of necessary pre-known initial state entries to reconstruct an internal state completely. Our method can reconstruct an internal state by using less pre-known entries than the method. The result of our method is that an internal state with first 73 pre-known entries can be reconstructed within 2^{20} computational times.

2. Preliminaries

2.1 Description of RC4

We follow the description of RC4 as given in [9]. The internal state of RC4 at time t consists of a permutation table $S_t = (S_t[x])_{x=0}^{2^n-1}$ of 2^n n -bit words. Two n -bit word pointers i_t and j_t at time t are used and pointers i_0 and j_0 are initialized to zero. Let Z_t denote the n -bit word output of RC4 at time t . Then, the next-state and output functions of RC4 for every $t \geq 1$ are defined as

$$i_t = (i_{t-1} + 1) \bmod 2^n, \quad (1)$$

$$j_t = (j_{t-1} + S_{t-1}[i_t]) \bmod 2^n, \quad (2)$$

$$S_t[i_t] = S_{t-1}[j_t], \quad S_t[j_t] = S_{t-1}[i_t], \quad (3)$$

$$Z_t = S_t[(S_t[i_t] + S_t[j_t]) \bmod 2^n]. \quad (4)$$

Ciphertext C_t of length n is produced by

$$C_t = M_t \oplus Z_t,$$

where M_t is an n -bit piece of plaintext. The initial table S_0 is generated from a secret key. Details of this initialization are not important for our method.

2.2 Internal-State Reconstruction

The goal of an internal-state reconstruction attack is to reconstruct S_0 out of Z_t .

We show an internal-state reconstruction method proposed by Knudsen et al. [5] here. We will call this K-method. The algorithm uses recursive function calls with the time variable t . Let a_t denote the number of entries which had been assigned values in the internal-state table, and $S_t^* = (S_t^*[x])_{x=0}^{2^n-1}$ denote an array to check whether or not an x -th element is assigned. $S_t^*[x] \in \{*, 0, 1, \dots, 2^n - 1\}$; if $S_t^*[x] = *$, then the x -th element is unassigned, else $S_t^*[x] = S_t[x]$. We also define an array $E_t = (E_t[y])_{y=0}^{2^n-1}$ to check whether or not

Manuscript received January 17, 2003.

Manuscript revised April 18, 2003.

Final manuscript received June 6, 2003.

[†]The author is with the Department of Informatics, Kinki University, Higashi-Osaka-shi, 577-8502 Japan.

^{††}The authors are with the Department of Information Science and Intelligent Systems, The University of Tokushima, Tokushima-shi, 770-8506 Japan.

value y is assigned. If value y is assigned, then we denote $E_t[y] = \phi$, else $E_t[y] = \psi$. The following equation obtained from Eqs. (3) and (4) is used in the algorithm:

$$S_t^{-1}[Z_t] = (S_{t-1}[i_t] + S_{t-1}[j_t]) \bmod 2^n. \quad (5)$$

By Eq. (5), if two of three variables are assigned, the remaining variable can be computed. From the definition of RC4, all values in internal-state table S_t differ from each other; that is, if $x \neq x'$, $S_t[x] \neq S_t[x']$. If the value computed by Eq. (5) has already been assigned to other positions of S_t , then we can check a contradiction of the computed value.

[Algorithm of K-method] [5]

StepK1 Check $S_{t-1}^*[i_t] = *$:

- (a) if $S_{t-1}^*[i_t] \neq *$, proceed to StepK2.
- (b) otherwise assign, one after another, the $2^n - a_t$ unassigned elements to $S_{t-1}[i_t]$, increment a_t , $E_t[S_{t-1}[i_t]] := \phi$; then go to StepK2.

StepK2 Check $E_t[Z_t] = \phi$:

- (a) if $E_t[Z_t] = \phi$, calculate the expected value of $S_{t-1}[j_t]$ from Eq. (5). If this does not lead to a contradiction, then $E_t[S_{t-1}[j_t]] := \phi$; proceed to time $t + 1$ and go to StepK1.
- (b) otherwise, go to StepK3.

StepK3 Check $S_{t-1}^*[j_t] = *$:

- (a) if $S_{t-1}^*[j_t] \neq *$, calculate the expected value of $S_t^{-1}[Z_t]$ from Eq. (5). If this does not lead to a contradiction, then $E_t[Z_t] := \phi$; proceed to time $t + 1$ and go to StepK1.
- (b) otherwise, then assign, one after another, the $2^n - a_t$ unassigned elements to $S_{t-1}[j_t]$ and update a_t . Subsequently, check whether the given values of i_t, j_t and Z_t lead to a contradiction. If they do not, then $E_t[S_{t-1}[j_t]] := \phi$, $E_t[Z_t] := \phi$; proceed to time $t + 1$ and go to StepK1.

The method includes two important processes. One is the computation of unknown variables using Eq. (5) and then the check of contradiction; the other is recursive search using candidates of all unassigned values.

3. Proposed Method

We propose a new efficient internal-state reconstruction method improving the K-method. This section presents our algorithm and its experimental results.

3.1 Algorithm

Note that computational complexity of the K-method is affected by the computation of unknown variables, the check for contradiction in StepK2-(a) and StepK3-(a), and the recursive search in StepK1-(b) and StepK3-(b).

Obviously, the latter greatly influences the complexity in comparison with the former. We infer that the complexity of the method using either of the two recursive processes is less than using both processes such as in the K-method. In addition, it is also expected that we can compute some other unknown internal-state entries by reusing values obtained up to the time of computation. In view of the discussion, we propose the following algorithm. The points changed from K-method are adding new process called *backtracking for candidate reduction* to StepK1-(b) and the deletion of recursive search in StepK3-(b).

[Algorithm of the proposed method]

StepP1 Check $S_{t-1}^*[i_t] = *$:

- (a) if $S_{t-1}^*[i_t] \neq *$, proceed to StepP2.
- (b) otherwise execute *backtracking for candidate reduction*. Subsequently, assign, one after another, the $2^n - a_t$ unassigned elements to $S_{t-1}[i_t]$, increment a_t , $E_t[S_{t-1}[i_t]] := \phi$; then, go to StepP2.

StepP2 Check $E_t[Z_t] = \phi$:

- (a) if $E_t[Z_t] = \phi$, calculate the expected value of $S_{t-1}[j_t]$ from Eq. (5). If this does not lead to a contradiction, then $E_t[S_{t-1}[j_t]] := \phi$; proceed to time $t + 1$ and go to StepP1.
- (b) otherwise, go to StepP3.

StepP3 Check $S_{t-1}^*[j_t] = *$:

- (a) if $S_{t-1}^*[j_t] \neq *$, calculate the expected value of $S_t^{-1}[Z_t]$ from Eq. (5). If this does not lead to a contradiction, then $E_t[Z_t] := \phi$; proceed to time $t + 1$ and go to StepP1.
- (b) otherwise, proceed to time $t + 1$ and go to StepP1.

The concrete process of backtracking for candidate reduction is as follows:

[backtracking for candidate reduction]

StepB1 Check time variable t' such that correct values have already been assigned to all three variables in Eq. (5) for each time $t = 1, 2, \dots, t' - 1$; then go to StepB2.

StepB2 Check $E_t[Z_{t'}] = \phi$:

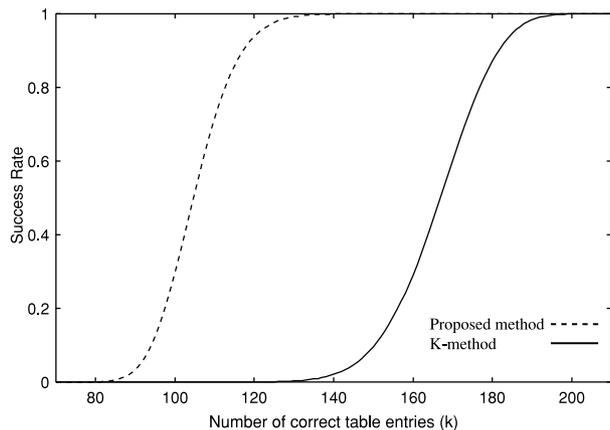
- (a) if $E_t[Z_{t'}] = \phi$, calculate the expected value of $S_{t'-1}[j_{t'}]$ from Eq. (5). If this does not lead to a contradiction, then update a_t , $E_t[S_{t'-1}[j_{t'}]] := \phi$; proceed to time $t' + 1$ and go to StepB2.
- (b) otherwise, go to StepB3.

StepB3 Check $S_{t'-1}^*[j_{t'}] = *$:

- (a) if $S_{t'-1}^*[j_{t'}] \neq *$, calculate the expected value of $S_{t'}^{-1}[Z_{t'}]$ from Eq. (5). If this does not lead to a contradiction, increment a_t , $E_t[Z_{t'}] := \phi$; proceed to time $t' + 1$ and go to StepB2.

Table 1 Simulation parameters.

word size	$n = 8$
target of attack	100,000 initial states
unit of calculation	updating internal states
success condition	complete case reconstructed initial state within 2^{20} times calculation
known entries of initial state	first k words
known plaintexts	2^{10} words

**Fig. 1** Success rates of the proposed method and the K-method.

- (b) otherwise, proceed to time $t' + 1$ and go to StepB2.

When $t' = t - 1$, the process is stopped. It is clear that candidates of recursive search just after backtracking can be reduced because there are some cases in which values are assigned to unknown variables in StepP1 at time t in StepB2 or StepB3. This is a valid process in terms of reducing complexity. Moreover, although the recursive search in StepP3-(b) is deleted, we can still expect benefits from effects of assigned values in the backtracking instead of from the recursive search as a whole.

3.2 Experimental Results

We conduct experiments of both the proposed method and the K-method using the parameter in Table 1, and obtain success rates for 100,000 initial states in the case where time complexity is within 2^{20} times updating of the internal state. We follow the unit of time complexity as used in [5].

Figure 1 shows the success rate result. As the figure indicates, our method can reconstruct the internal state by using the pre-known k entries of initial state, which is less than for the K-method. That is the reason

why time complexity is reduced by adding backtracking and deleting the recursive search procedure.

We make another experiment to demonstrate that we can reduce the value k such that at least one of 100,000 initial states can be reconstructed by our method. The result is that we reconstruct some internal states in $k = 73$ within one second using a Pentium4 1.7 GHz CPU with 256 MB RAM.

4. Conclusion

In this letter, we described an algorithm reconstructing the internal state of RC4. The time complexity of the algorithm is reduced by introducing a process of backtracking and deleting one of two recursive searches from K-method. Consequently, with lesser known entries in initial state than K-method, our method still succeeds in reconstructing the internal state. Moreover, we found some internal states which can be reconstructed easily when the number of known entries is equal to 73 in $n = 8$. The theoretical analysis of complexity remains as a further study.

Acknowledgment

The authors are grateful for helpful comments provided by an anonymous reviewer.

References

- [1] S. Fluhrer, I. Mantin, and A. Shamir, "Weaknesses in the key scheduling algorithm of RC4," Proc. SAC2001, Lecture Notes in Computer Science, vol.2259, pp.1-24, Springer-Verlag, 2001.
- [2] S. Fluhrer and D. McGrew, "Statistical analysis of the alleged RC4 keystream generator," Proc. FSE2000, Lecture Notes in Computer Science, vol.1978, pp.19-30, 2001.
- [3] A. Freier, P. Karlton, and P. Kocher, "The SSL 3.0 protocol," Netscape Communications, Nov. 1996.
- [4] J. Dj. Golić, "Linear statistical weakness of alleged RC4 keystream generator," Proc. EUROCRYPT'97, Lecture Notes in Computer Science, vol.1233, pp.226-238, Springer-Verlag, 1997.
- [5] L.R. Knudsen, W. Meier, B. Preneel, V. Rijmen, and S. Verdoolaege, "Analysis methods for (alleged) RC4," Proc. ASIACRYPT'98, Lecture Notes in Computer Science, vol.1514, pp.327-341, Springer-Verlag, 1998.
- [6] I. Mironov, "(Not so) random shuffles of RC4," Proc. CRYPTO2002, Lecture Notes in Computer Science, vol.2442, pp.304-319, Springer-Verlag, 2002.
- [7] R.A. Rueppel, Analysis and Design of Stream Ciphers, Springer-Verlag, Berlin, 1986.
- [8] R.A. Rueppel, "Stream ciphers," in Contemporary Cryptology, ed. G.J. Simmons, pp.65-134, IEEE Press, New York, 1992.
- [9] B. Schneier, Applied Cryptography, Wiley, New York, 1996.