Full Plaintext Recovery Attack on Broadcast RC4

Takanori Isobe[1], Toshihiro Ohigashi[2], Yuhei Watanabe[1], and Masakatu Morii[1]

[1] Kobe University
1-1 Rokkoudai, Nada-ku, Kobe 657-8501, Japan
yuheiwatanabe@stu.kobe-u.ac.jp
mmorii@kobe-u.ac.jp
[2] Hiroshima University
1-4-2 Kagamiyama, Higashi-Hiroshima, Hiroshima 739-8511, Japan
ohigashi@hiroshima-u.ac.jp

Abstract. This paper investigates the practical security of RC4 in broadcast setting where the same plaintext is encrypted with different user keys. We introduce several new biases in the initial (1st to 257th) bytes of the RC4 keystream, which are substantially stronger than known biases. Combining the new biases with the known ones, a cumulative list of strong biases in the first 257 bytes of the RC4 keystream is constructed. We demonstrate a plaintext recovery attack using our strong bias set of initial bytes by the means of a computer experiment. We show that almost all of the first 257 bytes of the plaintext can be recovered, with probability more than 0.8, using only 2^{32} ciphertexts encrypted by randomly-chosen keys. We also propose an efficient method to extract later bytes of the plaintext, after the 258th byte. The proposed method exploits our bias set of first 257 bytes in conjunction with the digraph repetition bias proposed by Mantin in EUROCRYPT 2005, and sequentially recovers the later bytes of the plaintext after recovering the first 257 bytes. Once the possible candidates for the first 257 bytes are obtained by our bias set, the later bytes can be recovered from about 2^{34} ciphertexts with probability close to 1. Our sequential method can recover any plaintext byte as the digraph repetition is a long-term bias present in every keystream byte of RC4. It is expected that our method can recover the first 2^{50} bytes $\approx$ 1000 T bytes of the plaintext, with probability close to 1, from only 2^{34} ciphertexts.

Key words: RC4, broadcast setting, plaintext recovery attack, bias, experimentally-verified attack

1 Introduction

RC4, designed by Rivest in 1987, is one of most widely used stream ciphers in the world. It is adopted in many software applications and standard protocols such as SSL/TLS, WEP, Microsoft Lotus, Oracle secure SQL and more. RC4 consists of a key scheduling algorithm (KSA) and a pseudo-random generation algorithm (PRGA). The KSA converts a user-provided variable-length key (typically, 5–32 bytes) into an initial state S consisting of a permutation of $\{0, 1, 2, \ldots, N-1\}$, where N is typically 256. The PRGA generates a keystream Z_1, Z_2, ..., Z_r, ... from S, where r is a round number of the PRGA. Z_r is XOR-ed with the r-th plaintext byte P_r to obtain the ciphertext byte C_r. The algorithm of RC4 is shown in Algorithm 1, where + denotes arithmetic addition modulo N, and i and j are used to point to the locations of S, respectively. Then, $S[x]$ denotes the value of S indexed x.

After the disclosure of its algorithm in 1994, RC4 has attracted intensive cryptanalytic efforts over past 20 years. Distinguishing attacks, which attempt to distinguish an RC4 keystream from a random stream, were proposed in [4, 3, 10, 11, 14, 16, 8]. State recovery attack, which recovers a full state instead of the user-provided key, was shown by Knudsen et al. [7], and it was improved by Maximov and Khovratovich [13]. Other types of attacks are also proposed, e.g., key collision attack [12], keystream predictive attack [10] and key recovery attacks from a state [15, 1].

In FSE2001, Mantin and Shamir presented an attack on RC4 in the broadcast setting where the same plaintext is encrypted with different user keys [11]. The Mantin-Shamir attack can extract the second byte of the plaintext from only $\Omega(N)$ ciphertexts encrypted with randomly-chosen different keys by exploiting a bias of Z_2. Specifically, the event $Z_2 = 0$ occurs with twice the expected probability of a random one. In FSE2011, Maitra, Paul and Sen Gupta showed that $Z_3, Z_4, \ldots, Z_{255}$ are also biased to 0 [8]. Then the bytes 3 to 255 can also be recovered in the broadcast setting, from $\Omega(N^3)$ ciphertexts.

Although the broadcast attacks were theoretically estimated, we find that three questions are still open in terms of a practical security of broadcast RC4.

Algorithm 1 RC4 Algorithm

KSA($K[0 \dots l-1]$):	**PRGA**(K):
for $i = 0$ to $N-1$ **do**	$i \leftarrow 0$
$S[i] \leftarrow i$	$j \leftarrow 0$
end for	$S \leftarrow KSA(K)$
$j \leftarrow 0$	**loop**
for $i = 0$ to $N-1$ **do**	$i \leftarrow i + 1$
$j \leftarrow j + S[i] + K[i \bmod l]$	$j \leftarrow j + S[i]$
Swap $S[i]$ and $S[j]$	Swap $S[i]$ and $S[j]$
end for	Output $Z \leftarrow S[S[i] + S[j]]$
	end loop

1. *Are the biases exploited in the previous attacks the strongest biases for the initial bytes 1 to 255?*
2. *While the previous results [11, 8] estimate only lower bounds (Ω), how many ciphertexts encrypted with different keys are actually required for a practical attack on broadcast RC4?*
3. *Is it possible to efficiently recover the later bytes of the plaintext, after byte 256?*

1.1 Our Contribution

In this paper, we provide answers to all the aforesaid questions. To begin with, we introduce a new bias regarding Z_1, which is a conditional bias such that Z_1 is biased to 0 when Z_2 is 0. Using this bias in conjunction with the bias of $Z_2 = 0$ [11], the first byte of a plaintext is extracted from $\Omega(N^2)$ ciphertexts encrypted with different keys. Although the strong bias of the first byte, which is a negative bias towards zero, has already been pointed out in [14, 6], it requires $\Omega(N^3)$ ciphertexts to extract the first byte of the plaintext. Thus, the new conditional bias observed by us is very useful, because the number of required ciphertexts to recover the first byte reduces by a factor of $N/2$ compared the straightforward method. Besides, we introduce new strong biases, i.e., $Z_3 = 131$, $Z_r = r$ for $3 \leq r \leq 255$, and extended keylength-dependent biases such that $Z_{x \cdot l} = -x \cdot l$ for $x = 2, 3, \dots, 7$ and $l = 16$, which are an extension of the keylength-dependent biases in which only the parameter of $x = 1$ is considered [5]. These new biases are substantially stronger than known biases of $Z_r = 0$ in case of certain bytes within $Z_3, Z_4, \dots, Z_{255}$. Thus, we need to utilize not only the known bias of $Z_r = 0$ but also these new biases for an efficient and successful plaintext recovery attack. After providing theoretical considerations for these biases, we experimentally confirm the validity of the same. Combining the new biases with known biases, we construct a cumulative list of strongest known biases in $Z_1, Z_2, \dots, Z_{255}$. At the same time, we experimentally show two new biases of Z_{256} and Z_{257}, and add these to our bias set. We also experimentally confirm that biases of $Z_2, Z_3, \dots, Z_{257}$ included in our bias set are *strongest* biases amongst all *unconditional* positive and negative biases of each byte when a 16-byte (128-bit) key is used.

We demonstrate a plaintext recovery attack using our bias set by the computer experiment, and estimate the number of required ciphertexts and success probability when $N = 256$. As a result, almost all first 257 bytes, $P_1, P_2, \dots, P_{257}$, can be extracted with probability more than 0.8 from 2^{32} ciphertexts encrypted by randomly-chosen keys. Given 2^{34} ciphertexts, all bytes of $P_1, P_2, \dots, P_{257}$ can be narrowed down to two candidates each with probability one. This is a first practical security evaluation of broadcast RC4 using all known biases of the cipher, and some new ones that we observe.

Finally, an efficient method to extract later bytes of the plaintext, namely bytes after P_{258}, is given. It exploits our bias set of $Z_1, Z_2, \dots, Z_{257}$ in conjunction with the digraph repetition bias proposed by Mantin [10], and then sequentially recovers bytes of the plaintext. Once the possible candidates for $P_1, P_2, \dots, P_{257}$ are obtained by our bias set, P_r ($r \geq 258$) are recovered from about 2^{34} ciphertexts with probability one. Since the digraph repetition bias is a long-term bias, which occurs in any keystream byte, our sequential method is expected to recover any plaintext byte from only ciphertexts produced by different randomly-chosen keys. We show that the first 2^{50} bytes $\approx$ 1000 T bytes of the plaintext can be recovered from 2^{34} ciphertexts with probability of 0.97170. Therefore, it can be said that our attack is a full plaintext recovery attack on broadcast RC4.

2 Known Attacks on Broadcast RC4

This section briefly reviews known attacks on RC4 in the broadcast setting where the same plaintext is encrypted with different randomly-chosen keys.

2.1 Mantin-Shamir (MS) Attack

Mantin and Shamir first presented a broadcast RC4 attack exploiting a bias of Z_2 [11].

Theorem 1 [11] *Assume that the initial permutation S is randomly chosen from the set of all the possible permutations of $\{0, 1, 2, \ldots, N-1\}$. Then the probability that the second output byte of RC4 is 0 is approximately $\frac{2}{N}$.*

This probability is estimated as $\frac{2}{256}$ when $N = 256$. Based on this bias, the broadcast RC4 attack is demonstrated by Theorems 2 and 3.

Theorem 2 [11] *Let X and Y be two distributions, and suppose that the event e happens in X with probability p and in Y with probability $p \cdot (1+q)$. Then for small p and q, $\mathrm{O}(\frac{1}{p \cdot q^2})$ samples suffice to distinguish X from Y with a constant probability of success.*

In this case, p and q are given as $p = 1/N$ and $q = 1$. The number of samples is about $1/N$.

Theorem 3 [11] *Let P be a plaintext, and let $C^{(1)}, C^{(2)}, \ldots, C^{(k)}$ be the RC4 encryptions of P under k uniformly distributed keys. Then, if $k = \Omega(N)$, the second byte of P can be reliably extracted from $C^{(1)}, C^{(2)}, \ldots, C^{(k)}$.*

According to the relation $C_2^{(i)} = P_2^{(i)} \oplus Z_2^{(i)}$, if $Z_2^{(i)} = 0$ holds, then $C_2^{(i)}$ is same as $P_2^{(i)}$. From Theorem 1, $Z_2 = 0$ occurs with twice the expected probability of a random one. Thus, most frequent byte in amongst $C_2^{(1)}, C_2^{(2)}, \ldots, C_2^{(k)}$ is likely to be P_2 itself. When $N = 256$, it requires more than 2^8 ciphertexts encrypted with randomly-chosen keys.

2.2 Maitra, Paul and Sen Gupta (MPS) Attack

Maitra, Paul and Sen Gupta showed that $Z_3, Z_4, \ldots, Z_{255}$ are also biased to 0 [8, 6]. Although the MS attack assumes that an initial permutation S is random, the MPS attack exploits biases of S after the KSA [9]. Let $S_r[x]$ be the value of S indexed x after r round, where S_0 is the initial state of RC4 after the KSA. Biases of the initial state of the PRGA are given as follow.

Proposition 1 [9] *After the end of KSA, for $0 \le u \le N-1, 0 \le v \le N-1$,*

$$\Pr(S_0[u] = v) = \begin{cases} \frac{1}{N} \cdot \left((\frac{N-1}{N})^v + (1 - (\frac{N-1}{N})^v) \cdot (\frac{N-1}{N})^{N-u-1} \right) & (v \le u), \\ \frac{1}{N} \cdot \left((\frac{N-1}{N})^{N-u-1} + (\frac{N-1}{N})^v \right) & (v > u). \end{cases}$$

The probability of $S_{r-1}[r]$ in the PRGA are given as the follows.

Theorem 4 [6] [3] *For $3 \le r \le N-1$, the probability $\Pr(S_{r-1}[r] = v)$ is approximately*

$$\Pr(S_1[r] = v) \cdot \left(1 - \frac{1}{N}\right)^{r-2} + \sum_{t=2}^{r-1} \sum_{w=0}^{r-t} \frac{\Pr(S_1[t] = v)}{w! \cdot N} \cdot \left(\frac{r-t-1}{N}\right)^w \cdot \left(1 - \frac{1}{N}\right)^{r-3-w},$$

where $\Pr(S_1[t] = v)$ is given as

$$\Pr(S_1[t] = v) = \begin{cases} \Pr(S_0[1] = 1) + \sum_{X \neq 1} \Pr(S_0[1] = X \wedge S_0[X] = 1) & (t = 1, v = 1), \\ \sum_{X \neq 1, v} \Pr(S_0[1] = X \wedge S_0[X] = v) & (t = 1, v \neq 1), \\ \Pr(S_0[1] = t) + \sum_{X \neq t} \Pr(S_0[1] = X \wedge S_0[t] = t) & (t \neq 1, v = t), \\ \sum_{X \neq t, v} \Pr(S_0[1] = X \wedge S_0[t] = v) & (t \neq 1, v \neq t). \end{cases}$$

[3] The theorems with respect to $Z_r = 0$ in [8] and [6] are slightly different. This paper uses the results from the full version [6].

Then, the bias of $\Pr(Z_r = 0)$ is estimated as follows.

Theorem 5 **[6]** *For* $3 \leq r \leq N-1$, $\Pr(Z_r = 0)$ *is approximately*

$$\Pr(Z_r = 0) \approx \frac{1}{N} + \frac{c_r}{N^2},$$

where c_r *is given as*

$$c_r = \begin{cases} \frac{N}{N-1} \cdot (N \cdot \Pr(S_{r-1}[r] = r) - 1) - \frac{N-2}{N-1} & (r = 3), \\ \frac{N}{N-1} \cdot (N \cdot \Pr(S_{r-1}[r] = r) - 1) & (r \neq 3). \end{cases}$$

Since the parameters of p and q are given as $p = 1/N$ and $q = c_r/N$, The number of required ciphertexts with different keys for the extraction of $P_3, P_4, \ldots, P_{255}$ is roughly estimated as $\Omega(N^3)$.

3 New Biases : Theory and Experiment

This section introduces four new biases in the keystream of RC4. To begin with, we prove a conditional bias of Z_1 towards 0 when $Z_2 = 0$. After that, we present new biases in the events, $Z_3 = 131$, $Z_r = r$, and extended keylength-dependent biases, which are substantially stronger than the known biases such as $Z_r = 0$. Then, we construct a cumulative list of strong biases in $Z_1, Z_2, \ldots, Z_{257}$ to mount an efficient plaintext recovery attack on broadcast RC4.

3.1 Bias of $Z_1 = 0 | Z_2 = 0$

A new conditional bias such that Z_1 is biased to 0 when $Z_2 = 0$ is given as Theorem 6.

Theorem 6 $\Pr(Z_1 = 0 | Z_2 = 0)$ *is approximately*

$$\Pr(Z_1 = 0 | Z_2 = 0) \approx \frac{1}{2} \cdot \Big(\Pr(S_0[1] = 1) + (1 - \Pr(S_0[1] = 1)) \cdot \frac{1}{N}\Big) + \frac{1}{2} \cdot \frac{1}{N}.$$

Proof. Two cases of $S_0[2] = 0$ and $S_0[2] \neq 0$ are considered. As mentioned in [11], when Z_2 is 0, $S_0[2]$ is also 0 with probability of $\frac{1}{2}$.

- $S_0[2] = 0$
  For $i = 1$, if $S_0[1]$ is 1, the index j is updated as $j = S_0[i] = S_0[1] = 1$. Then the first output byte Z_1 is expressed as follows (see Fig. 1),

  $$Z_1 = S_1[S_1[i] + S_1[j]] = S_1[S_1[1] + S_1[1]] = S_1[2] = S_0[2] = 0.$$

  Assuming that $Z_1 = 0$ holds with probability of $\frac{1}{N}$ when $S_0[1] \neq 1$, the probability of $\Pr(Z_1 = 0 | S_0[2] = 0)$ is estimated as

  $$\Pr(Z_1 = 0 | S_0[2] = 0) = \Pr(S_0[1] = 1) + (1 - \Pr(S_0[1] = 1)) \cdot \frac{1}{N}.$$

- $S_0[2] \neq 0$
  Suppose that the event of $Z_1 = 0$ occurs with probability of $\frac{1}{N}$. Then $\Pr(Z_1 = 0 | S_0[2] = 0)$ is estimated as

  $$\Pr(Z_1 = 0 | S_0[2] \neq 0) = \frac{1}{N}.$$

Therefore $\Pr(Z_1 = 0 | Z_2 = 0)$ is approximately

$$\begin{aligned} \Pr(Z_1 = 0 | Z_2 = 0) &= \Pr(Z_1 = 0 | S_0[2] = 0) \cdot \Pr(S_0[2] = 0 | Z_2 = 0) \\ &\quad + \Pr(Z_1 = 0 | S_0[2] \neq 0) \cdot \Pr(S_0[2] \neq 0 | Z_2 = 0) \\ &\approx \frac{1}{2} \cdot \Big(\Pr(S_0[1] = 1) + (1 - \Pr(S_0[1] = 1)) \cdot \frac{1}{N}\Big) + \frac{1}{2} \cdot \frac{1}{N}. \end{aligned}$$

□

When $N = 256$, $\Pr(S_0[1] = 1)$ is obtained by Proposition 1.

$$\Pr(S_0[1] = 1) = \frac{1}{256} \cdot \left(\left(\frac{1}{256}\right) + \left(1 - \left(\frac{1}{256}\right)\right) \cdot \left(\frac{1}{256}\right)^{254} \right) = 0.0038966.$$

Then, $\Pr(Z_1 = 0 | Z_2 = 0)$ is computed as

$$\Pr(Z_1 = 0|Z_2 = 0) = \frac{1}{2} \cdot \left(\Pr(S_0[1] = 1) + (1 - \Pr(S_0[1] = 1)) \cdot \frac{1}{256} \right) + \frac{1}{2} \cdot \frac{1}{256}$$
$$= 0.0058470 = 2^{-7.418} = 2^{-8} \cdot (1 + 2^{-1.009}).$$

Since the experimental value of $\Pr(Z_1 = 0|Z_2 = 0)$ for 2^{40} randomly-chosen keys is obtained as $0.0058109 = 2^{-8} \cdot (1 + 2^{-1.036})$, the theoretical value is correctly approximated.

From this bias, $\Pr(Z_1 = 0 \wedge Z_2 = 0)$ can also be estimated, as follows.

$$\Pr(Z_1 = 0 \wedge Z_2 = 0) = \Pr(Z_2 = 0) \cdot \Pr(Z_1 = 0|Z_2 = 0).$$

When $N = 256$, it is estimated as

$$\Pr(Z_1 = 0 \wedge Z_2 = 0) = \frac{2}{256} \cdot 2^{-7.418} = 2^{-14.418} = 2^{-16} \cdot (1 + 2^{0.996}).$$

This type of bias, called digraph bias, was proved as a long term bias by Fluhrer and McGrew [3]. However, such a strong bias in initial bytes was not reported. Specifically, the probability of the general long-term digraph bias is estimated as $2^{-16} \cdot (1 + 2^{-8})$ in [3] when $N = 256$, while that of our bias is $2^{-16} \cdot (1 + 2^{0.996})$. Thus our result reveals that the digraph bias in initial bytes is much stronger than what is estimated in [3].

Note that we searched for the similar form of conditional biases in first 256 bytes of the RC4 keystream. In particular, we check following specific patterns, $(Z_{r-a} = X|Z_r = Y)$ for $0 \le X, Y \le 255$, $2 \le r \le 256$, $1 \le a \le 8$. However, such a strong bias could not be found in our experiment, while all conditional biases are not covered.

Application to Broadcast RC4 attack: Using this new conditional bias of $Z_1 = 0|Z_2 = 0$ in conjunction with the bias of $Z_2 = 0$ [11], the first byte of the plaintext can be efficiently extracted, where $N = 256$. After 2^{17} ciphertexts with randomly-chosen keys are collected, following procedures are performed.

Step 1 Extract the second byte of the target plaintext, P_2, from 2^8 ciphertexts [11].
Step 2 Find the ciphertext in which $Z_2 = 0$ is XOR-ed by the computation of $C_2 \oplus P_2$. Then, $2^{10} = 2^{17} \cdot 2/256$ ciphertexts matching this criterion are expected to be obtained.
Step 3 Regard the most frequent byte in the first byte C_1 of these matching 2^{10} ciphertexts as P_1.

In Step 3, using the bias of $\Pr(Z_1 = 0|Z_2 = 0) = 2^{-8} \cdot (1 + 2^{-1.009})$, P_1 is extracted from remaining $2^{10} (\sim \frac{1}{2^{-8} \cdot (2^{-1.009})^2})$ ciphertexts by Theorems 2 and 3, assuming the relation of $C_1 = P_1 \oplus Z_1 = P_1$ holds. Although the bias of the first byte has already been pointed out in [14, 6], it requires 2^{24} ciphertexts to extract the first byte using the known biases, because the probability of the strongest bias, which is a negative bias of Z_1 towards 0, is estimated as about $2^{-8} \cdot (1 - 2^{-8})$ [6]. Thus, the new conditional bias identified by us is very efficient, because the number of required ciphertexts reduces by a factor close to $N/2$ compared to that of the straightforward method.

3.2 Bias of $Z_3 = 131$

A new bias of $Z_3 = 131$, which is stronger than $Z_3 = 0$ [8, 6], is given as Theorem 7.

Theorem 7 $\Pr(Z_3 = 131)$ *is approximately*

$$\Pr(Z_3 = 131) \approx \Pr(S_0[1] = 131) \cdot \Pr(S_0[2] = 128) +$$
$$(1 - \Pr(S_0[1] = 131) \cdot \Pr(S_0[2] = 128)) \cdot 1/N.$$

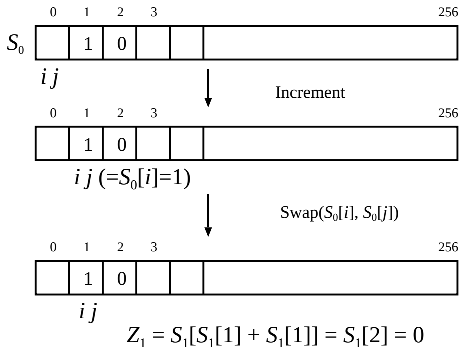

Fig. 1. Event for bias of $Z_1 = 0 | Z_2 = 0$

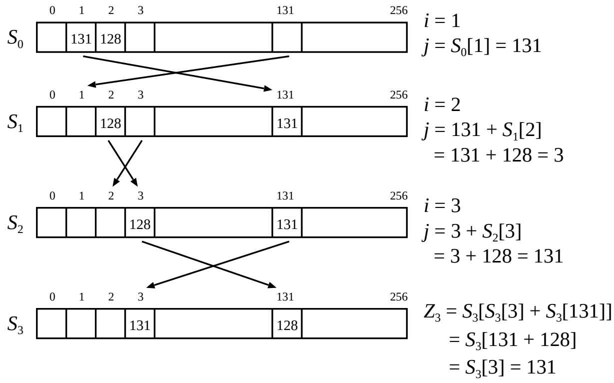

Fig. 2. Event for bias of $Z_3 = 131$

Proof. Suppose the events $S_0[1] = 131$ and $S_0[2] = 128$ occur after the KSA. For $i = 1$, j is updated as $S_0[1] = 131$. After $S_0[1]$ and $S_0[131]$ are swapped, $S_1[131]$ becomes 131. For $i = 2$, j is updated as $131 + S_1[2] = 131 + S_0[2] = 131 + 128 = 3$, and $S_1[2]$ and $S_1[3]$ are swapped. Then $S_2[3] = 128$ is obtained. Finally, for $i = 3$, j is updated as $3 + S_2[3] = 3 + 128 = 131$. After $S_2[3]$ and $S_2[131]$ are swapped, $S_3[3] = 131$ and $S_3[131] = 128$ holds. Then, a third output byte Z_3 is $Z_3 = S_3[S_3[3]+S_3[131]] = S_3[131 + 128] = S_3[3] = 131$. Thus, when $S_0[1] = 131$ and $S_0[2] = 128$ hold, $Z_3 = 131$ holds with probability one. Figure 2 depicts this event.

Assuming that in other cases, that is when $S_0[1] \neq 131$ or $S_0[2] \neq 128$, the event $Z_3 = 131$ holds with probability of $1/N$, the probability of $\Pr(Z_3 = 131)$ is estimated as

$$\Pr(Z_3 = 131) \approx \Pr(S_0[1] = 131) \cdot \Pr(S_0[2] = 128) + (1 - \Pr(S_0[1] = 131) \cdot \Pr(S_0[2] = 128)) \cdot 1/N.$$

□

When $N = 256$, by Proposition 1, $\Pr(S_0[1] = 131)$ and $\Pr(S_0[2] = 128)$ are estimated as

$$\Pr(S_0[1] = 131) = \frac{1}{256} \cdot \left(\left(\frac{255}{256}\right)^{256-1-1} + \left(\frac{255}{256}\right)^{131} \right) = 0.0037848,$$

$$\Pr(S_0[2] = 128) = \frac{1}{256} \cdot \left(\left(\frac{255}{256}\right)^{256-2-1} + \left(\frac{255}{256}\right)^{128} \right) = 0.0038181.$$

Thus, $\Pr(Z_r = 131)$ is computed as

$$\Pr(Z_3 = 131) \approx 0.0039206 = 2^{-8} \cdot (1 + 2^{-8.089}).$$

Since experimental value of this bias for 2^{40} randomly-chosen keys is obtained as $0.0039204 = 2^{-8} \cdot (1 + 2^{-8.109})$, the theoretical value is correctly approximated.

Let us compare it to the bias of $Z_3 = 0$ of the MPS attack [8, 6]. The experimental value for 2^{40} randomly-chosen keys is obtained as

$$\Pr(Z_3 = 0) = 0.0039116 = 2^{-8} \cdot (1 + 2^{-9.512}).$$

Thus, the bias of $Z_3 = 131$ is stronger than that of $Z_3 = 0$.

We should utilize $Z_3 = 131$ instead of $Z_3 = 0$ for the efficient plaintext recovery attack. When $Z_3 = 131$ and $Z_3 = 0$ are jointly used, two candidates of P_3 remain. Thus, in order to detect one correct value of P_3, the only use of $Z_3 = 131$ is more efficient.

3.3 Bias of $Z_r = r$ for $3 \leq r \leq N - 1$

We also present a new bias in the event $Z_r = r$ for $3 \leq r \leq N - 1$, whose probabilities are very close to those of $Z_r = 0$ [8], and the new biases are stronger than those of $Z_r = 0$ in some rounds. Thus, for an efficient attack, we need to carefully consider which biases are stronger in each round. The probability of $Z_r = r$ is given as Theorem 8.

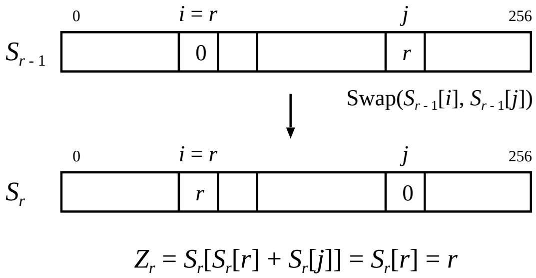

Fig. 3. Event (Case 1) for bias of $Z_r = r$

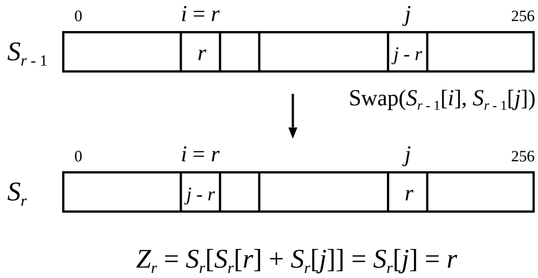

Fig. 4. Event (Case 2) for bias of $Z_r = r$

Theorem 8 $\Pr(Z_r = r)$ *for* $3 \leq r \leq N-1$ *is approximately*

$$\Pr(Z_r = r) \approx p_{r-1,0} \cdot \frac{1}{N} + p_{r-1,r} \cdot \frac{1}{N} \cdot \frac{N-2}{N} + (1 - p_{r-1,0} \cdot \frac{1}{N} - p_{r-1,r} \cdot \frac{1}{N} - (1 - p_{r-1,0}) \cdot \frac{1}{N} \cdot 2) \cdot \frac{1}{N},$$

where $p_{r-1,0} = \Pr(S_{r-1}[r] = 0)$ *and* $p_{r-1,r} = \Pr(S_{r-1}[r] = r)$.

Proof. Let i_r and j_r be r-th i and j, respectively. For $i_r = r$, an output Z_r is expressed as

$$Z_r = S_r[S_r[i_r] + S_r[j_r]] = S_r[S_r[r] + S_{r-1}[r]].$$

Then, let us consider four independent cases.

Case 1 : $S_{r-1}[r] = 0 \wedge S_r[r] = r$
Case 2 : $S_{r-1}[r] = r \wedge S_r[r] = j_r - r \wedge j_r \neq r, r+r$
Case 3 : $S_{r-1}[r] \neq 0 \wedge S_r[r] = r - S_{r-1}[r]$
Case 4 : $S_{r-1}[r] \neq 0 \wedge S_r[r] = r$

In Case 1 and Case 2, the output is always $Z_r = r$. On the other hand, in Case 3 and Case 4, the output is not $Z_r = r$.

Case 1 : $S_{r-1}[r] = 0 \wedge S_r[r] = r$
The output is expressed as $Z_r = S_r[S_r[r] + S_{r-1}[r]] = S_r[r+0] = S_r[r] = r$ (see Fig. 3). Then, the probability of $Z_r = r$ is one. Here $S_r[r]$ is chosen by pointer j. Since j_r for $r \geq 3$ behaves randomly [8], $S_r[r]$ is assumed to be uniformly random. it is estimated as

$$\Pr(S_{r-1}[r] = 0 \wedge S_r[r] = r) = p_{r-1,0} \cdot \frac{1}{N}.$$

Case 2 : $S_{r-1}[r] = r \wedge S_r[r] = j_r - r \wedge j_r \neq r, r+r$
The output is expressed as $Z_r = S_r[S_r[r] + S_{r-1}[r]] = S_r[j_r - r + r] = S_r[j_r] = S_{r-1}[r] = r$ (see Fig. 4). Then, the probability of $Z_r = r$ is one. Similar to Case 1, $S_r[r]$ is assumed to be uniformly random.

When $j_r = r$, the probability of $Z_r = r$ is zero because of the relation of $Z_r = S_r[S_r[r] + S_{r-1}[r]] = S_r[0+r] = S_r[r] = 0$. Also, when $j_r = r+r$, since $S_r[r] = r$ and $Z_r = S_r[S_r[r] + S_{r-1}[r]] = S_r[r+r] \neq r$, the probability of $Z_r = r$ is zero. Thus, the conditions of $j_r \neq r, r+r$ are necessary for $Z_r = r$. Then, it is estimated as

$$\Pr(S_{r-1}[r] = r \wedge S_r[r] = j_r - r \wedge j_r \neq r, r+r) = p_{r-1,r} \cdot \frac{1}{N} \cdot \frac{N-2}{N}.$$

Case 3 : $S_{r-1}[r] \neq 0 \wedge S_r[r] = r - S_{r-1}[r]$
The equation of $Z_r = S_r[r - S_{r-1}[r] + S_{r-1}[r]] = S_r[r]$ holds. Then, $S_r[r] = r - S_{r-1}[r]$ is not r, because $S_{r-1}[r]$ is not 0. Thus, it is estimated as

$$\Pr(S_{r-1}[r] \neq 0 \wedge S_r[r] = r - S_{r-1}[r]) = (1 - p_{r-1,0}) \cdot \frac{1}{N}.$$

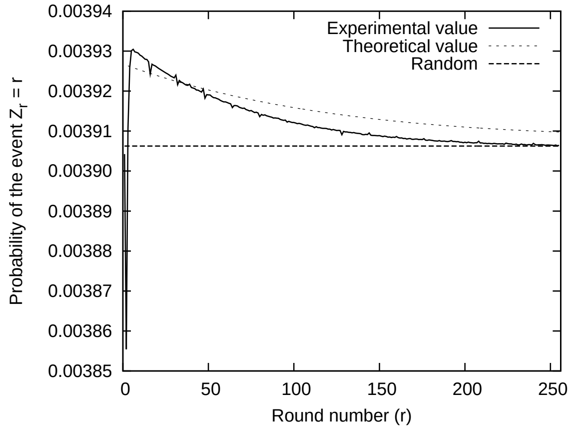

Fig. 5. Theoretical values and experimental values of $Z_r = r$

Case 4 : $S_{r-1}[r] \neq 0 \wedge S_r[r] = r$
The output is expressed as $Z_r = S_r[r+S_{r-1}[r]]$. According to the equation of $S_{r-1}[r] \neq 0$, The probability of $Z_r = r$ is zero. Thus, it is estimated as

$$\Pr(S_{r-1}[r] \neq (0,r) \wedge S_r[r] = r - S_{r-1}[r]) = (1 - p_{r-1,0}) \cdot \frac{1}{N}.$$

Assuming that in other cases, $Z_r = r$ holds with probability of $1/N$, the probability of $\Pr(Z_r = r)$ is estimated as

$$\Pr(Z_r = r) \approx p_{r-1,0} \cdot \frac{1}{N} + p_{r-1,r} \cdot \frac{1}{N} \cdot \frac{N-2}{N} + (1 - p_{r-1,0} \cdot \frac{1}{N} - p_{r-1,r} \cdot \frac{1}{N} - (1 - p_{r-1,0}) \cdot \frac{1}{N} \cdot 2) \cdot \frac{1}{N}.$$

□

Here, $p_{r-1,r}$ and $p_{r-1,0}$ are obtained from Theorem 4. Figure 5 shows the comparison of theoretical values and experimental values of $Z_r = r$ for 2^{40} randomly-chosen keys when $N = 256$. Since the theoretical values do not exactly coincide with the experimental values, we do not claim that Theorem 8 completely prove this bias. We guess that several minor events are not covered in our approach. However, the order of the bias seems to be well matched. At least it can be said that the main event causing this bias is discovered.

3.4 Extended Keylength-dependent Biases

We present new biases called extended keylength-dependent biases, which are an extension of keylength-dependent biases [17, 5], that is, the bias of $Z_l = -l$ when the key length is l bytes. For example, when using a 128-bit key (16 bytes), Z_{16} is biased to -16 (= 240). In addition to it, we experimentally found that when key length is l bytes, $Z_{x \cdot l}$ is also biased to $-x \cdot l$ ($x = 2, 3, 4, 5, 6, 7$), e.g., $Z_r = -r$ for $r =$ 32, 48, 64, 80, 96, 112, assuming $l = 16$. Importantly, the extended keylength-dependent biases are much stronger than the other known biases such as $Z_r = 0$ and $Z_r = r$. Thus, it must be taken into account for constructing the efficient broadcast RC4 attack. Table 1 shows experimental values of the extended keylength-dependent bias $Z_r = -r$, $Z_r = 0$, and $Z_r = r$ for 2^{40} randomly-chosen keys, when r is a multiple of the keylength, $l = 16$ in this case.

We tackle theoretical proofs of extended keylength-dependent biases. Figure 6 shows our experimental values and known theoretical values, which are calculated by Theorem 9 in [5], of these extended keylength-dependent biases. In [5], the theoretical proof only considers the relevant parameters for $Z_r = -r$ when $r = l$, and our biases of $Z_r = -r$ for $r =$ 32, 48, 64, 80, 96, 112 are out of scope. This

Table 1. Experimental values of $Z_r = -r$, $Z_r = 0$ and $Z_r = r$

r	$\Pr(Z_r = -r)$	$\Pr(Z_r = 0)$	$\Pr(Z_r = r)$
16	$2^{-8} \cdot (1 + 2^{-4.811})$	$2^{-8} \cdot (1 + 2^{-7.714})$	$2^{-8} \cdot (1 + 2^{-7.762})$
32	$2^{-8} \cdot (1 + 2^{-5.383})$	$2^{-8} \cdot (1 + 2^{-7.880})$	$2^{-8} \cdot (1 + 2^{-7.991})$
48	$2^{-8} \cdot (1 + 2^{-5.938})$	$2^{-8} \cdot (1 + 2^{-8.043})$	$2^{-8} \cdot (1 + 2^{-8.350})$
64	$2^{-8} \cdot (1 + 2^{-6.496})$	$2^{-8} \cdot (1 + 2^{-8.244})$	$2^{-8} \cdot (1 + 2^{-8.664})$
80	$2^{-8} \cdot (1 + 2^{-7.224})$	$2^{-8} \cdot (1 + 2^{-8.407})$	$2^{-8} \cdot (1 + 2^{-9.052})$
96	$2^{-8} \cdot (1 + 2^{-7.911})$	$2^{-8} \cdot (1 + 2^{-8.577})$	$2^{-8} \cdot (1 + 2^{-9.351})$
112	$2^{-8} \cdot (1 + 2^{-8.666})$	$2^{-8} \cdot (1 + 2^{-8.747})$	$2^{-8} \cdot (1 + 2^{-9.732})$

Table 2. Experimental values of E_1, E_2 and E_3 and conditional probability when $l = 16$

r	$\Pr(E_1)$	$\Pr(E_2)$	$\Pr(Z_r = -r \mid E_2)$	$\Pr(E_3)$	$\Pr(Z_r = -r \mid E_3)$
16	0.0000196	0.0038998	0.0266066	0.0030881	0.0185482
32	0.0000186	0.0078119	0.0057877	0.0036847	0.0242181
48	0.0000169	0.0039107	0.0039338	0.0038805	0.0195860
64	0.0000160	0.0156367	0.0039003	0.0038698	0.0146792
80	0.0000154	0.0039061	0.0039464	0.0038687	0.0104982
96	0.0000147	0.0078133	0.0039161	0.0038525	0.0081722
112	0.0000143	0.0039030	0.0039195	0.0038742	0.0066151

is the reason why there is a large gap between the theoretical estimates and the experimental values in these cases. Thus, we need to reconsider theoretical proofs of the extended keylength-dependent biases.

As a result of our thorough analysis of state transitions when $Z_r = -r$, three events are found that mainly affect the extended keylength-dependent biases.

Source Event 1 : $S_{r-1}[r] = 2r \land S_r[r] = -r$
Source Event 2 : $f_{r-1} = -r$
Source Event 3 : $f_{r-1} \neq -r \land S_{r-1}[r] = 0$

In all of the above events, f_y is given as

$$f_y = \sum_{x=0}^{y} K[x] + \frac{y \cdot (y+1)}{2}.$$

Hereafter, these events are termed as E_1, E_2, E_3, respectively. E_1 and E_3 are new events found by us, while E_2 is used in the proof of the original keylength-dependent bias [5]. Note that when the event E_1 occurs, output Z_r always becomes $-r$, because of the relation of $Z_r = S_r[S_r[r]+S_{r-1}[r]] = S_r[(-r)+2r] = S_r[r] = -r$. Experimental values of E_1, E_2 and E_3 when $l = 16$ are shown in Table 2.

The probability of $Z_r = -r$ $(r = 16, 32, 48, 64, 80, 96, 112)$ is estimated as

$$\begin{aligned}\Pr(Z_r = -r) \approx{} & \Pr(E_1) + \Pr(Z_r = -r \mid E_2) \cdot \Pr(E_2) \\ & +\Pr(Z_r = -r \mid E_3) \cdot \Pr(E_3) \\ & + (1 - \Pr(E_1) - \Pr(E_2) - \Pr(E_3)) \cdot \frac{1}{N}.\end{aligned}$$

assuming the probability of $Z_r = -r$ is $1/N$ in the cases except E_1, E_2 and E_3.

Figure 6 shows our semi-theoretical values and experimental values of extended keylength-dependent biases, i.e., $Z_r = -r$ $(r = 16, 32, 48, 64, 80, 96, 112)$, where semi-theoretical values are computed by assigning experimental values of Table 2 to the above equation. Since semi-theoretical value are partially based on experimental results, we can not claim that complete theoretical proof of these bias are given. However, it is confirmed that the three events E_1, E_2 and E_3 mainly affect the extended keylength-dependent biases. In Appendix B, the theoretical analysis of E_1 and E_2 are given while the proof of E_3 is still open, and this theoretical values are also included in Fig. 6.

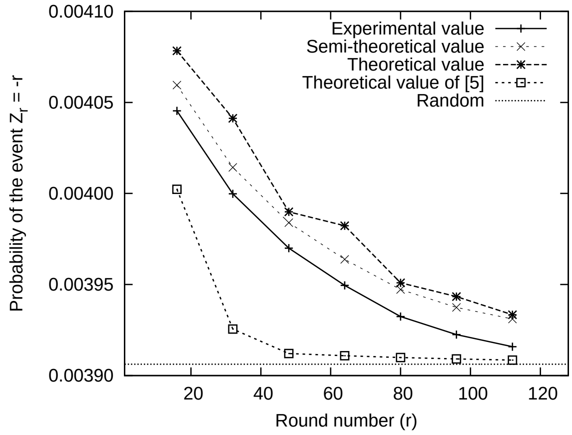

Fig. 6. Experimental values, semi-theoretical values, and theoretical values of $Z_r = -r$ when $l = 16$

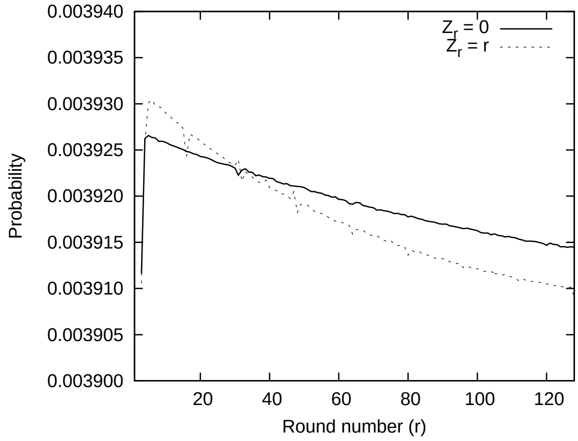

Fig. 7. Comparison between $Z_r = 0$ and $Z_r = r$ for $3 \leq r \leq 255$

3.5 Cumulative Bias Set of First 257 Bytes

When $N = 256$, a set of strong biases in $Z_1, Z_2, \ldots, Z_{255}$ is given in Table 3. Our new biases, namely the ones involving Z_1, Z_3, Z_{32}, Z_{48}, Z_{64}, Z_{80}, Z_{96}, Z_{112}, are included. Here, let us compare between the biases of $Z_r = 0$ [8, 6] and $Z_r = r$, whose probabilities are of the same order, and are very close in the range $3 \leq r \leq 255$. According to our experiments with 2^{40} randomly-chosen keys (see Fig. 7), $Z_r = r$ is stronger than $Z_r = 0$ in $Z_5, Z_6, \ldots, Z_{31}$. Thus we choose the bias $Z_r = r$ in $Z_5, Z_6, \ldots, Z_{31}$ and the bias $Z_r = 0$ in the other cases as the strongest bias except for the cases involving Z_3, Z_{16}, Z_{32}, Z_{48}, Z_{64}, Z_{80}, Z_{96}, Z_{112}. Besides, we experimentally found two new biases for the events $Z_{256} \neq 0$ and $Z_{257} = 0$, and added these to our bias set, while we could not provide the theoretical proofs. Note that it is experimentally confirmed that biases of $Z_2, Z_3, \ldots, Z_{257}$ included in our bias set are strongest known biases amongst all the positive and negative biases that have been discovered for these bytes.

For the first time, we propose a cumulative list of strongest known biases in the initial bytes of RC4 that can be exploited in a practical attack against the broadcast mode of the cipher.

[4] Values of extended keylength-dependent biases are semi-theoretical ones.

Table 3. Cumulative bias set of first 257 bytes

r	Strongest known bias of Z_r	Prob.(Theoretical)[4]	Prob.(Experimental)
1	$Z_1 = 0 \| Z_2 = 0$ (Our)	$2^{-8} \cdot (1 + 2^{-1.009})$	$2^{-8} \cdot (1 + 2^{-1.036})$
2	$Z_2 = 0$ [11]	$2^{-8} \cdot (1 + 2^{0})$	$2^{-8} \cdot (1 + 2^{0.002})$
3	$Z_3 = 131$ (Our)	$2^{-8} \cdot (1 + 2^{-8.089})$	$2^{-8} \cdot (1 + 2^{-8.109})$
4	$Z_4 = 0$ [8]	$2^{-8} \cdot (1 + 2^{-7.581})$	$2^{-8} \cdot (1 + 2^{-7.611})$
5–15	$Z_r = r$ (Our)	max: $2^{-8} \cdot (1 + 2^{-7.627})$ min: $2^{-8} \cdot (1 + 2^{-7.737})$	max: $2^{-8} \cdot (1 + 2^{-7.335})$ min: $2^{-8} \cdot (1 + 2^{-7.535})$
16	$Z_{16} = 240$ [5]	$2^{-8} \cdot (1 + 2^{-4.671})$	$2^{-8} \cdot (1 + 2^{-4.811})$
17–31	$Z_r = r$ (Our)	max: $2^{-8} \cdot (1 + 2^{-7.759})$ min: $2^{-8} \cdot (1 + 2^{-7.912})$	max: $2^{-8} \cdot (1 + 2^{-7.576})$ min: $2^{-8} \cdot (1 + 2^{-7.839})$
32	$Z_{32} = 224$ (Our)	$2^{-8} \cdot (1 + 2^{-5.176})$	$2^{-8} \cdot (1 + 2^{-5.383})$
33–47	$Z_r = 0$ [8]	max: $2^{-8} \cdot (1 + 2^{-7.897})$ min: $2^{-8} \cdot (1 + 2^{-8.050})$	max: $2^{-8} \cdot (1 + 2^{-7.868})$ min: $2^{-8} \cdot (1 + 2^{-8.039})$
48	$Z_{48} = 208$ (Our)	$2^{-8} \cdot (1 + 2^{-5.651})$	$2^{-8} \cdot (1 + 2^{-5.938})$
49–63	$Z_r = 0$ [8]	max: $2^{-8} \cdot (1 + 2^{-8.072})$ min: $2^{-8} \cdot (1 + 2^{-8.224})$	max: $2^{-8} \cdot (1 + 2^{-8.046})$ min: $2^{-8} \cdot (1 + 2^{-8.238})$
64	$Z_{64} = 192$ (Our)	$2^{-8} \cdot (1 + 2^{-6.085})$	$2^{-8} \cdot (1 + 2^{-6.496})$
65–79	$Z_r = 0$ [8]	max: $2^{-8} \cdot (1 + 2^{-8.246})$ min: $2^{-8} \cdot (1 + 2^{-8.398})$	max: $2^{-8} \cdot (1 + 2^{-8.223})$ min: $2^{-8} \cdot (1 + 2^{-8.376})$
80	$Z_{80} = 176$ (Our)	$2^{-8} \cdot (1 + 2^{-6.574})$	$2^{-8} \cdot (1 + 2^{-7.224})$
81–95	$Z_r = 0$ [8]	max: $2^{-8} \cdot (1 + 2^{-8.420})$ min: $2^{-8} \cdot (1 + 2^{-8.571})$	max: $2^{-8} \cdot (1 + 2^{-8.398})$ min: $2^{-8} \cdot (1 + 2^{-8.565})$
96	$Z_{96} = 160$ (Our)	$2^{-8} \cdot (1 + 2^{-6.970})$	$2^{-8} \cdot (1 + 2^{-7.911})$
97–111	$Z_r = 0$ [8]	max: $2^{-8} \cdot (1 + 2^{-8.592})$ min: $2^{-8} \cdot (1 + 2^{-8.741})$	max: $2^{-8} \cdot (1 + 2^{-8.570})$ min: $2^{-8} \cdot (1 + 2^{-8.722})$
112	$Z_{112} = 144$ (Our)	$2^{-8} \cdot (1 + 2^{-7.300})$	$2^{-8} \cdot (1 + 2^{-8.666})$
113–255	$Z_r = 0$ [8]	max: $2^{-8} \cdot (1 + 2^{-8.763})$ min: $2^{-8} \cdot (1 + 2^{-10.052})$	max: $2^{-8} \cdot (1 + 2^{-8.760})$ min: $2^{-8} \cdot (1 + 2^{-10.041})$
256	$Z_r = 0$(*negative bias*) (Our)	N/A	$2^{-8} \cdot (1 - 2^{-9.407})$
257	$Z_r = 0$ (Our)	N/A	$2^{-8} \cdot (1 + 2^{-9.531})$

4 Experimental Results of Plaintext Recovery Attack

We demonstrate a plaintext recovery attack using our cumulative bias set of first 257 bytes by a computer experiment, when $N = 256$, and estimate the number of required ciphertexts and the probability of success for our attack. The details of our experiment are as follows.

Step 1 Randomly generate a target plaintext P.
Step 2 Encrypt P with 2^x randomly-chosen keys, and obtain 2^x ciphertexts C.
Step 3 Find most frequent byte in each byte, and extract P_r, assuming $P_r = C_r \oplus Z_r$ where Z_r is the value of the keystream byte from our bias set.

In the case of P_1, the method mentioned in Section 3.1 is used for efficient extraction of P_1. Specifically, after P_2 is recovered, we extract P_1 by using the conditional bias such that $Z_1 = 0$ when $Z_2 = 0$.

We perform the above experiment for 256 different plaintexts in the cases where $2^6, 2^7, \ldots, 2^{35}$ ciphertexts with randomly-chosen keys are given. Figure 8 shows the probability of successfully recovering the values of P_1, P_2, P_3, P_5, and P_{16} for each amount of ciphertexts. Here, the success probability is estimated by the number of correctly-extracted plaintexts for each byte. For example, if the target byte of only 100 plaintexts out of 256 plaintexts can be correctly recovered, the probability is estimated as 0.39 (= 100/256). The second byte of plaintext P_2 can be extracted from 2^{12} ciphertexts with probability one. In previous attacks such as the MS attack [11] and the MPS attack [8], the number of required ciphertexts is theoretically estimated only in terms of the lower bound Ω. Our results first reveal the concrete number of ciphertexts, and the corresponding success probability.

Figure 9 shows that the success probability of extracting each byte P_r ($1 \leq r \leq 257$) when 2^{24}, 2^{28}, 2^{32}, 2^{35} ciphertexts are given. Note that the probability of a random guess is $1/256 = 0.00390625$. Given 2^{32} ciphertexts, all bytes of $P_1, P_2, \ldots, P_{257}$ can be extracted with probability more than 0.5. In addition,

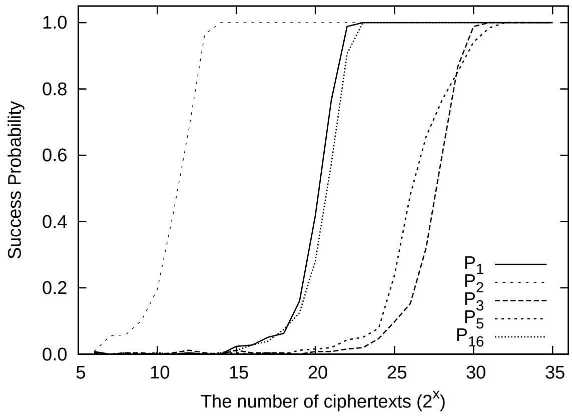

Fig. 8. Relation of the number of ciphertexts and success probability of recovering P_1, P_2, P_3, P_5, and P_{16}

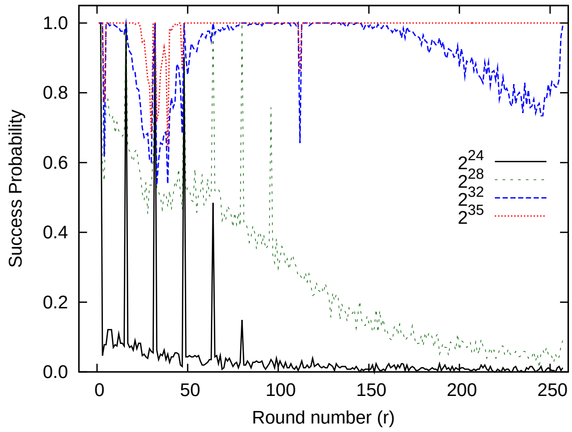

Fig. 9. Success probability of extracting P_r $(1 \leq r \leq 257)$ with different number of samples (one candidate)

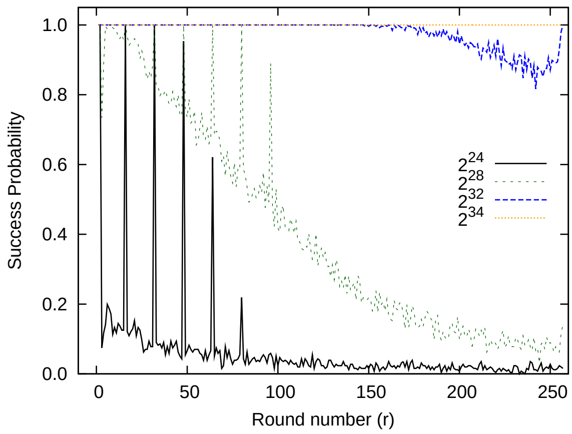

Fig. 10. Success probability of extracting P_r $(1 \leq r \leq 257)$ with different number of samples (two candidates)

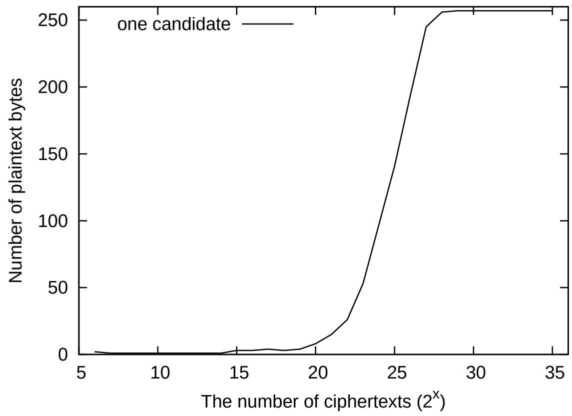

Fig. 11. The number of plaintext bytes that are extracted with five times higher than that of a random guess

most bytes can be extracted with probability more than 0.8. Also, the bytes having stronger bias such as P_1, P_2, P_{16}, P_{32}, P_{48}, P_{64}, are extracted from only 2^{24} ciphertexts with high probability. However, even if 2^{35} ciphertexts are given, the probability does not become one in some bytes. It is guessed that in such bytes, the difference of probability of the strongest known bias (as in our cumulative bias set) and the second one is very small. Thus, more ciphertexts are required for an attack with probability one.

We additionally utilize the second most frequent byte in the ciphertexts for extracting plaintext bytes. In other words, two candidates are obtained by using the relation of $P_r = C_r \oplus Z_r$, where C_r are most and second most frequent ciphertext bytes and Z_r is chosen from our bias set. This result is shown in Fig. 10, and its success probability is estimated as the probability that the guess for the correct plaintext byte is narrowed down to two possible candidates. Note that the probability of a random guess for such a scenario is $2/256 = 0.0078125$. Given 2^{34} ciphertexts, each byte of $P_1, P_2, \ldots, P_{257}$ can be extracted with probability one. In this case, although we can not obtain the correct byte of the plaintext, it is narrowed down to only two candidates. For the experiments of Fig. 9 and 10, it requires about one day if one uses a single CPU core (Intel(R) Core(TM) i7 CPU 920@ 2.67GHz) to obtain the result of one plaintext, where 256 plaintexts are used.

Figure 11 shows the number of plaintext bytes that are extracted with five times higher probability than that of a random guess, i.e., where the success probability is more than $\frac{5}{256}$. Given 2^{29} ciphertexts, all the plaintext bytes $P_1, P_2, \ldots, P_{257}$ are guessed with much higher probability than random guesses.

5 How to Recover Bytes of the Plaintext after P_{258}

In this section, we propose an efficient method to recover later bytes of the plaintext, namely bytes after P_{258}. The method using our bias in initial bytes is not directly applied to extract these bytes, because it exploits biases existing in only the initial keystream. For the extraction of the later bytes, a long-term bias, which occurs in any keystream bytes, is utilized. In particular, the digraph repetition bias (also called $ABSAB$ bias) proposed by Mantin [10], which is the strongest known long-term bias, is used. Combining it with our cumulative bias set of $Z_1, Z_2, \ldots, Z_{257}$, we can sequentially recover bytes of a plaintext, even after P_{258}, given only the ciphertexts.

5.1 Best Known Long-term Bias (*ABSAB* bias) [10]

$ABSAB$ bias is statistical biases of the digraph distribution in the RC4 keystream. Specifically, digraphs AB tend to repeat with short gaps S between them, e.g., $ABAB$, $ABCAB$ and $ABCDAB$, where gap S is defined as zero, C, and CD, respectively. The detail of $ABSAB$ bias is expressed as follows,

$$Z_r \mid\mid Z_{r+1} = Z_{r+2+G} \mid\mid Z_{r+3+G} \quad \text{for } G \geq 0, \tag{1}$$

where $||$ is a concatenation. The probability that Eq. (1) holds is given as Theorem 9.

Theorem 9 [10] *For small values of G the probability of the pattern ABSAB in RC4 keystream, where S is a G-byte string, is* $(1 + e^{(-4-8G)/N}/N) \cdot 1/N^2$.

For the enhancement of these biases, combining use of $ABSAB$ biases with different G is considered by using the following lemma for the discrimination.

Lemma 1 [10] *Let X and Y be two distributions and suppose that the independent events $\{E_i \colon 1 \leq i \leq k\}$ occur with probabilities $p_X(E_i) = p_i$ in X and $p_Y(E_i) = (1 + b_i) \cdot p_i$ in Y. Then the discrimination D of the distributions is $\sum_i p_i \cdot b_i^2$.*

The number of required samples for distinguishing the biased distribution from the random distribution with probability of $1 - \alpha$ is given as the following lemma.

Lemma 2 [10] *The number of samples that is required for distinguishing two distributions that have discrimination D with success rate $1 - \alpha$ (for both directions) is* $(1/D) \cdot (1 - 2\alpha) \cdot log_2 \frac{1-\alpha}{\alpha}$.

This lemma shows that in the broadcast RC4 attack, given D and the number of samples $N_{ciphertext}$, the success probability for distinguishing the distribution of correct candidate plaintext byte (the biased distribution) from the distribution of one wrong candidate of plaintext byte (a random distribution) is a constant. $\Pr_{distingush}$ denotes this probability.

5.2 Plaintext Recovery Method using *ABSAB* Bias and Our Bias Set

The following equation allows us to efficiently use $ABSAB$ bias in the broadcast RC4 attack.

$$\begin{aligned}
&(C_r \mid\mid C_{r+1}) \oplus (C_{r+2+G} \mid\mid C_{r+3+G}) \\
&= (P_r \oplus Z_r \mid\mid P_{r+1} \oplus Z_{r+1}) \oplus (P_{r+2+G} \oplus Z_{r+2+G} \mid\mid P_{r+3+G} \oplus Z_{r+3+G}) \\
&= (P_r \oplus P_{r+2+G} \oplus Z_r \oplus Z_{r+2+G} \mid\mid P_{r+1} \oplus P_{r+3+G} \oplus Z_{r+1} \oplus Z_{r+3+G}).
\end{aligned} \tag{2}$$

Assuming that Eq. (1) (event of the $ABSAB$ bias) holds, the relation of plaintexts and ciphertexts without keystreams is obtained, i.e., $(C_r \mid\mid C_{r+1}) \oplus (C_{r+2+G} \mid\mid C_{r+3+G}) = (P_r \oplus P_{r+2+G} \mid\mid P_{r+1} \oplus P_{r+3+G}) = (P_r \mid\mid P_{r+1}) \oplus (P_{r+2+G} \mid\mid P_{r+3+G})$.

However, in the straight way, we can not combine these relations with different G to enhance the biases, as we do in the distinguishing attack setting. When the value of G is different, the above equation is surely different even if r is properly chosen. For example, in the cases of (r and $G = 1$) and ($r + 1$ and $G = 0$), right parts of equations are given as $(P_r \mid\mid P_{r+1}) \oplus (P_{r+3} \mid\mid P_{r+4})$ and $(P_{r+1} \mid\mid P_{r+2}) \oplus (P_{r+3} \mid\mid P_{r+4})$, respectively. Thus, due to independent use of these equations with different G, we are not able to efficiently make use of $ABSAB$ bias in the broadcast setting.

In order to get rid of this problem, we give a method that sequentially recovers the plaintext after P_{258} with the knowledge of pre-guessed plaintext bytes. For example, in the cases of (r and $G = 1$) and ($r + 1$ and $G = 0$), if P_r, P_{r+1}, and P_{r+2} are already known, the two equations with respected to ($P_{r+3} \mid\mid P_{r+4}$) is obtained by transposing P_r, P_{r+1}, and P_{r+2} to the left part of the equation. Then, these equations with different G can be merged.

Suppose that $P_1, P_2, \ldots, P_{257}$ are guessed by our cumulative bias set of the initial bytes, where the success probability of finding these bytes are evaluated in Section 4. Then we aim to sequentially find P_r for $r = 258, 259, \ldots, P_{MAX}$ by using $ABSAB$ biases of $G = 0, 1, \ldots, G_{MAX}$. The detailed procedures are given as follows.

Step 1 Obtain $C_{258-3-G_{MAX}}, C_{258-2-G_{MAX}}, \ldots, C_{P_{MAX}}$ in each ciphertext, and make frequency tables $T_{count}[r][G]$ of $(C_{r-3-G} \mid\mid C_{r-2-G}) \oplus (C_{r-1} \mid\mid C_r)$ for all $r = 258, 259, \ldots, P_{MAX}$ and $G = 0, 1, \ldots, G_{MAX}$, where $(C_{r-3-G} \mid\mid C_{r-2-G}) \oplus (C_{r-1} \mid\mid C_r) = (P_{r-3-G} \mid\mid P_{r-2-G}) \oplus (P_{r-1} \mid\mid P_r)$ only if Eq. (1) holds.

Step 2 Set $r = 258$.

Step 3 Guess the value of P_r.

- **Step 3.1** For $G = 0, 1, \ldots, G_{MAX}$, convert $T_{count}[r][G]$ into a frequency table $T_{marge}[r]$ of $(P_{r-1} \mid\mid P_r)$ by using pre-guessed values of $P_{r-3-G_{MAX}}, \ldots, P_{r-2}$, and merge counter values of all tables.
- **Step 3.2** Make a frequency table $T_{guess}[r]$ indexed by only P_r from $T_{marge}[r]$ with knowledge of the P_{r-1}. To put it more precisely, using a pre-guessed value of P_{r-1}, only Tables $T_{marge}[r]$ corresponding to the value of P_{r-1} is taken into consideration. Finally, regard most frequency one in table $T_{guess}[r]$ as the correct P_r.

Step 4 Increment r. If $r = P_{MAX} + 1$, terminate this algorithm. Otherwise, go to Step 3.

The bytes of the plaintext are correctly extracted from $T_{marge}[r]$ only if it is distinguished from other $N^2 - 1$ wrong candidate distributions. Assuming that wrong candidates are randomly distributed, a probability of the correct extraction from $T_{marge}[r]$ is estimated as $(\Pr_{distingush})^{N^2-1}$. In Step 3.2, our method converts $T_{marge}[r]$ into $T_{guess}[r]$ by using knowledge of P_{r-1}, where $T_{guess}[r]$ has $N - 1$ wrong candidates. It enables us to reduce the number of wrong candidates from $N^2 - 1$ to $N - 1$. Then, a probability of the correct extraction from $T_{guess}[r]$ is estimated as $(\Pr_{distingush})^{N-1}$, which is $1/(\Pr_{distingush})^{N+1}$ times higher than that of $T_{marge}[r]$. Therefore, the table reduction technique of Step 3.2 enables us to further optimize the attack.

Experimental Results: We perform practical experiments using our algorithm to find P_{258}, P_{259}, P_{260}, and P_{261} ($P_{MAX} = 261$). As a parameter of $ABSAB$ bias, $G_{MAX} = 63$ is chosen, because the increase of D is converged around $G_{MAX} = 63$. Then, D is estimated as $D = 2^{-28.0}$. The success probability of our algorithm for recovering P_r ($r \geq 258$) when 2^{30} to 2^{34} ciphertexts are given is shown in Table 4, where the number of tests is 256. Note that $P_1, P_2, \ldots, P_{257}$ are obtained by using our bias set (candidate one) with success probability as shown in Fig. 9. For this experiment, it requires about one week if one uses a single CPU core (Intel(R) Core(TM) i7 CPU 920@ 2.67GHz) to get the result of one plaintext, where 256 plaintexts are used.

Interestingly, given 2^{34} ciphertexts, P_{258}, P_{259}, P_{260}, and P_{261} can be recovered with probability one, while the success probability of some bytes in $P_1, P_2, \ldots, P_{257}$ is not one. Combining multiple biases allows us to omit negative effects of some uncorrected value of $P_1, P_2, \ldots, P_{257}$. Although our experiment is performed until P_{261}, the success probability is expected not to change even in the case of later bytes, because $ABSAB$ bias is a long-term bias.

Let us discuss the success probability of extracting bytes after P_{262} when 2^{34} ciphertexts are given. According to Lemma 2 and $D = 2^{-28.0}$, 2^{34} ciphertexts allow us to distinguish an RC4 keystream from a random stream with the probability of $\Pr_{distinguish} = 1 - 10^{-19}$. Then, assuming that wrong candidates are randomly distributed, the probability of correctly extracting the candidate from $(N - 1)$ wrong candidates is estimated as $(\Pr_{distinguish})^{N-1}$. Therefore, our method enables to extract consecutive $(257 + X)$ bytes of a plaintext with the probability of $((\Pr_{distinguish})^{N-1})^X = (\Pr_{distinguish})^{(N-1)\cdot X}$. For instance, when $X = 2^{40}$ and $X = 2^{50}$, the success probabilities are estimated as 0.99997 and 0.97170, respectively.

As a result, by using our sequential method, a large amount of plaintext bytes, e.g., first 2^{50} bytes $\approx$ 1000 T bytes, is recovered from 2^{34} ciphertext with a probability of almost one. Therefore, it can be said that our attack is a full plaintext recovery attack on broadcast RC4, the first of its kind proposed in the literature.

Table 4. Success Probability of our algorithm for recovering P_r ($r \geq 258$).

# of ciphertexts	P_{258}	P_{259}	P_{260}	P_{261}
2^{30}	0.003906	0.003906	0.000000	0.000000
2^{31}	0.039062	0.007812	0.003906	0.007812
2^{32}	0.386719	0.152344	0.070312	0.027344
2^{33}	0.964844	0.941406	0.921875	0.902344
2^{34}	1.000000	1.000000	1.000000	1.000000

6 Conclusion

In this paper, we have evaluated the practical security of RC4 in the broadcast setting. After the introduction of four new biases of the keystream of RC4, i.e., the conditional bias of Z_1, the biases of $Z_3 = 131$ and $Z_r = r$ for $3 \leq r \leq 255$, and the extended keylength-dependent biases, a cumulative list of strongest known biases in $Z_1, Z_2, \ldots, Z_{257}$ is given. Then, we demonstrate a practical plaintext recovery attack using our bias set by a computer experiment. As a result, most bytes of $P_1, P_2, \ldots, P_{257}$ could be extracted with probability more than 0.8 using 2^{32} ciphertexts encrypted by randomly-chosen keys. Finally, we have proposed an efficient method to extract bytes of plaintexts after P_{258}. Our attack is able to recover any plaintext byte from only ciphertexts generated using different keys. For example, first 2^{50} bytes of the plaintext are expected to be recovered from 2^{34} ciphertexts with high probability.

Note that our attack on broadcast RC4, as proposed in this paper, utilizes the advantage of sequential recovery of plaintext bytes. If the initial 256/512/768 bytes of the keystream are suppressed in the protocol, as recommended in case of RC4 usages [14], our attack does not work any more. However, widely-used protocols such as SSL/TLS use initial bytes of the keystream. For SSL/TLS, the broadcast setting is converted into the multi-session setting where the target plaintext block are repeatedly sent in the same position in the plaintexts in multiple SSL/TLS sessions [2].

Our evaluation reveals that broadcast RC4 is practically vulnerable to the plaintext recovery attacks as moderate amount of ciphertexts, i.e., 2^{24} to 2^{34} ciphertexts generated by different keys, leaks considerable information about the plaintext. Thus, RC4 is not to be recommended for the broadcast encryption in case of the typical broadcast setting, while we have not found a practical situation that justifies the availability of 2^{24} to 2^{34} ciphertexts.

Acknowledgments We would like to thank to Sourav Sen Gupta and the anonymous referees for their fruitful comments and suggestions. We also would like to thank to Tubasa Tsukaune and Atsushi Nagao for insightful discussions. This work was supported in part by Grant-in-Aid for Scientific Research (C) (KAKENHI 23560455) for Japan Society for the Promotion of Science and Cryptography Research and Evaluation Committee (CRYPTREC).

References

1. Eli Biham and Yaniv Carmeli. Efficient Reconstruction of RC4 Keys from Internal States. In Kaisa Nyberg, editor, *FSE*, volume 5086 of *Lecture Notes in Computer Science*, pages 270–288. Springer, 2008.
2. Brice Canvel, Alain P. Hiltgen, Serge Vaudenay, and Martin Vuagnoux. Password Interception in a SSL/TLS Channel. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 583–599. Springer, 2003.
3. Scott R. Fluhrer and David A. McGrew. Statistical Analysis of the Alleged RC4 Keystream Generator. In Bruce Schneier, editor, *FSE*, volume 1978 of *Lecture Notes in Computer Science*, pages 19–30. Springer, 2000.
4. Jovan Dj. Golic. Linear Statistical Weakness of Alleged RC4 Keystream Generator. In Walter Fumy, editor, *EUROCRYPT*, volume 1233 of *Lecture Notes in Computer Science*, pages 226–238. Springer, 1997.
5. Sourav Sen Gupta, Subhamoy Maitra, Goutam Paul, and Santanu Sarkar. Proof of Empirical RC4 Biases and New Key Correlations. In Ali Miri and Serge Vaudenay, editors, *Selected Areas in Cryptography*, volume 7118 of *Lecture Notes in Computer Science*, pages 151–168. Springer, 2011.
6. Sourav Sen Gupta, Subhamoy Maitra, Goutam Paul, and Santanu Sarkar. (Non-)Random Sequences from (Non-)Random Permutations - Analysis of RC4 stream cipher. Journal of Cryptology, 2012. (to appear).
7. Lars R. Knudsen, Willi Meier, Bart Preneel, Vincent Rijmen, and Sven Verdoolaege. Analysis Methods for (Alleged) RC4. In Kazuo Ohta and Dingyi Pei, editors, *ASIACRYPT*, volume 1514 of *Lecture Notes in Computer Science*, pages 327–341. Springer, 1998.

8. Subhamoy Maitra, Goutam Paul, and Sourav Sengupta. Attack on Broadcast RC4 Revisited. In Antoine Joux, editor, *FSE*, volume 6733 of *Lecture Notes in Computer Science*, pages 199–217. Springer, 2011.
9. Itsik Mantin. Analysis of the stream cipher rc4. Master's Thesis, The Weizmann Institute of Science, Israel, 2001. `http://www.wisdom.weizmann.ac.il/~itsik/RC4/rc4.html`.
10. Itsik Mantin. Predicting and Distinguishing Attacks on RC4 Keystream Generator. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 491–506. Springer, 2005.
11. Itsik Mantin and Adi Shamir. A Practical Attack on Broadcast RC4. In Mitsuru Matsui, editor, *FSE*, volume 2355 of *Lecture Notes in Computer Science*, pages 152–164. Springer, 2001.
12. Mitsuru Matsui. Key Collisions of the RC4 Stream Cipher. In Orr Dunkelman, editor, *FSE*, volume 5665 of *Lecture Notes in Computer Science*, pages 38–50. Springer, 2009.
13. Alexander Maximov and Dmitry Khovratovich. New State Recovery Attack on RC4. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 297–316. Springer, 2008.
14. Ilya Mironov. (Not So) Random Shuffles of RC4. In Moti Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 304–319. Springer, 2002.
15. Goutam Paul and Subhamoy Maitra. Permutation After RC4 Key Scheduling Reveals the Secret Key. In Carlisle M. Adams, Ali Miri, and Michael J. Wiener, editors, *Selected Areas in Cryptography*, volume 4876 of *Lecture Notes in Computer Science*, pages 360–377. Springer, 2007.
16. Souradyuti Paul and Bart Preneel. A New Weakness in the RC4 Keystream Generator and an Approach to Improve the Security of the Cipher. In Bimal K. Roy and Willi Meier, editors, *FSE*, volume 3017 of *Lecture Notes in Computer Science*, pages 245–259. Springer, 2004.
17. Pouyan Sepehrdad, Serge Vaudenay, and Martin Vuagnoux. Discovery and Exploitation of New Biases in RC4. In Alex Biryukov, Guang Gong, and Douglas R. Stinson, editors, *Selected Areas in Cryptography*, volume 6544 of *Lecture Notes in Computer Science*, pages 74–91. Springer, 2010.

A Detailed Evaluation of Extended Keyength-dependent Bias

We analyze events of E_1 and E_2 causing the extended keylength-dependent bias, In the proof of the keylength-dependent bias [5], the probability $\Pr(Z_l = -l \mid f_{l-1} = -l)$ is given as follows.

Lemma 3 [5] *Given that* $f_{l-1} = -l$*, the probability* $\Pr(Z_l = -l \mid f_{l-1} = -l)$ *is estimated as,*

$$\frac{1}{N} + \left(1 - \frac{1}{N}\right) \cdot \left(\frac{1}{N} + \left(1 - \frac{l}{N}\right) \cdot \left(1 - \frac{1}{N}\right)^{N+l-2} \cdot \left(\left(1 - \frac{1}{N}\right)^{1+l} + \frac{1}{N}\right)\right.$$
$$\left. \cdot \left(\frac{1}{N} + \left(1 - \frac{1}{N}\right)^{l+1} \cdot \Pr(S_0[S_0[l-1]] = f_{l-1})\right)\right),$$

where $\Pr(S_0[S_0[l-1]] = f_{l-1})$ *is computed by* Proposition 4 *in* [5] *for* $1 \leq l \leq 32$*. When* $l > 32$*, it is assumed as* $1/N$*.*

The probability of $Z_r = -r$ $(r = 16, 32, 48, 64, 80, 96, 112)$ is estimated as Theorem 10.

Theorem 10 *The probability* $\Pr(Z_r = -r)$ *is approximately*

$$\begin{aligned}\Pr(Z_r = -r) \approx\ & p_{r-1,2r} \cdot \frac{1}{N} + \Pr(Z_r = -r \mid f_{r-1} = -r) \cdot \Pr(f_{r-1} = -r) \\ & + \Pr(Z_r = -r \mid E_3) \cdot \Pr(E_3) \\ & + \left(1 - p_{r-1,2r} \cdot \frac{1}{N} - \Pr(f_{r-1} = -r) - \Pr(E_3)\right) \cdot \frac{1}{N},\end{aligned}$$

where $p_{r-1,2r} = \Pr(S_{r-1}[r] = 2r)$ *and* $\Pr(E_3) = \Pr(f_{r-1} \neq -r \wedge S_{r-1}[r] = 0)$.

Proof. For $i_r = r$, an output Z_r is expressed as

$$Z_r = S_r[S_r[i_r] + S_r[j_r]] = S_r[S_r[r] + S_{r-1}[r]].$$

Case 1 : $S_{r-1}[r] = 2r \wedge S_r[r] = -r$
Case 2 : $f_{r-1} = -r$

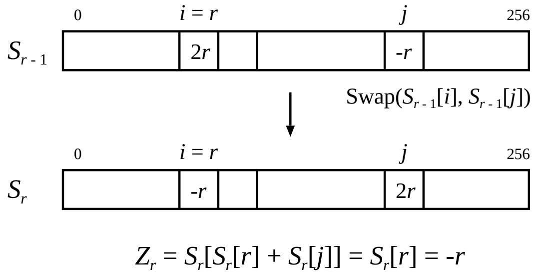

Fig. 12. Event (Case 1) for Bias of $Z_r = -r$

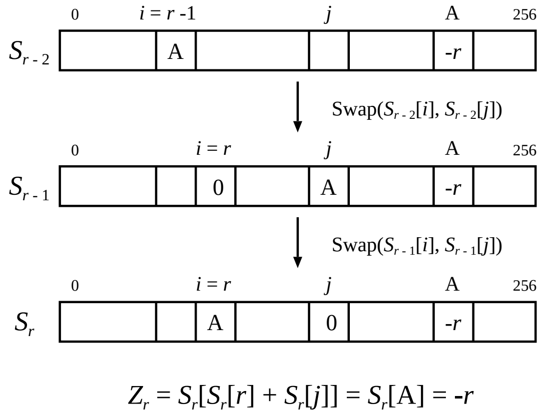

Fig. 13. Event (Case 2) for Bias of $Z_r = -r$

Case 3 : $f_{r-1} \neq -r \wedge S_{r-1}[r] = 0$

In Case 1, the output is always $Z_r = -r$. Case 2 is an expanded event of Lemma 3. The event of Case3 is experimentally found, and theoretical reason is not given.

Case 1 : $S_{r-1}[r] = 2r \wedge S_r[r] = -r$
The output is expressed as $Z_r = S_r[S_r[r] + S_{r-1}[r]] = S_r[(-r) + 2r] = S_r[r] = -r$. Then, the probability of $\Pr(Z_r = -r)$ is one. Figure 12 shows this event. Since $S_r[r]$ is chosen by pointer j, we assume that there is no bias [8].

$$\Pr(S_{r-1}[r] = 2r \wedge S_r[r] = -r) = p_{r-1,2r} \cdot \frac{1}{N},$$

where $p_{r-1,2r} = \Pr(S_{r-1}[r] = 2r)$.
Case 2 : $f_{r-1} = -r$
Lemma 3 is expanded so that the event of $Z_r = -r \mid f_{r-1} = -r$ is covered when $l = 16$. If $S_{r-1}[r] = 0 \wedge S_r[S_{r-2}[r-1]] = -r$, we will always have $Z_r = S_r[S_r[r] + S_{r-1}[r]] = S_r[0 + S_{r-2}[r-1]] = S_r[S_{r-2}[r-1]] = -r$. Given that $f_{l-1} = -l$, Lemma 3 considers the event of $S_{l-1}[l] = 0 \wedge S_l[S_{l-2}[l-1]] = -l$. Then, the probability $\Pr(Z_r = -r \mid E_2)$ is estimated as follows.

$$\begin{aligned} \Pr(Z_r = -r \mid E_2) \cdot \Pr(E_2) \approx \Pr(f_{r-1} = -r) \cdot \Bigg(\frac{1}{N} + \left(1 - \frac{1}{N}\right) \\ \cdot \left(\frac{1}{N} + \left(1 - \frac{r}{N}\right) \cdot \left(1 - \frac{1}{N}\right)^{N+r-2} \cdot \left(\left(1 - \frac{1}{N}\right)^{1+r} + \frac{1}{N} \right) \right) \\ \cdot \left(\frac{1}{N} + \left(1 - \frac{1}{N}\right)^{r+1} \cdot \Pr(S_0[S_0[r-1]] = f_{r-1}) \right) \Bigg) \end{aligned}$$

Figure 13 shows this event. Here, let us consider the probability $\Pr(f_{r-1} = -r)$. When keylength l is 16 bytes, $K[x] = K[x \bmod 16]$ is obtained. Thus if $r = 16 \cdot k$, $f_{r-1} = -r$ is given as.

$$\begin{aligned} f_{r-1} &= k \cdot (K[0] + \ldots + K[15]) + \frac{16 \cdot k \cdot (16 \cdot k - 1)}{2} \\ &= -16 \cdot k \pmod{N}, \end{aligned}$$

and it holds only if the secret key satisfies the following equation.

$$k \cdot (K[0] + \ldots + K[15]) = -16 \cdot k - \frac{16 \cdot k \cdot (16 \cdot k - 1)}{2} \pmod{N}.$$

When $k = 1, 3, 5, 7$, since the above equation has only one solution of $(K[0] + \ldots + K[15])$, the probability is $1/N$. When $k = 2$ and $k = 6$, since there are two solutions, the probability is $2/N$. When $k = 4$, since

there are four solutions, the probability is $4/N$. This property is experimentally confirmed.
Case 3 : $f_{r-1} \neq -r \wedge S_{r-1}[r] = 0$
Case 3 is experimentally found event, and theoretical proof is open.

Thus, the probability $\Pr(Z_r = -r)$ is approximately

$$\begin{aligned}\Pr(Z_r = -r) \approx\ & p_{r-1,2r} \cdot \frac{1}{N} + \Pr(Z_r = -r \mid f_{r-1} = -r) \cdot \Pr(f_{r-1} = -r) \\ & + \Pr(Z_r = -r \mid E_3) \cdot \Pr(E_3) \\ & + \left(1 - p_{r-1,2r} \cdot \frac{1}{N} - \Pr(f_{r-1} = -r) - \Pr(E_3)\right) \cdot \frac{1}{N}.\end{aligned}$$

□

$p_{r-1,2r}$ is obtained from Theorem 4.