On non-negligible bias of the first output byte of RC4 towards the first three bytes of the secret key

Goutam Paul · Siddheshwar Rathi · Subhamoy Maitra

Abstract In this paper, we show that the first byte of the keystream output of RC4 has non-negligible bias towards the sum of the first three bytes of the secret key. This result is based on our observation that the index, where the first byte of the keystream output is chosen from, is approximately twice more likely to be 2 than any other value. Our technique is further used to theoretically prove Roos's experimental observation (A class of weak keys in the RC4 stream cipher, 1995) related to weak keys.

Keywords Bias · Cryptanalysis · Keystream · Permutation · RC4 · Stream cipher

1 Introduction

RC4 is one of the most popular stream ciphers in cryptologic literature and it has wide applications in industry till date. This cipher has been analyzed for more than a decade in open

Siddheshwar Rathi was a researcher at Applied Statistics Unit, Indian Statistical Institute, Kolkata. We were shocked by his sudden demise on October 28, 2006. Siddheshwar had observed and proved Theorem 2. However, we, the other two co-authors, noted the consequence of this theorem at a later date. In fact, we were attracted towards the analysis of RC4 due to Siddheshwar's enthusiasm. Unfortunately, Siddheshwar could not see this paper published.

literature and many different weaknesses have been identified. Even then RC4 is believed to be a secure stream cipher when used with certain precautions.

A brief description of RC4 is presented in Sect. 1.1 and necessary background is available in Sect. 1.2. Section 2 contains the contribution of this paper. In Sect. 2.1, we present theoretical results to show for the first time (Theorem 3) that $P(z_1 = (K[0] + K[1] + K[2] + 3) \bmod 256) \approx \frac{1}{256}(1 + 0.37)$, where z_1 is the first byte of the keystream output after the first round of the Pseudo-Random Generation Algorithm (PRGA) of RC4 and $K[0]$, $K[1]$, $K[2]$ are the first three bytes of the secret key. This result is based on (i) Roos's result [11, Section 2, Result B], summarized as Theorem 1 in this paper, that $P(S_0[2] = (K[0]+K[1]+K[2]+3) \bmod 256) \approx 0.37$, where S_0 is the permutation after the complete Key Scheduling Algorithm (KSA), and (ii) more importantly, our observation (Theorem 2) that the index, where the first byte of the keystream output is chosen from, is 2 with probability $\approx \frac{2}{256}$. We emphasize here that Roos's observation [11] was that $S_0[2]$ is correlated to $K[0] + K[1] + K[2] + 3$ (see Corollary 1); but our theoretical result (Theorem 3), that this bias is in fact propagated to z_1, is only due to our Theorem 2. Thus, our results connect the weakness of the KSA and that of the PRGA of RC4.

Further, in Sect. 2.2, we theoretically prove (Theorem 5) Roos's experimental observation [11] that $P(z_1 = (K[2] + 3) \bmod 256 \mid K[0] + K[1] = 0 \bmod 256) \approx 0.13$ and the proof is again based on our newly identified stronger conditional bias (Theorem 4) of the index 2 being chosen (given the condition $K[0] + K[1] = 0 \bmod 256$) while generating the first byte of the keystream output. This observation of 1995 [11] stayed unproved for more than a decade.

Finally, cryptanalytic applications of our results are demonstrated in Sect. 2.3.

Our results clearly point out two important issues in shuffle exchange kind of stream cipher design.

- The assumption that the permutation after the completion of the KSA is random does not provide any guarantee that the indices, from where the keystream output bytes are generated, will be random. The designers should properly take care of the initializations after the KSA and before the PRGA.
- If, due to problems in design, the random-looking permutation after the KSA is biased to some combinations of the bytes in the secret key, then information related to the secret key bytes may be leaked in the first few bytes of the keystream output.

1.1 Description of RC4

The RC4 stream cipher has been designed by Ron Rivest for RSA Data Security in 1987, and was a propriety algorithm until 1994. It uses an S-Box $S = (S[0], \ldots, S[N-1])$ of length N, each location being of 8 bits. Typically, $N = 256$. S is initialized as the identity permutation, i.e., $S[i] = i$ for $0 \le i \le N-1$. A secret key of size typically varying from 40 to 128 bits is used to scramble this permutation. Another array $K = (K[0], \ldots, K[N-1])$ is used to hold the secret key, where each location is of 8 bits. The key is repeated in the array K at key length boundaries. For example, if the key size is 40 bits, then $K[0], \ldots, K[4]$ are filled by the key and then this pattern is repeated to fill up the entire array K.

The RC4 cipher has two components: the Key Scheduling Algorithm (KSA) and the Pseudo-Random Generation Algorithm (PRGA). The KSA turns the random key K into an initial permutation S of $0, 1, \ldots, N-1$ and PRGA uses this permutation to generate a pseudo-random keystream. This keystream output byte z is XOR-ed with the message byte to generate the ciphertext byte at the sender end and again z is XOR-ed with the ciphertext byte to generate the message byte at the receiver end.

The KSA initializes both i and j to 0, and S to be the identity permutation. It then steps i across S looping N times, and updates j by adding the ith bytes of S and K. Each iteration ends with a swap of the two bytes in S pointed by the current values of i and j.

The PRGA also initializes both i and j to 0. It then loops over four operations in sequence: incrementing i as a counter, updating j pseudo-randomly by adding S[i], swapping the two bytes of S pointed by the current values of i and j, and outputting the value of S at index $S[i] + S[j]$ as the value of z.

Algorithm KSA
Initialization:
For $i = 0, \ldots, N-1$
$S[i] = i$;
$j = 0$;
Scrambling:
For $i = 0, \ldots, N-1$
$j = (j + S[i] + K[i])$;
Swap($S[i]$, $S[j]$);

Algorithm PRGA
Initialization:
$i = j = 0$;
Output Key-stream Generation Loop:
$i = i + 1$;
$j = j + S[i]$;
Swap($S[i]$, $S[j]$);
$t = S[i] + S[j]$;
Output $z = S[t]$;

Note that defining the array K to be of size N enables us to write $K[i]$ instead of the typical $K[i \bmod key\,length]$ in the description of the algorithm. This is done for the sake of simplification in the subsequent analysis of the algorithm.

Any addition, used in the RC4 description or in addition of key bytes in this paper, is in general addition modulo N unless specified otherwise. For example, $(K[0] + K[1] + K[2] + 3) \bmod 256$ is usually written as $K[0] + K[1] + K[2] + 3$.

1.2 Background

Cryptanalysis of RC4 and RC4-like stream ciphers can be broadly classified into two categories: attacks based on the weakness of the key scheduling and attacks based on the internal state and round operation of the key generation part.

In exploiting the weakness of the PRGA, distinguishing attacks are the main motivation [1,3,7,5,6,9,10]. In particular, [7] proves a bias in the second output byte being zero and [10] proves a bias in the equality of the first two output bytes.

Initial empirical observations about the correlation between the bytes of the secret key and that of the keystream output were reported in [11,12], but theoretical justifications of these observations were not found. In this paper, we present a more general theoretical framework to find the correlation between the secret key bytes and the first keystream output byte. Roos [11] observed only the conditional correlation when $K[0] + K[1] = 0$ [11, Section 3, Result C] which is not required for our general result that the first output byte is correlated with $K[0] + K[1] + K[2] + 3$ (Theorem 3). Moreover, a consequence of our result (Theorem 5) explains the above empirical observation of [11, Section 3, Result C].

In [2], weaknesses of RC4 key scheduling algorithm have been addressed in great detail and practical attacks have been mounted on several modes of RC4 where IV's are used (e.g. WEP [4]). In [2, Sect. 4], propagation of weak key patterns to the keystream output bytes has been discussed. The first output byte of RC4 has been noted in [2, Sect. 6] considering related key attacks, but they did not observe the relationship between the first output byte and the secret key bytes. In this paper, we do not focus on any weak or related keys for our analysis. We show that given any arbitrary secret key, there is a significant correlation between the first three key bytes and the first byte of the keystream output.

In [8, Sect. 6], a very small non-uniformity in the distribution of the first byte of the keystream output is reported (without any proof). Interestingly, we could not observe this bias after considerable amount of experimentation. We like to mention once again that though several works refer to the first output byte of RC4 keystream, its correlation with the bytes of any arbitrary secret key was surprisingly never identified before our work.

We now present some important technical results identified in [11]. These results will be used further for our analysis.

Lemma 1 *Assume that during the KSA the index j takes its values uniformly at random. Then for any fixed index x, the probability that j does not equal x in any round of the KSA is $(\frac{N-1}{N})^N$.*

Theorem 1 [11] *After the KSA, the most likely value of the xth element of the permutation for the first few values of x is given by*

$$S_0[x] = \sum_{i=0}^{x} K[i] + \frac{x(x+1)}{2}.$$

When x is small enough (e.g., $x <$ the key size), then, as Roos points out in [11], the above equality happens with approximately the same probability as stated in Lemma 1, i.e.,

$$P\left(S_0[x] = \sum_{i=0}^{x} K[i] + \frac{x(x+1)}{2}\right) \approx \left(\frac{N-1}{N}\right)^N.$$

Corollary 1 *After the KSA, the biases of the second and the third bytes of the permutation towards the secret key are given by*

1. $P(S_0[1] = K[0] + K[1] + 1) \approx (\frac{N-1}{N})^N$.
2. $P(S_0[2] = K[0] + K[1] + K[2] + 3) \approx (\frac{N-1}{N})^N$.

2 Our results

For our analysis, we use subscript r to the permutation S, the keystream output byte z and the index t (in S) from where z is chosen to denote the corresponding variables after the rth round of the PRGA. Thus, S_0 denotes the permutation just after the completion of the KSA, i.e., before the first round of the PRGA (as in Theorem 1 above). S_1 denotes the permutation after the first round of the PRGA. The first byte z_1 of the keystream output is selected from the index t_1 of the permutation S_1. As the following expression will be used a number of times in the paper, we denote

$$\phi_N = \left(\frac{N-1}{N}\right)^N \left(1 - \frac{1}{N} - \frac{1}{N^2}\right) + \frac{1}{N^2}.$$

Let us now analyze the first round of the PRGA. Initially, $i = j = 0$. Let $S_0[1] = X$ and $S_0[X] = Y$. In the first round, i is updated to 1 and j is updated to $0 + S_0[1] = X$. Then the contents X and Y of locations 1 and X respectively are interchanged. Thus, $S_1[1] = Y$ and $S_1[X] = X$.

2.1 Results for any arbitrary secret key

In the following theorem, we prove that the index where the first output byte is chosen from is biased. The probability of the first output index being 2 is approximately twice as large as it being any other value.

Theorem 2 *Assume that the initial (just after the KSA) permutation S_0 is chosen uniformly at random from the set of all possible permutations of the set $\{0, 1, \ldots, N-1\}$. Then the probability distribution of the output index t_1, that selects the first byte of the keystream output, is given by*

$$\begin{aligned} P(t_1 = x) &= \tfrac{1}{N}, && \text{for odd } x \\ &= \tfrac{1}{N} - \tfrac{2}{N(N-1)}, && \text{for even } x \neq 2 \\ &= \tfrac{2}{N} - \tfrac{1}{N(N-1)}, && \text{for } x = 2 \end{aligned}$$

Proof In the first round of the PRGA, we have $i = 1$, $j = S_0[1]$ and $S_1[S_0[1]] = S_0[1]$. Thus, $S_1[i] = S_1[1] = S_0[S_0[1]]$ and $S_1[j] = S_1[S_0[1]] = S_0[1]$. Now,

$$\begin{aligned} S_1[j] = 1 &\iff S_0[1] = 1, && \text{since } S_1[j] = S_0[1] \\ &\iff j = 1, && \text{since } j = S_0[1] \\ &\implies S_1[1] = 1, && \text{since we started with } S_1[j] = 1 \\ &\implies S_1[i] = 1, && \text{since we started with } i = 1. \end{aligned}$$

Hence,

$$\begin{aligned} P(S_1[i] = \alpha, S_1[j] = \beta) &= \tfrac{1}{N}, && \text{when } \alpha = 1, \beta = 1 \\ &= 0, && \text{when } \alpha \neq 1, \beta = 1 \\ &= 0, && \text{when } \alpha \neq 1, \beta = \alpha \\ &= \tfrac{1}{N(N-1)}, && \text{otherwise.} \end{aligned}$$

The probability distribution of $t_1 = S_1[i] + S_1[j]$ can be computed as follows.

- For odd x, $P(t_1 = x) = \sum_{\substack{k=0 \\ k \neq 1}}^{N-1} P(S_1[j] = k, S_1[i] = N - k + x)$

$$= (N-1) \cdot \frac{1}{N(N-1)} = \frac{1}{N}.$$

- For even $x \neq 2$, $P(t_1 = x) = \sum_{\substack{k=0 \\ k \neq 1, \frac{x}{2}, \frac{N+x}{2}}}^{N-1} P(S_1[j] = k, S_1[i] = N - k + x)$

$$= (N-3) \cdot \frac{1}{N(N-1)} = \frac{1}{N} - \frac{2}{N(N-1)}.$$

- For $x = 2$, $P(t_1 = 2) = P(S_1[j] = 1, S_1[i] = 1) + \sum_{\substack{k=0 \\ k \neq 1, \frac{N+2}{2}}}^{N-1} P(S_1[j] = k, S_1[i] = N - k + 2)$

$$= \frac{1}{N} + (N-2) \cdot \frac{1}{N(N-1)} = \frac{2N-3}{N(N-1)} = \frac{2}{N} - \frac{1}{N(N-1)}.$$

□

Using our Theorem 2 and Roos's Theorem 1, we can show (Theorem 3) that for any arbitrary key, the first byte of the keystream output is significantly biased to the initial bytes of the secret key, thus revealing a weakness in the KSA. For this, we first present the following technical result that will be used in the proof.

Proposition 1 *After the first round of the PRGA, the bias of the second permutation byte towards the secret key is given by*

$$P(S_1[2] = K[0] + K[1] + K[2] + 3) \approx \phi_N.$$

Proof As in the KSA, the index j of the PRGA is assumed to take values uniformly at random. During the first round of the PRGA, i takes the value 1 and j takes the value $S_0[1]$. Let $f(K) = K[0] + K[1] + K[2] + 3$. Recall from item 2 of Corollary 1 that $P(S_0[2] = f(K)) \approx (\frac{N-1}{N})^N$. The event $(S_1[2] = K[0] + K[1] + K[2] + 3)$ can occur in two ways.

1. After the KSA, $S_0[2] = f(K)$, and there is no swap involving index 2 in the first round of the PRGA. The contribution of this part is
   $P(S_0[2] = f(K)) \cdot P(S_0[1] \neq 2) \approx (\frac{N-1}{N})^N (1 - \frac{1}{N})$.
2. After the KSA, $S_0[2] \neq f(K)$, and $f(K)$ comes into index 2 from index 1 by the swap in the first round of the PRGA. The contribution of this part is
   $P(S_0[2] \neq f(K)) \cdot P(S_0[1] = f(K), S_0[1] = 2)$
   $= P(S_0[2] \neq f(K)) \cdot P(S_0[1] = 2) \cdot P(f(K) = 2)$
   $\approx \left((1 - (\frac{N-1}{N})^N\right) \cdot \frac{1}{N^2}$.

Adding the above two contributions, we get
$P(S_1[2] = f(K)) \approx (\frac{N-1}{N})^N (1 - \frac{1}{N} - \frac{1}{N^2}) + \frac{1}{N^2} = \phi_N$. □

The value of ϕ_N is approximately 0.37 for $N = 256$.

Now we present the result that shows the bias of the first keystream output byte towards the first three bytes of the secret key.

Theorem 3 *For any arbitrary secret key, the correlation between the key bytes and the first byte of the keystream output is given by*

$$P(z_1 = K[0] + K[1] + K[2] + 3) \approx \frac{1}{N}(1 + \phi_N).$$

Proof For the sake of brevity, let $f(K) = K[0] + K[1] + K[2] + 3$. Then

$$\begin{aligned}
P(z_1 = f(K)) &= P(S_1[t_1] = f(K)) \\
&= \sum_{i=0}^{N-1} P(t_1 = i) \cdot P(S_1[t_1] = f(K) \mid t_1 = i) \\
&= \sum_{i=0}^{N-1} P(t_1 = i) \cdot P(S_1[i] = f(K)) \\
&= P(t_1 = 2) \cdot P(S_1[2] = f(K)) + \sum_{\substack{i=0 \\ even\ i \neq 2}}^{N-1} P(t_1 = i) \cdot P(S_1[i] = f(K)) \\
&\quad + \sum_{\substack{i=0 \\ odd\ i}}^{N-1} P(t_1 = i) \cdot P(S_1[i] = f(K))
\end{aligned}$$

$$
\begin{aligned}
&= \left(\frac{2}{N} - \frac{1}{N(N-1)}\right) \cdot P(S_1[2] = f(K)) \\
&\quad + \sum_{\substack{i=0 \\ even\ i \neq 2}}^{N-1} \left(\frac{1}{N} - \frac{2}{N(N-1)}\right) \cdot P(S_1[i] = f(K)) \\
&\quad + \sum_{\substack{i=0 \\ odd\ i}}^{N-1} \frac{1}{N} \cdot P(S_1[i] = f(K)) \text{ (by Theorem 2)} \\
&\approx \frac{2}{N} \cdot P(S_1[2] = f(K)) + \sum_{\substack{i=0 \\ even\ i \neq 2}}^{N-1} \frac{1}{N} \cdot P(S_1[i] = f(K)) \\
&\quad \left(as \frac{1}{N(N-1)} \ll \frac{1}{N}\right) + \sum_{\substack{i=0 \\ odd\ i}}^{N-1} \frac{1}{N} \cdot P(S_1[i] = f(K)) \\
&= \frac{1}{N} \cdot P(S_1[2] = f(K)) + \frac{1}{N} \cdot P(S_1[2] = f(K)) \\
&\quad + \frac{1}{N} \cdot \sum_{\substack{i=0 \\ even\ i \neq 2}}^{N-1} P(S_1[i] = f(K)) + \frac{1}{N} \cdot \sum_{\substack{i=0 \\ odd\ i}}^{N-1} P(S_1[i] = f(K)) \\
&= \frac{1}{N} \cdot P(S_1[2] = f(K)) + \frac{1}{N} \cdot \sum_{i=0}^{N-1} P(S_1[i] = f(K)) \\
&\approx \frac{1}{N} \cdot \phi_N + \frac{1}{N} \cdot 1 \quad \text{(by Proposition 1)} \\
&= \frac{1}{N}(1 + \phi_N).
\end{aligned}
$$

□

Let us now present the detailed interpretation of the probabilities involved in Corollary 1, Proposition 1 and Theorem 3. The event considered here is the equality of two random variables X and Y, where X is some byte of the permutation (e.g. $S[i]$ as in Corollary 1 and Proposition 1) or some byte of the keystream output (e.g. z_1 as in Theorem 3), and Y is some function $f(K)$ of the key. Each of X and Y can take values from $\{0, \ldots, N-1\}$. Thus the joint space of (X, Y) consists of N^2 different points (x, y), where $x \in \{0, \ldots, N-1\}$ and $y \in \{0, \ldots, N-1\}$. If X and Y are independently and identically distributed (*iid*) uniform random variables, then for any (x, y), $P(X = x, Y = y)$ should be $\frac{1}{N^2}$, and $P(X = Y) = \sum_{x=0}^{N-1} P(X = x, Y = x) = \sum_{x=0}^{N-1} \frac{1}{N^2} = N \cdot \frac{1}{N^2} = \frac{1}{N}$. For $N = 256$, this value is 0.0039, whereas the observed values of the probabilities are much higher. In case of Proposition 1, this is 0.37 and in case of Theorem 3, this is $\frac{1}{N}(1 + \phi_N) \approx \frac{1}{256} \cdot (1 + 0.37) \approx 0.0053$. Thus, according to Theorem 3, knowledge of z_1 reduces the uncertainty about the secret key by $-\log_2(0.0039) - (-\log_2(0.0053)) = 0.44$ bit.

Note that the bias in S observed by Roos (Theorem 1, item 2 of Corollary 1) and extended in our Proposition 1 do not necessarily imply the bias in z_1. For example, assume that $S_0[2]$ (or $S_1[2]$) equals some combination of the bytes of the key with probability 1. The output z_1

may still be unbiased, if the index t_1 from where z_1 is selected is uniformly random. However, we have proved (Theorem 2) that t_1 is not uniformly random. In other words, there may exist some bias in $S_0[2]$ due to the weakness of the KSA. But it is the bias in t_1 due to the weakness of the PRGA that propagates the bias from $S_0[2]$ to z_1. The proof of our Theorem 3 connects the bias in $S_0[2]$ and that in t_1 and relates them to the bias in z_1.

What we have discussed so far applies to any arbitrary secret key. However, similar biases (both in the index t_1 and in the value z_1 of the first byte of the keystream output) can be observed with much higher probabilities if we assume the condition that $K[0] + K[1] = 0$. We provide theoretical results for this special case in the next section.

2.2 Results for secret keys whose first two bytes sum to zero

Theorem 2 shows that there is a significant bias in the index in S from where the first byte of the keystream output is selected. In the following theorem, we show that this bias is increased significantly if the first two key bytes satisfy the condition $K[0] + K[1] = 0$. In the proof of this theorem as well as in the proof of the next theorem of this section, we use the fact that if event $A \implies$ event B, then $B \supseteq A$ and hence $P(B) \geq P(A)$.

Theorem 4 *Assume that the first two bytes $K[0]$, $K[1]$ of the secret key add to* 0 mod N, *and the initial (just after the KSA) permutation S_0 is chosen uniformly at random from the set of all possible permutations of the set $\{0, 1, \ldots, N-1\}$. Then the bias of the output index t_1, that selects the first byte of the keystream output, is given by*

$$P(t_1 = 2 \mid K[0] + K[1] = 0) > \left(\frac{N-1}{N}\right)^N.$$

Proof

$$\begin{aligned}
P(t_1 = 2 \mid K[0] + K[1] = 0) &= P(S_1[i] + S_1[j] = 2 \mid K[0] + K[1] = 0) \\
&= P(S_1[i] = 1, S_1[j] = 1 \mid K[0] + K[1] = 0) \\
&\quad + \sum_{\substack{k=0 \\ k \neq 1, \frac{N+2}{2}}}^{N-1} P(S_1[j] = k, S_1[i] \\
&= N - k + 2 \mid K[0] + K[1] = 0) \\
&> P(S_1[i] = 1, S_1[j] = 1 \mid K[0] + K[1] = 0) \\
&= P(S_0[1] = 1 \mid K[0] + K[1] = 0) \\
&\quad \text{(as } (S_1[i] = 1 \text{ and } S_1[j] = 1) \iff (S_0[1] = 1) \text{)} \\
&\geq P(S_0[1] = K[0] + K[1] + 1) \\
&\quad \text{(because this event implies the conditional event above)} \\
&= \left(\frac{N-1}{N}\right)^N \quad \text{(by Corollary 1, item 1).}
\end{aligned}$$

□

To the best of our knowledge, the bias in the output index (Theorem 2), the conditional bias in the output index (Theorem 4), and the bias of the first byte of the output towards the first three bytes of the secret key (Theorem 3) have not been observed before our work. However, Roos [11] experimentally observed that given any RC4 key with the restriction

$K[0] + K[1] = 0$, the probability that the first byte generated by RC4 will be $K[2] + 3$ ranges between 12% and 16% with an average of 13.8%. We, for the first time, provide a theoretical justification of this observation in Theorem 5 below. The proof of this theorem depends on the conditional bias of the index being 2 (Theorem 4) which has been proved above.

Theorem 5 *Assume that the first two bytes $K[0]$, $K[1]$ of the secret key add to* 0 mod N. *Then the correlation between the key bytes and the first byte of the keystream output is given by*

$$P(z_1 = K[2] + 3 \mid K[0] + K[1] = 0) > \left(\frac{N-1}{N}\right)^N \phi_N.$$

Proof

$$\begin{aligned}
P(z_1 = K[2]+3 \mid K[0]+K[1]=0) &= P(S_1[t_1] = K[2]+3 \mid K[0]+K[1]=0) \\
&= \sum_{i=0}^{N-1} P(t_1 = i \mid K[0]+K[1]=0) \cdot P(S_1[t_1] \\
&= K[2]+3 \mid K[0]+K[1]=0, t_1 = i) \\
&= \sum_{i=0}^{N-1} P(t_1 = i \mid K[0]+K[1]=0) \cdot P(S_1[i] \\
&= K[2]+3 \mid K[0]+K[1]=0) \\
&> P(t_1 = 2 \mid K[0]+K[1]=0) \cdot P(S_1[2] \\
&= K[2]+3 \mid K[0]+K[1]=0) \\
&\geq P(t_1 = 2 \mid K[0]+K[1]=0) \cdot P(S_1[2] \\
&= K[0]+K[1]+K[2]+3) \\
&\quad \text{(because this 2nd event implies the conditional 2nd event in the above step)} \\
&> \left(\frac{N-1}{N}\right)^N \phi_N \\
&\quad \text{(by Theorem 4 and Proposition 1)}
\end{aligned}$$

□

For $N = 256$, the value of $\left(\frac{N-1}{N}\right)^N \phi_N$ is approximately 0.13 that conforms with the experimental observation of [11]. Thus, knowledge of z_1 in this case reduces the uncertainty about the secret key (as also pointed out in [11]) by at least $-\log_2(0.0039) - (-\log_2(0.13)) \approx 5.1$ bits.

2.3 Cryptanalytic applications

We present this section to demonstrate applications of Theorems 3 and 5.

Let m_1 and c_1 denote the first bytes of the message and the ciphertext respectively. Then $c_1 = m_1 \oplus z_1$. One can use [7, Theorem 2] to calculate the number of samples that suffice to mount our cryptanalysis. Theorem 2 of [7] states that if the event e happens in distribution X with probability p and in distribution Y with probability $p(1+q)$, then for small p and q, $O\left(\frac{1}{pq^2}\right)$ samples suffice to distinguish X from Y with a *constant* probability of success. In

our case, let X be the joint distribution of the two variables $(K[0]+K[1]+K[2]+3)$ and z_1, when $K[0]$, $K[1]$, $K[2]$, z_1 are chosen uniformly at random from 0 to $N-1$, and Y be the joint distribution of the same two variables for RC4 for randomly chosen keys. Let e be the event that these two variables are equal. Here $p = \frac{1}{N}$ and $q = \phi_N$ (following Theorem 3). Thus the number of samples required is $\frac{1}{pq^2} = \frac{N}{\phi_N^2}$. For $N = 256$, this value turns out to be 1870.

2.3.1 When the IV precedes the secret key

Consider that the same message is broadcasted after being encrypted by RC4 with uniformly distributed keys for multiple recipients. If the first three bytes of each of the keys are known, then one can calculate $m_1' = c_1 \oplus (K[0]+K[1]+K[2]+3)$ for each of the keys. According to Theorem 3, the most frequent value of m_1' will give the actual value of m_1 with high probability.

Note that knowing the first three bytes of the key is not always impractical. In Wireless Equivalent Privacy (WEP) protocol [4], a secret key is used with known IV modifiers in RC4 [2,5]. If the IV bytes precede the secret key bytes then the first three bytes of the key are actually known. Our analysis is different from [2,5], as we use the first byte of the ciphertext output only. In [7], the second byte of the ciphertext is used for cryptanalysis in broadcast mode.

Further, if one can ensure that the first two bytes of the key add to zero, then following Theorem 5, one can perform the attack much more efficiently. Empirically we have checked that only 25 samples suffice to recover m_1 with an average probability of success exceeding 0.80 in this case.

Table 1 shows the success probability of recovering m_1 when 3 bytes IV is preceded by 5 bytes secret key. For a given m_1, we choose n (values of n appear in the second column) number of samples, i.e., n many first byte c_1 of the ciphertexts corresponding to n many runs of RC4, each time with a randomly chosen $< IV(3\ bytes), Key(5\ bytes) >$ pair. We compute $m_1' = c_1 \oplus (K[0]+K[1]+K[2]+3)$ for each of the n samples and find out the most frequently occurring value m_1'. If this value matches with the given m_1 then the recovery is considered to be successful. To estimate the success probability, we repeat the above process with 1000 randomly chosen m_1 values and find out in how many cases, say τ, m_1 is recovered successfully. That is, the success probability is $\frac{\tau}{1000}$.

We repeat the above experiment 100 times to get a set of 100 probabilities and then we compute the minimum, maximum, average and standard deviation of the success probabilities. This is presented in Table 1. The low values of the standard deviation confirms that the success probability is constant.

Table 1 Results with 3 bytes IV preceding 5 bytes secret key

Condition	#Samples	min	max	avg	sd
Unconditional	1870	0.020000	0.062000	0.037990	0.008738
Unconditional	25000	0.574000	0.667000	0.628230	0.019447
Unconditional	50000	0.914000	0.955000	0.933380	0.007945
$K[0]+K[1]=0$	25	0.740000	0.922000	0.806840	0.039063
$K[0]+K[1]=0$	50	0.957000	0.989000	0.975410	0.007297
$K[0]+K[1]=0$	100	0.998000	1.000000	0.999790	0.000431

Table 2 Results with 11 bytes IV following 5 bytes secret key

Condition	#Samples	min	max	avg	sd
Unconditional	1870	0.012000	0.032000	0.022540	0.004318
Unconditional	25000	0.113000	0.164000	0.144340	0.010867
Unconditional	50000	0.125000	0.182000	0.152910	0.011410
$K[0]+K[1]=0$	25	0.759000	0.821000	0.789010	0.013393
$K[0]+K[1]=0$	50	0.921000	0.968000	0.943020	0.007708
$K[0]+K[1]=0$	100	0.952000	0.977000	0.965300	0.005442

2.3.2 *When the IV follows the secret key*

Assume that we can observe the first byte z_1 of the keystream output. Then with probability 0.0053 we know the value of $K[0]+K[1]+K[2]+3$. Note that if the key is not appended with IV, then the same secret key would give rise to the same keystream (and hence the same z_1) each time. Appending different IV's makes the keystream changing and helps in achieving a frequency distribution of z_1. Then the most frequent value of z_1 can be treated as the value of $K[0]+K[1]+K[2]+3$. We need the same secret key to be appended by different IV's to generate z_1 for recovering the value of $K[0]+K[1]+K[2]+3$ reliably.

Further, if one can ensure that the first two bytes of the key add to zero ($IV[0]+IV[1]=0$ for Sect. 2.3.1 and $K[0]+K[1]=0$ for Sect. 2.3.2), then following Theorem 5, one can perform the attack (e.g. recovering $K[2]$) much more efficiently requiring a very few (practically ≤ 100) samples.

Table 2 shows the success probability of recovering the sum of the first three bytes when 11 bytes IV is followed by 5 bytes secret key. We use the same experimental setup as in Sect. 2.3.1. The only difference is that in order to generate a distribution of z_1, we need to assume that a different IV is used with each of the n samples.

Note that here the success probability is less compared to the case where the IV bytes precede the secrete key. In the previous case, we use a different IV and a different key for each sample, so that the effective key is uniformly randomly distributed. However, here we use many different IV's with the same key n number of times. If the number of IV bytes is not sufficiently greater than the number of key bytes, then the key is not uniformly randomly distributed. That is why, we assume 11 byte IV and 5 byte secret key for the experiments. Even with such large IV's, the success probabilities are much less than the previous case due to non-uniformity in the distribution of the effective keys (i.e., same secret key appended by different IV's).

3 Conclusion

In this paper, we identify and prove that after the KSA of RC4, the index, where the first byte z_1 of the keystream output is chosen from, is 2 with a non-negligible bias. This in turn leaks the sum of the first three bytes of the secret key through z_1. Under the assumption that the initial few hundred keystream output bytes of RC4 may be discarded, this information leakage (similar to many other published attacks in this area) will not provide any threat to the practical use of RC4. However, the result clearly points out an unobserved structural