Some results on RC4 in WPA

Sourav Sen Gupta[1], Subhamoy Maitra[1], Willi Meier[2], Goutam Paul[1], and Santanu Sarkar[3]

[1] Indian Statistical Institute, Kolkata, India
[2] FHNW, Windisch, Switzerland
[3] Chennai Mathematical Institute, Chennai, India

Abstract. Motivated by the work of AlFardan et al 2013, in this paper we present several results related to RC4 non-randomness in WPA. We first prove the interesting zig-zag distribution of the first byte and the similar nature for the biases in the initial keystream bytes to zero. As we note, this zig-zag nature surfaces due to the dependency of first and second key bytes in WPA/TKIP, both derived from the same byte of the IV. Further, we also note that the correlation of certain keystream bytes to the first three IV bytes provides much higher biases than what had been presented in the work by AlFardan et al 2013. We notice that the correlations of the keystream bytes with publicly known IV values of WPA potentially strengthens the practical plaintext recovery attack on the protocol; formulation of the exact details related to this attack is in progress.

Keywords: RC4, Bias, Plaintext Recovery, TKIP, WPA.

1 Introduction

RC4 or Alleged RC4 is the most widely deployed commercial stream cipher, having applications in protocols like SSL, WEP, WPA and in Microsoft Windows, Apple OCE, Secure SQL, etc. The cipher consists of a Key Scheduling Algorithm (KSA) and a Pseudorandom Generation Algorithm (PRGA). The internal state of RC4 is obtained as a permutation of all 8-bit words, i.e., a permutation of $N = 2^8 = 256$ bytes, and the KSA produces the initial pseudorandom permutation of RC4 by scrambling an identity permutation using the secret key k. The secret key k of RC4 is of length typically between 5 to 32 bytes, which generates the expanded key K of length $N = 256$ bytes by simple repetition. If the length of the secret key $k = k_0, \ldots, k_{l-1}$ is l bytes (typically $5 \leq l \leq 32$), then the expanded key K is constructed as $K[i] = k_{i \bmod l}$ for $0 \leq i \leq N-1$. The initial permutation produced by the KSA acts as an input to the next procedure PRGA that generates the keystream. The RC4 algorithms KSA and PRGA are depicted in Fig. 1.

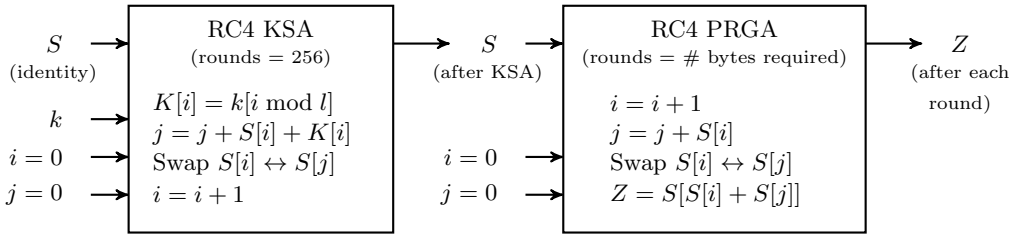

Fig. 1. Description of RC4 stream cipher.

For round $r = 1, 2, \ldots$ of RC4 PRGA, we denote the indices by i_r, j_r, the keystream output byte by Z_r, the output byte-extraction index as $t_r = S_r[i_r] + S_r[j_r]$, and the permutations before and after the swap by S_{r-1} and S_r respectively. After r rounds of KSA, we denote the state variables by adding a superscript K to each variable. All additions (subtractions) in context of RC4 are to be considered as 'addition (subtraction) modulo N', and all equalities in context of RC4 are to be considered as 'congruent modulo N'.

1.1 Description of WPA

IEEE 802.11 standard protocol for WiFi security used to be Wired Equivalent Privacy (WEP), which has now been replaced by Wi-Fi Protected Access (WPA). Both WEP and WPA use RC4 as their core module. In case of WEP, the protocol uses RC4 with a pre-shared key appended to a public initialization vector (nonce) for self-synchronization. Using the technique of related key attacks on RC4, this scheme has been broken through passive full-key recovery attacks, and thus WEP is considered insecure in practice.

To mitigate this problem, WEP has been replaced by WPA. The goal of WPA was to resolve all security threats of WEP. However, the original WEP protocol was extensively adopted by the industry, and it was already implemented in several commercial products, both in software and hardware. This rendered a design of WPA from scratch quite impractical and costly. The work-around was to fix the full-key recovery problems of WEP using a patch, as minimal as possible, on top of the original protocol.

The WPA protocol can be thought of as a wrapper on top of WEP to provide good key mixing features. WPA introduces a key hashing module in the original WEP design to defend against the Fluhrer, Mantin and Shamir attack [2]. It also includes a message integrity feature and a key management scheme to avoid key reuse in the protocol.

TKIP key schedule. WPA uses a 16-byte secret key for RC4 PRNG, the core encryption module of the system. This RC4 secret key is generated through a key schedule procedure known as TKIP [4], which takes as input a 128-bit *temporal key* TK (shared between the parties), transmitter's 48-bit MAC address TA and a 48-bit *initialization vector* IV, and passes those through two phases to obtain the final RC4 secret key.

In Phase 1, a 80-bit key P1K is generated from TK, TA and IV32, the upper 32 bits of the IV, using an unbalanced Feistel cipher with 80-bit block and 128-bit key structure. In Phase 2, the 128-bit RC4KEY is generated from TK, P1K (from Phase 1) and IV16, the lower 16 bits of the IV. In this phase, TK and P1K are mixed (using a temporary key PPK) to construct the last 104 bits (13 bytes) of the RC4KEY, and the first 24 bits (3 bytes) of the RC4KEY are constructed directly from the IV16, as follows [4, Annex H.1].

```
RC4KEY[0] = Hi8(IV16);                    /* RC4KEY[0..2] is the WEP IV */
RC4KEY[1] = (Hi8(IV16) | 0x20) & 0x7F;    /* Help avoid FMS weak keys */
RC4KEY[2] = Lo8(IV16);
```

In the above expression, Hi8(IV16) and Lo8(IV16) indicate the top and lower bytes of IV16, respectively. RC4KEY[0] and RC4KEY[2] are simply two parts of the counter IV16, while RC4KEY[1] is purposefully constructed to avoid the known WEP attack by Fluhrer, Mantin and Shamir [2]. Once the 128-byte (16-byte) RC4KEY is prepared, it is directly used for encryption in the RC4 PRNG core of the protocol.

1.2 Existing biases in WPA keystream

Note that the best bias in RC4 keystream to date is the one involving the second byte (for event $Z_2 = 0$), identified and proved by Mantin and Shamir [10] in 2001. Using this bias, one requires roughly $N = 2^8$ bytes to distinguish the keystream generated by RC4 from a truly random sequence of bytes. This bias is effective even if the secret key of RC4 is truly random, and thus it prevails in WPA as well. This produces a natural $O(N)$ distinguisher of WPA keystream from truly random sequence of bytes. However, the second-byte distinguisher of [10] fails to distinguish between WPA and a generic RC4-based protocol, if the bias is prominent in both cases.

Although various security analysis of WPA are available in the literature, mostly targeted towards key-recovery of WPA using vulnerabilities of TKIP key schedule, the first distinguisher of WPA was proposed quite recently (during 2011-12) by Sepehrdad, Vaudenay and Vuagnoux [14, 15]. The distinguisher of [15], first presented in EUROCRYPT 2011, achieves a 0.5 probability of success in distinguishing WPA with time complexity 2^{43} and packet complexity 2^{40}. Later in [14], the distinguisher was improved to achieve 0.5 probability of success in distinguishing WPA with time complexity 2^{42} and packet complexity 2^{42}.

1.3 Existing broadcast attacks on RC4

In this section, we review the existing broadcast attacks on RC4. The assumption is that the same plaintext message is sent to multiple recipients after being encrypted with different keys. Let M be a plaintext, and let $C^{(1)}, C^{(2)}, \ldots, C^{(n)}$ be the RC4 encryptions of M under n uniformly distributed keys for n different users or recipients. Let $C_r^{(u)}$ be the r-th byte of the ciphertext $C^{(u)}$, $u = 1, 2, \ldots, n$.

Mantin and Shamir attack. The first broadcast attack on RC4 appeared in [10]. It used the bias of the second byte Z_2 towards 0. In [10], it was proved that $P(Z_2 = 0) = 2/N$, which is twice than the random case of $1/N$. They also showed the following result.

Proposition 1. *Suppose an event e happens in distribution X with probability p and in 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 constant probability of success.*

Hence, if the attacker collects $\Omega(N)$ number of ciphertexts corresponding to the same plaintext M, encrypted under different keys for different recipients, then the attacker can easily deduce the second byte of M, by the most frequent value of the second byte of the ciphertexts.

MPS attack. In [7,12], Maitra, Paul and Sen Gupta have shown that each of the bytes Z_3 to Z_{255} is biased to 0 with probability $P(Z_r = 0) = 1/N + c_r/N^2$, where $0.242811 \leq c_r \leq 1.337057$, for $1 \leq r \leq 255$. While these results were indeed useful, the way they have mounted the broadcast attack was not supposed to be successful as pointed out in [1].

Isobe et al. attack. Isobe et al. [5] proposed a full plaintext recovery attack on RC4 from only 2^{34} ciphertexts. They discovered some new biases in the keystream, e.g., bias of $Z_1 = 0 | Z_2 = 0$, bias of $Z_3 = 131$, bias of $Z_r = 3$ for $3 \leq r \leq 255$ and bias of $Z_{xl} = -l$, $x = 1, 2, 3, \ldots$. They also experimentally found two new biases in Z_{256} and Z_{257}. For each position r, Isobe et al. proposes to use the strongest known bias involving Z_r, $1 \leq r \leq 257$, to mount a plaintext recovery attack. They consider two candidates for each r, namely, the one with the maximum frequency and the one with the second highest frequency. To recover plaintext bytes M_{258} and beyond, they use the $ABTAB$ bias of [9]. Assuming this bias, they show that

$$(C_r^{(u)} || C_{r+1}^{(u)}) \oplus (C_{r+2+G}^{(u)} || C_{r+3+G}^{(u)}) = (P_r || P_{r+1}) \oplus (P_{r+2+G} || P_{r+3+G})$$

and use this result sequentially with $G = 0, 1, \ldots, G_{MAX} = 63$ to recover successive plaintext bytes.

AlFardan et al. attack. In [1], AlFardan et al. argue that if there are multiple biases of approximately the same size, then the majority voting may lead to an error. To mitigate this, they propose to take into account all possible single-byte RC4 biases at the same time, along with their strengths. First, they empirically estimate $p_{r,v} = P(Z_r = v)$, for $0 \leq v \leq 255$ and all $r \geq 1$. Then for each possible value m of M_r, $0 \leq m \leq 255$, one calculates a frequency distribution of Z_r from the values of $C_r^{(u)} \oplus m$, $1 \leq u \leq n$, with the frequencies, say, $n_{m,0}, n_{m,1}, \ldots, n_{m,255}$. Then the probability that the plaintext byte M_r was m is given by

$$\rho_{r,m} = \frac{n!}{n_{m,0}! n_{m,1}! \ldots n_{m,255}!} \prod_{v=0}^{255} (p_{r,v})^{n_{m,v}}.$$

The value m with the highest probability is taken to be M_r. For a different r and m, the new frequencies are a permutation of the values $n_{m,0}, n_{m,1}, \ldots, n_{m,255}$. Hence, when comparing different $\rho_{r,m}$'s, one may ignore the coefficient before the $\prod$ sign. Moreover, the computation becomes easier, if logarithms of the product are considered, i.e., the quantity that is to be maximized is $\sum_{v=0}^{255} n_{m,v} \log p_{r,v}$. Next, for full plaintext recovery at any position, the authors use the same technique on Fluhrer-McGrew type double byte biases [3] of overlapping plaintext pairs.

1.4 Contribution of this paper

We have two important motivations in this work. The first one is to provide theoretical justification of some of the experimental observations on WPA by AlFardan, Bernstein, Paterson, Poettering and Schuldt [1]. Theoretical analysis provides further insight to such problems and also it discovers some related results that may lead to some further weaknesses. We derive the complete distribution of the first keystream byte, when RC4 is executed with IV's as in WPA. This gives a method to distinguish the keystream of WPA from that of a generic instance of RC4 with a packet complexity of approximately 2^{19}. In the same direction, we also show how the initial keystream bytes are biased to zero again in a zig-zag manner.

The next one is find out correlations between the keystream bytes and some known quantity that are better than what explained in [1]. The biases presented in [1] are correlated to absolute values between $[0, 255]$ and in most of the cases the additional biases are of the order of $\frac{1}{N^2}$ over the random association $\frac{1}{N}$. We note that there are quite a few much more significant biases of the keystream bytes with the initial key bytes of RC4. The first three key bytes of RC4 are IV bytes in WPA and thus the correlation with any combination of those key bytes can be successfully exploited in broadcast attack. In particular, there exist biases in RC4 keystream bytes Z_1, Z_2, Z_3, Z_{256}, Z_{257} towards the first three bytes (i.e., the IV bytes in WPA) of the secret key. These biases (with slightly different probabilities) hold in WPA also. This approach reduces the number of required ciphertexts significantly compared to the existing works.

2 Biases in WPA resulting from TKIP

Equation (1) summarizes the construction of the first three bytes of the RC4 secret key in WPA/TKIP.

$$K[0] = (\mathsf{IV16} >> 8)\ \&\ \mathsf{0xFF} \qquad K[1] = ((\mathsf{IV16} >> 8)\ |\ \mathsf{0x20})\ \&\ \mathsf{0x7F} \qquad K[2] = \mathsf{IV16}\ \&\ \mathsf{0xFF} \qquad (1)$$

Note that only a 16-bit (2-byte) IV16 is expanded to the initial 3 bytes of the key, and the first two bytes $K[0]$ and $K[1]$ have quite a few bits in common. Specifically, the expansion of IV16 is as shown in Fig. 2, and one may note that 6 bits are shared by $K[0]$ and $K[1]$, apart from the two fixed bits in $K[1]$. The third key-byte, $K[2]$ however, is independent of the first two bytes of the key. Thus, TKIP can generate only 2^{16}, and not 2^{24}, distinct values of the first 3 bytes of the RC4 secret key – a loss in entropy that we believe may result into some non-random behavior in the initial phases of the cipher.

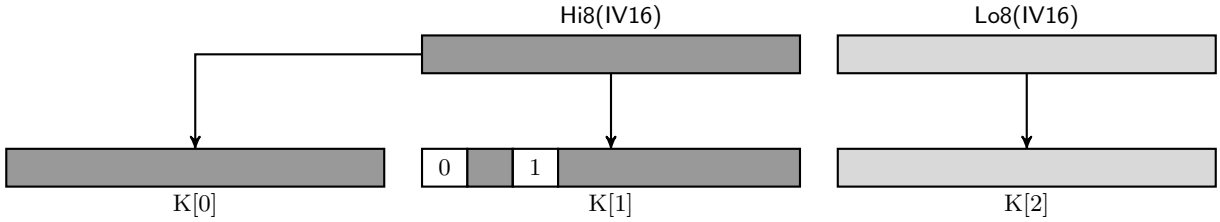

Fig. 2. Expansion of WPA IV16 into the first three bytes of the RC4 secret key.

2.1 Bias in $K[0] + K[1]$ for WPA/TKIP

As $K[0]$ and $K[1]$ share 6 bits from the common source Hi8(IV16), we first take a look at their sum, $K[0]+K[1]$, for potential non-randomness. We notice the following pattern in this direction.

1. The value of $K[0] + K[1]$ must always be *even*, as $K[0]$ and $K[1]$ have the same LSB.
2. The value of $K[1]$ can never exceed 127 as the MSB is 0. The value can not attain all possible numbers below 127 either, as the 6-th bit (from LSB side) is fixed at 1.
3. Value of $K[1]$ and hence $K[0] + K[1]$ strictly depend on the value and range of $K[0]$.

The above restrictions result in corresponding conditions on the range of $K[1]$ and $K[0]+K[1]$, depending on the range of $K[0]$. The complete set of conditions on the respective ranges is shown in Table 1, which results in a consolidated probability distribution of $K[0] + K[1]$, as follows.

Theorem 1. *The probability distribution of the sum of first two bytes of the RC4 key generated by TKIP key schedule in WPA, i.e., the distribution of* $\Pr(K[0] + K[1] = v)$ *for* $v = 0, 1, \ldots, 255$*, is as in Table 1:*

$$\begin{aligned}
&\Pr(K[0] + K[1] = v) = 0 && \text{if } v \text{ is odd;}\\
&\Pr(K[0] + K[1] = v) = 0 && \text{if } v \text{ is even and } v \in [0, 31] \cup [128, 159];\\
&\Pr(K[0] + K[1] = v) = 2/256 && \text{if } v \text{ is even and } v \in [32, 63] \cup [96, 127] \cup [160, 191] \cup [224, 255];\\
&\Pr(K[0] + K[1] = v) = 4/256 && \text{if } v \text{ is even and } v \in [64, 95] \cup [192, 223].
\end{aligned}$$

Proof. The value of $K[0] + K[1]$ is always even, as discussed earlier. The value and range of $K[1]$, and hence that of $K[0] + K[1]$, depends on the range of $K[0]$; shown in Table 1. The probability distribution of $K[0] + K[1]$ may be calculated directly from this dependence pattern; also shown in Table 1. One may check

$$\underbrace{(128 \times 0)}_{\text{odd values}} + \left(16 \times 0 + 16 \times \frac{2}{256} + 16 \times \frac{4}{256} + 16 \times \frac{2}{256} + 16 \times 0 + 16 \times \frac{2}{256} + 16 \times \frac{4}{256} + 16 \times \frac{2}{256}\right) = 1,$$

to validate the consistency of the probability distribution of $K[0] + K[1]$, as depicted in Table 1. □

Table 1. Probability distribution of $K[0] + K[1]$ resulting due to TKIP key scheduling in WPA.

$K[0]$	$K[1]$ (depends on $K[0]$)		$K[0] + K[1]$ (only even)	
Range	Value	Range	Value	Range
0 – 31	$K[0] + 32$	32 – 63	$2K[0] + 32$	32 – 95
32 – 63	$K[0]$	32 – 63	$2K[0]$	64 – 127
64 – 95	$K[0] + 32$	96 – 127	$2K[0] + 32$	160 – 223
96 – 127	$K[0]$	96 – 127	$2K[0]$	192 – 255
128 – 159	$K[0] - 96$	32 – 63	$2K[0] - 96$	160 – 233
160 – 191	$K[0] - 128$	32 – 63	$2K[0] - 128$	192 – 255
192 – 223	$K[0] - 96$	96 – 127	$2K[0] - 96$	32 – 95
224 – 255	$K[0] - 128$	96 – 127	$2K[0] - 128$	64 – 127

$K[0] + K[1]$ (only even)	Probability (0 for odd)
0 – 31	0
32 – 63	2/256
64 – 95	4/256
96 – 127	2/256
128 – 159	0
160 – 191	2/256
192 – 223	4/256
224 – 255	2/256

2.2 Bias in RC4 PRGA initial permutation S_0 for WPA/TKIP

In 2008, Maitra and Paul [6] proved the famous Roos' biases [11], which states that the initial bytes of the permutation S_0, right after the completion of RC4 KSA, are biased towards certain combination of secret key bytes. We get $S_0[0]$ biased towards $K[0]$, which is uniformly distributed, identical to the lower half of the counter IV16. For $S_0[1]$ however, we get the following result.

Theorem 2. *In WPA/TKIP, the probability distribution of the second location of the RC4 permutation* S_0 *generated after KSA, i.e., the distribution of* $\Pr(S_0[1] = v)$ *for* $v = 0, 1, \ldots, 255$*, is given as*

$$\Pr(S_0[1] = v) = \alpha \cdot \Pr(K[0] + K[1] = v - 1) + (1 - \alpha) \cdot (1/N),$$

where $\alpha = \frac{1}{N} + \left(1 - \frac{1}{N}\right)^{N+2}$*, and the term* $\Pr(K[0] + K[1] = v - 1)$ *can be computed using Theorem 1.*

Proof. From the proof of Roos' biases in [6], the initial permutation byte $S_0[y]$ of RC4 is biased towards $f_y = \sum_{x=0}^{y} K[x] + y(y+1)/2$. In particular, for $y = 1$, we get $S_0[1]$ biased towards $K[0]+K[1]+1$, as follows:

$$\Pr(S_0[1] = K[0] + K[1] + 1) \;\approx\; \frac{1}{N} + \left(1 - \frac{1}{N}\right)^{N+2} \;=\; \alpha, \quad \text{say.}$$

Thus, the probability distribution of $S_0[1]$ in case of WPA/TKIP is given as

$$\begin{aligned}\Pr(S_0[1] = v) &= \Pr(S_0[1] = v \ \wedge \ S_0[1] = K[0] + K[1] + 1) + \Pr(S_0[1] = v \ \wedge \ S_0[1] \neq K[0] + K[1] + 1) \\ &= \Pr(S_0[1] = K[0] + K[1] + 1) \cdot \Pr(K[0] + K[1] + 1 = v) \\ &\qquad + \Pr(S_0[1] \neq K[0] + K[1] + 1) \cdot \Pr(S_0[1] = v) \\ &\approx \alpha \cdot \Pr(K[0] + K[1] = v - 1) + (1 - \alpha) \cdot (1/N),\end{aligned}$$

where we have assumed that $S_0[1] = K[0] + K[1] + 1$ and $K[0] + K[1] + 1 = v$ are mutually independent, and that $S_0[1] = v$ occurs with random probability of association $1/N$ in case $S_0[1] \neq K[0] + K[1] + 1$. □

While computing for $N = 256$, as in WPA and RC4, $\alpha \approx 0.368$ in Theorem 2, and we have:

$$\Pr(S_0[1] = v) = 0.368 \times \Pr(K[0] + K[1] = v - 1) + 0.00246875.$$

The values of $\Pr(K[0] + K[1] = v - 1)$ in case of WPA/TKIP may be taken from Theorem 1, while in case of generic RC4, the distribution $K[0] + K[1] = v - 1$ may be assumed to be uniform as the key bytes $K[0]$ and $K[1]$ are chosen at random. This produces two different distributions of $S_0[1]$, one for generic RC4, and another for WPA/TKIP, as shown in Fig. 3.

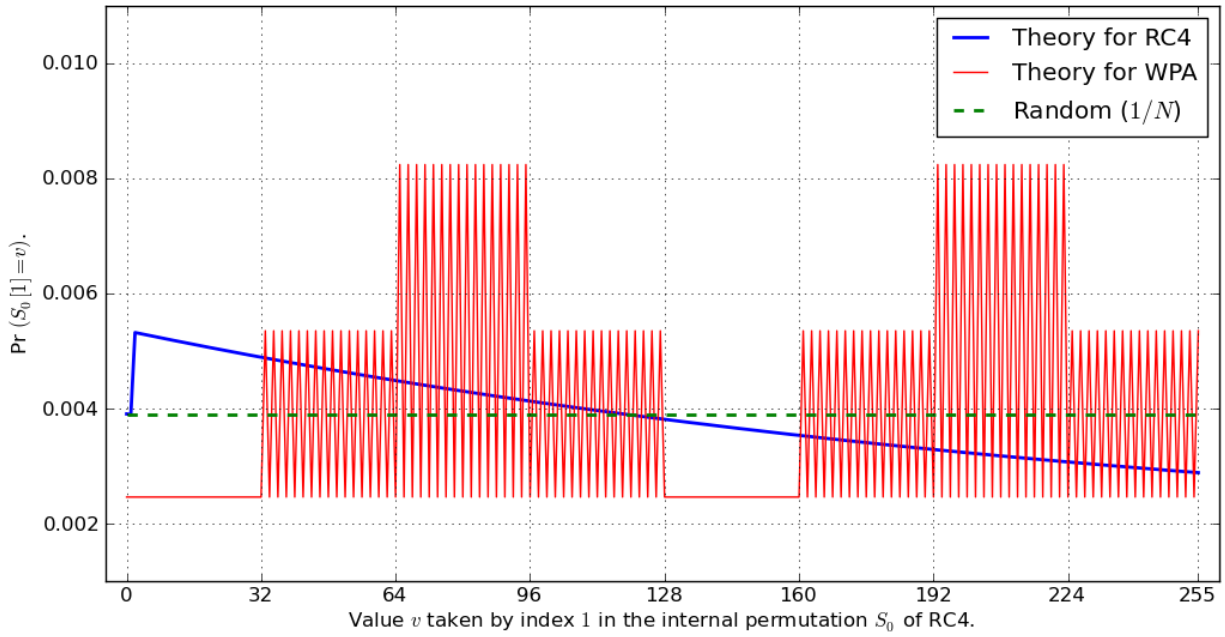

Fig. 3. Theoretical plot for $\Pr(S_0[1] = v)$ for RC4 and WPA, where $v = 0, \ldots, 255$.

2.3 Bias in the first keystream byte Z_1 of WPA/TKIP

Recall that in the first round of RC4 PRGA, the initial permutation entry $S_0[1]$ is crucial; it serves as $j_1 = S_0[i_1] = S_0[1]$, and plays an important role in determining the first keystream byte

$$Z_1 = S_1[S_1[i_1] + S_1[j_1]] = S_1[S_0[j_1] + S_0[i_1]] = S_1[S_0[S_0[1]] + S_0[1]].$$

In fact, we know that $S_0[1]$ is a vital component in the closed-form expression for Z_1, as proved by Sen Gupta, Maitra, Paul and Sarkar [13]. We reproduce the expression for Z_1 from [13, Theorem 13] as follows.

Proposition 2 (from [13]). *The probability distribution of the first output byte of RC4 keystream is as follows, where* $v \in \{0, \ldots, N-1\}$, $\mathcal{L}_v = \{0, 1, \ldots, N-1\} \setminus \{1, v\}$ *and* $\mathcal{T}_{v,X} = \{0, 1, \ldots, N-1\} \setminus \{0, X, 1-X, v\}$.

$$\Pr(Z_1 = v) = Q_v + \sum_{X \in \mathcal{L}_v} \sum_{Y \in \mathcal{T}_{v,X}} \Pr(S_0[1] = X \ \wedge \ S_0[X] = Y \ \wedge \ S_0[X+Y] = v),$$

$$\textit{with} \quad Q_v = \begin{cases} \Pr(S_0[1] = 1 \ \wedge \ S_0[2] = 0), & \textit{if } v = 0; \\ \Pr(S_0[1] = 0 \ \wedge \ S_0[0] = 1), & \textit{if } v = 1; \\ \Pr(S_0[1] = 1 \ \wedge \ S_0[2] = v) + \Pr(S_0[1] = v \ \wedge \ S_0[v] = 0) & \\ \quad + \Pr(S_0[1] = 1 - v \ \wedge \ S_0[1 - v] = v), & \textit{otherwise.} \end{cases}$$

We consider two cases while computing the numeric values of $\Pr(Z_1 = v)$. If the initial permutation S_0 of RC4 PRGA is constructed from the regular KSA with random key, the probabilities $\Pr(S_0[u] = v)$ closely follow the distribution proved by Mantin in [8, Theorem 6.2.1]. However, if the initial permutation S_0 originates from RC4 KSA using TKIP-generated keys, as in the case with WPA, then $\Pr(S_0[1] = v)$ must be computed using Theorem 2, including its idiosyncratic biases for WPA/TKIP shown in Fig. 3.

We compute the exact probabilities $\Pr(Z_1 = v)$ for generic RC4 and WPA/TKIP using the estimation strategy of joint probabilities proposed in [13], where the distribution of $S_0[1] = v$ is considered independently in each case. This results in two different distributions of Z_1; one for generic RC4 and the other for RC4 used with TKIP, as in WPA. Figure 4 displays the two distributions, clearly pointing out the bias resulting in the PRGA as a result of TKIP key schedule.

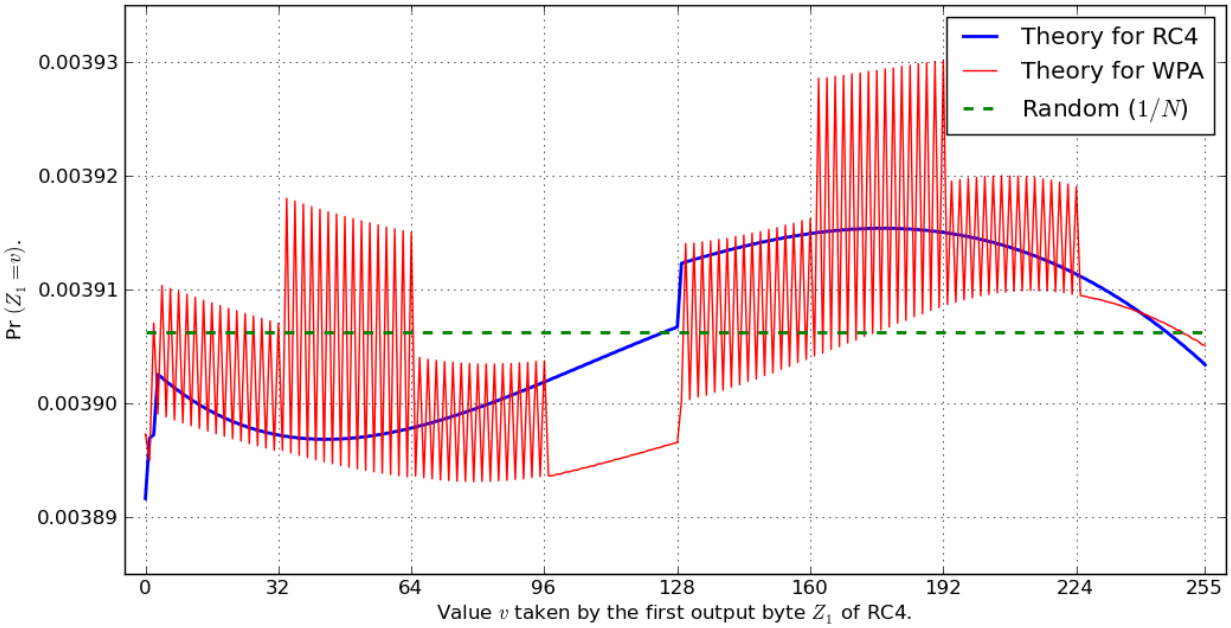

Fig. 4. Theoretical plot for $\Pr(Z_1 = v)$ for RC4 and WPA, where $v = 0, \ldots, 255$.

Note that the patterns of these two theoretical distributions closely match the recent experimental observations of AlFardan, Bernstein, Paterson, Poettering and Schuldt [1] (Fig. 10(a) in the full version of the paper, available online). The only difference is that there exist keylength dependent spikes at $Z_1 = 129$ for the observations in [1], as the experiments were done using 16-byte keys, whereas in our theoretical analysis, we disregard the keylength dependence altogether, and prove a general distribution of Z_1.

In fact, if WPA had employed RC4 with full-length 256-byte secret keys, where the first three bytes of the key $K[0], K[1], K[2]$ were constructed from the IV using TKIP key schedule principle (as in Equation (1)), the pattern of the bias in Z_1 for WPA/TKIP would have been the same. We have independently verified our theoretical results through experiments involving secret keys of various lengths.

Next, we attempt at combining the values of Z_1 in suitable subsets of $\{0, 1, \ldots, 255\}$ to construct a distinguisher between WPA and RC4. The structure of the event considered for distinguishing WPA from RC4 in this case is "$e_S : (Z_1 \in S)$ where $S \subseteq \{0, 1, \ldots, 255\}$".

In this case however, the subset S may be quite large, and thus the probability $\Pr(e_S)$ in either distribution may not be small. In other words, the base probability p is not essentially small in this case, and thus the estimates for distinguisher complexity Proposition 1 [10, Theorem 2] may not work directly. To circumvent this issue, we propose the following result for estimating the complexity of a distinguisher for general p and small q.

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

Proof. This is similar to the proof for [10, Theorem 2], with approximations on p, q reconsidered for general p.

Let X_e and Y_e be the random variables denoting the number of occurrences of event e in the t samples. Then $X_e \sim \mathsf{Bin}(t, p)$ and $Y_e \sim \mathsf{Bin}(t, p(1+q))$ respectively, and their expectations, variances and standard deviations may be computed as:

$$E[X_e] = tp,\ V(X_e) = tp(1-p),\ \sigma(X_e) = \sqrt{tp - tp^2};$$
$$E[Y_e] = tp(1+q),\ V(Y_e) = tp(1+q)(1-p(1+q)),\ \sigma(Y_e) = \sqrt{tp(1+q) - tp^2(1+q)^2} \approx \sqrt{tp - tp^2}.$$

The goal is to analyze the size of t for which the expectations of the two distributions are apart by at least one standard deviation. We get:

$$E[Y_e] - E[X_e] \geq \sigma(X_e) \ \Rightarrow\ tpq \geq \sqrt{tp - tp^2} \ \Rightarrow\ t \geq \frac{1-p}{pq^2}.$$

Consequently, $O(\frac{1-p}{pq^2})$ samples suffice for distinguishing the two distributions, where the constant depends on the desired probability of success. □

Now that we have a decent estimate for the distinguisher complexity, we may define a suitable set S for the target distinguishing event. As most of higher biases are for *even* values of the first byte, we assume that the distributions of WPA and RC4 differ the most in cases when Z_1 takes an even value. Based on this intuition, we pick the set S as the set of all even values $\{0, 2, 4, \ldots, 254\}$ within the range; thus defining the distinguishing event:

$$e_S \ :\ (Z_1 = 2k \text{ for } k = 0, 1, \ldots, 127).$$

From our theoretical results on the distribution of Z_1 in WPA and RC4, as proved in Section 2.3, we estimate the following probabilities:

$$p = \Pr(e_S) \text{ in RC4 } \approx 0.499995,\ p(1+q) = \Pr(e_S) \text{ in WPA } \approx 0.500713 \quad \Rightarrow \quad q \approx 0.001437 \approx 0.37/N.$$

For $N = 256$, as in the case with WPA, we require an estimated $8N^2 = 2^{19}$ keystream packets to distinguish WPA from a generic instance of RC4 with more than 70% probability of success. This is clearly the best distinguisher of WPA to date, improving the previous distinguishers of packet complexity more than 2^{40}, identified by Sepehrdad, Vaudenay and Vuagnoux [14, 15].

2.4 Bias towards zero in keystream bytes $Z_3, \ldots, Z_{255}$ of WPA/TKIP

We extend the effect of the bias in S_0 to the biases in the initial keystream bytes towards zero. Maitra et al. [7] proved the biases of the initial keystream bytes $Z_3, \ldots, Z_{255}$ towards zero, and we reproduce their result from [13, Theorem 14] in Proposition 3, as follows.

Proposition 3 (from [13]). *For PRGA rounds $3 \leq r \leq N-1$, the probability that $Z_r = 0$ is given by:*

$$\Pr(Z_r = 0) \approx \frac{1}{N} + \frac{c_r}{N^2}, \quad \textit{where} \quad c_r = \begin{cases} \frac{N}{N-1}\left(N \cdot \Pr(S_{r-1}[r] = r) - 1\right) - \frac{N-2}{N-1}, & \textit{for } r = 3; \\ \frac{N}{N-1}\left(N \cdot \Pr(S_{r-1}[r] = r) - 1\right), & \textit{otherwise.} \end{cases}$$

In [13], the computation of $\Pr(Z_r = 0)$ depended on the computation of $\Pr(S_{r-1}[r] = r)$, which in turn required the distribution of the initial permutations S_0 and S_1 of RC4 PRGA (refer to [13, Corollary 2] and [13, Lemma 1] for details). We consider two cases – one in which the initial permutation S_0 is generated by generic RC4 KSA using random keys, and the other where S_0 is biased (as discussed earlier) for using RC4 with keys originating from TKIP. These two cases produce two different distributions of $\Pr(Z_r = 0)$ for $r = 3, \ldots, 255$, as depicted in Fig. 5. The patterns closely match the experimental observations of AlFardan, Bernstein, Paterson, Poettering and Schuldt [1] (Fig. 11 in the full version of the paper, available online).

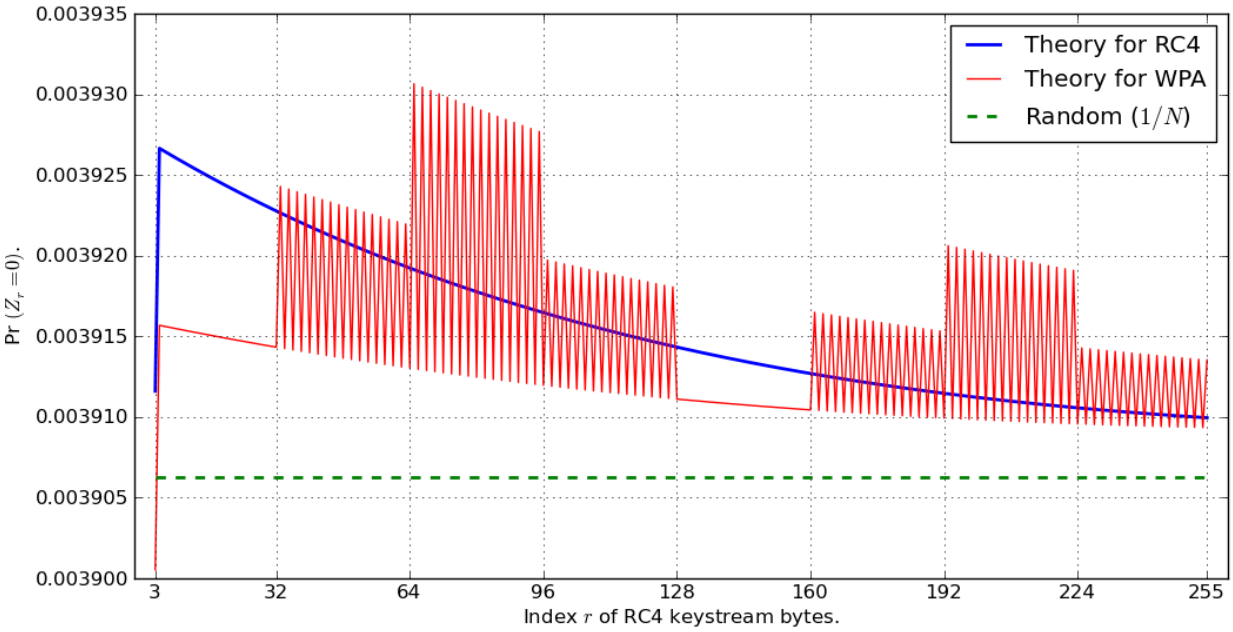

Fig. 5. Theoretical plot for $\Pr(Z_r = 0)$ for RC4 and WPA, where $r = 3, \ldots, 255$.

3 Correlation of keystream bytes with the *IV* in WPA

In WPA, the first three bytes of the RC4 key is used as the IV; thus the IV can be denoted by k_i where $i = 0, 1, 2$. For the u-th recipient, let us denote the IV by $k_i^{(u)}$ where $i = 0, 1, 2$ and $1 \leq u \leq n$. Note that the values of $k_i^{(u)}$ are publicly known, and hence these could be exploited towards plaintext-recovery attacks in case they have prominent correlations with the keystream bytes.

In this section, we will investigate for significant correlations between the keystream output bytes Z_r and certain linear combinations of the IV values $\{k_0, k_1, k_2\}$. Let us assume that the number of such correlations for Z_r with probability significantly higher than $1/N = 1/256$ is bounded by m_r. We shall denote the corresponding linear combinations as $L_{r,q}(k_0, k_1, k_2)$, where $q = 1, 2, \ldots, Q_r$.

Let us have the ciphertext bytes $C_r^{(u)}$ $(u = 1, 2, \ldots, n)$ available corresponding to the various keystream bytes $Z_r^{(u)}$ and fixed message byte M_r in the broadcast scenario. For each user u, we will substitute the $k_i^{(u)}$ values in $L_{r,q}$ to obtain Q_r many votes for $Z_r^{(u)}$, which when XOR-ed with $C_r^{(u)}$, give Q_r many votes for M_r. After merging the votes for all users, we consider the byte with the maximum vote as the estimated message $\hat{M}_r$. We describe the method in Algorithm 1.

It is expected that as the number n of users increase, the success probability of the algorithm increases. Note that the algorithm works with any number of linear combinations of the IV's. In fact the linearity is not a necessary conditions. Any linear or non-linear combinations of the IV's that have a strong correlation with the keystream bytes Z_r can be used to count the votes towards M_r.

Input:
1. Byte position r, $r \geq 1$.
2. n independent encryptions $C_r^{(u)}$, $1 \leq u \leq n$, of a fixed plaintext M_r.
3. IV bytes $k_0^{(u)}$, $k_1^{(u)}$ and $k_2^{(u)}$.
4. Q_r linear combinations $L_{r,q}$, $1 \leq q \leq Q_r$, of the IV's.

Output: An estimate $\hat{M}_r$ of the plaintext byte M_r.

for *m from 0 to 255 in steps of 1* **do**
  $N_m \leftarrow 0$;
end
for *u from 1 to n in steps of 1* **do**
  for *q from 1 to Q_r in steps of 1* **do**
    Compute $\hat{Z}_{r,q}^{(u)} \leftarrow L_{r,q}(k_0^{(u)}, k_1^{(u)}, k_2^{(u)})$;
    $m \leftarrow C_r^{(u)} \oplus Z_{r,q}^{(u)}$;
    $N_m \leftarrow N_m + 1$;
  end
end
$\hat{M}_r \leftarrow \underset{0 \leq m \leq 255}{\text{argmax}} \{N_m\}$;

Algorithm 1: Plaintext recovery using key correlations.

There could be many more linear or non-linear combinations of the IV's that might have a correlation with a certain keystream byte. However, we work with only the known correlations. An interesting feature of our Algorithm is that we do not need to assume that any other combination, if it exists, occur with uniformly random probability. As more and more correlations are discovered, that may be added to the pool of biases used in Algorithm 1. In fact, for the same success probability, more correlations help to reduce the number of samples or ciphertexts required. This is formally analyzed in the following result.

Theorem 3. *Let $P(Z_r = L_{r,q}) = p_{r,q}$, $1 \leq q \leq Q_r$. Then the number of samples (users or ciphertexts) required for Algorithm 1 to recover the correct plaintext byte M_r with a false positive rate α and a false negative rate β satisfies*

$$n > \left(\frac{\kappa_1 \sqrt{\sum_{q=1}^{Q_r} p_{r,q}(1-p_{r,q})} + \kappa_2 \sqrt{\sum_{q=1}^{Q_r} \left(\frac{1-p_{r,q}}{255}\right)\left(1-\frac{1-p_{r,q}}{255}\right)}}{P_r - \frac{Q_r - P_r}{255}} \right)^2,$$

where $P_r = \sum_{q=1}^{Q_r} p_{r,q}$, $\Phi(-\kappa_1) = \alpha$ and $\Phi(\kappa_2) = 1 - \beta$, $\Phi(.)$ being the standard normal cumulative distribution function.

Proof. Out of n samples from $L_{r,q}$ with a fixed q, $1 \leq q \leq Q_r$, let $X_{r,q}$ be the number of samples that give the correct value of Z_r. Hence, the number of samples that give the wrong value of Z_r is given by $n - X_{r,q}$. We assume that in the wrong votes of M_r from the fixed $L_{r,q}$, each of the 255 wrong values appear equally likely. Let $Y_{r,q}$ denote the number of samples with *any* specific wrong value. Both $X_{r,q}$ and $Y_{r,q}$ follow normal distributions with means $E[X_{r,q}] = np_{r,q}$ and $E[Y_{r,q}] = n(\frac{1-p_{r,q}}{255})$, and variances $V[X_{r,q}] = np_{r,q}(1-p_{r,q})$ and $V[Y_{r,q}] = n(\frac{1-p_{r,q}}{255})(1-\frac{1-p_{r,q}}{255})$.

Adding the votes from each $L_{r,q}$, $1 \leq q \leq Q_r$, we have the total number of correct votes as $X_r = X_{r,1} + \ldots + X_{r,Q_r}$ and the total number of votes for *any* specific wrong value (amongst 255 possible wrong values) as $Y_r = Y_{r,1} + \ldots + Y_{r,Q_r}$. Now, we use the fact the sum of multiple normal distributions also follow

a normal distribution. By linearity of expectation,

$$E[X_r] = \sum_{q=1}^{Q_r} E[X_{r,q}] = n \sum_{q=1}^{Q_r} p_{r,q} = nP_r, \tag{2}$$

$$E[Y_r] = \sum_{q=1}^{Q_r} E[Y_{r,q}] = n \sum_{q=1}^{Q_r} \left(\frac{1 - p_{r,q}}{255} \right) = \frac{n}{255}(Q_r - P_r). \tag{3}$$

We assume that each of the Q_r correlations are statistically independent. Hence, the variances of X_r and Y_r are given by

$$\sigma^2_{X_r} = \sum_{q=1}^{Q_r} V[X_{r,q}] = n \sum_{q=1}^{Q_r} p_{r,q}(1 - p_{r,q}), \tag{4}$$

$$\sigma^2_{Y_r} = \sum_{q=1}^{Q_r} V[Y_{r,q}] = n \sum_{q=1}^{Q_r} \left(\frac{1 - p_{r,q}}{255} \right) \left(1 - \frac{1 - p_{r,q}}{255} \right). \tag{5}$$

Here, we want to distinguish the two distributions, namely that of X_r (corresponding to the event $Z_r \oplus L_{r,q} = M_r$) and that of Y_r (corresponding to the event $Z_r \oplus L_{r,q} = w$, where w is a fixed value different from M_r). We require

$$E[X_r] - E[Y_r] > \kappa_1 \sigma_{X_r} + \kappa_2 \sigma_{Y_r}.$$

Substituting the values from Equations (2), (3), (4), and (5), and solving for n, we get the result. □

Corollary 1. *For 69.15% success probability of Algorithm 1, the number of ciphertexts required satisfies*

$$n > \frac{\left(\sqrt{\sum_{q=1}^{Q_r} p_{r,q}(1 - p_{r,q})} + \sqrt{\sum_{q=1}^{Q_r} \left(\frac{1 - p_{r,q}}{255} \right) \left(1 - \frac{1 - p_{r,q}}{255} \right)} \right)^2}{4 \left(P_r - \frac{Q_r - P_r}{255} \right)^2}.$$

Proof. For $\alpha = \beta = 1 - 0.6915$, we need $\kappa_1 = \kappa_2 = 0.5$. Substituting in Theorem 3, we get the result. □

Note that if the equations $L_{r,q}$ for a given r are statistically dependent, then Equation (4) would be modified as

$$\sigma^2_{X_r} = \sum_{q=1}^{Q_r} V[X_{r,q}] + 2 \sum_{1 \le i < j \le Q_r} Cov(X_{r,i}, X_{r,j}), \text{ where}$$

$$Cov(X_{r,i}, X_{r,j}) = E[X_{r,i}, X_{r,j}] - E[X_{r,i}]E[X_{r,j}].$$

Here, $E[X_{r,i}] = p_{r,i}$ and $E_[X_{r,j}] = p_{r,j}$ and $E[X_{r,i}, X_{r,j} = p_{r,(i,j)}$, where $p_{r,(i,j)}$ denotes the probability that both the equations $L_{r,i}$ and $L_{r,j}$ are correct[4].

3.1 Obtaining the plaintext bytes M_1, M_2, M_3, M_{256}, M_{257}

In Table 2, we list the known biases in RC4. The references for these biases can be found in [14, Fig. 4.9]. We also report the corresponding biases when RC4 is executed with key setup as in WPA[5]. Note that any random association between two bytes is $\frac{1}{256} = 0.003906$.

Let us consider one example with Z_1 in RC4. We have $Q_1 = 3$, $p_{1,1} = 0.005304$, $p_{1,2} = 0.004367$, $p_{1,3} = 0.005214$. Hence, $P_1 = p_{1,1} + p_{1,2} + p_{1,3} = 0.014885$. According to Corollary 1, we need $1306 < 2^{11}$ first ciphertext-bytes to recover the first plaintext byte M_1 with 69.15% success rate.

[4] We are currently deriving these joint probabilities.

[5] The exact proofs in case of WPA would appear in the complete version of this paper.

Table 2. Linear keystream-key correlations in RC4 and WPA

Keystream byte	Linear combinations	Probability in RC4	Probability in WPA
Z_1	$L_{1,1} = -k_0 - k_1$	0.005304	0.005300
	$L_{1,2} = k_0$	0.004367	0.004200
	$L_{1,3} = k_0 + k_1 + k_2 + 3$	0.005214	0.004700
Z_2	$L_{2,1} = -1 - k_0 - k_1 - k_2$	0.005317	0.005288
	$L_{2,2} = -3 - k_1 - k_2$	0.005348	0.005300
	$L_{2,3} = k_1 + k_2 + 3$	0.005341	0.005300
Z_3	$L_{3,1} = k_0 + k_1 + k_2 + 3$	0.004436	0.004400
Z_{256}	$L_{256,1} = -k_0$	0.004450	0.004380
Z_{257}	$L_{257,1} = -k_0 - k_1$	0.004115	0.004080

Why it is better than the absolute bias scenario? The existing cases where Z_i is biased to a specific value, then only the maximum bias or the second maximum etc. can be used separately, but they cannot be added up. Further, in WPA or TLS scenario the biases are of the order of $\frac{1}{N^2}$, which is much lower. In Isobe et al. [5] Z_1 was obtained conditional on Z_2. Here it can be obtained directly. Note that once the plaintext bytes Z_1, Z_2, Z_3, Z_{256}, Z_{257} are discovered, we can mount *full plaintext recovery* with less number of keystream bytes than that of [1, 5]. We are currently working on this.

3.2 New key correlations in WPA

To strengthen the set of biases applicable towards a plaintext recovery attack against WPA, we have started investigating key correlations in WPA of the general linear form ($Z_r = a \cdot k_0 + b \cdot k_1 + c \cdot k_2 + d$) where $a, b, c \in \{-1, 0, 1\}$ and $d \in \{-3, -2, -1, 0, 1, 2, 3\}$. As the first three bytes of the key, k_0, k_1, k_2, are known parameters, these biases may be added to the set of known biases for Z_r, and this may potentially result in a much stronger plaintext recovery attack on WPA.

During the course of our investigation, we have already found some correlations in the tune of $O(1/N^2)$. For example, Z_7 is biased towards $k_0 + k_1 + 1$ in WPA with a probability of $1/N + 0.5/N^2$ where $N = 256$. We strongly believe that there are several such biases of the keystream bytes with the IV of WPA, and these would produce a strong plaintext recovery attack. Detailed formulation of this attack and investigation for more such correlations is in progress.

4 Conclusion

Here we study several non-randomness results on RC4 when used in WPA. We analyze several biases of RC4 and also note how they get evolved in WPA as the initial three key bytes are used as IVs. We prove the interesting zig-zag distribution of the first byte and the similar nature for the biases in the initial keystream bytes to zero as pointed out in [1]. We also revisit correlation of certain keystream bytes to the first three IV bytes and they provide much higher biases than what had been presented in [1].

Our results complement the existing works in understanding the reason of the biases and also in adding new biases in the broadcast attack scenario. We are currently investigating for new correlations between the keystream bytes and the IV of WPA. We are also working out the details of a stronger plaintext recovery attack on WPA using such correlations.

References

1. Nadhem AlFardan, Dan Bernstein, Kenny Paterson, Bertram Poettering, and Jacob Schuldt. On the security of RC4 in TLS. In *USENIX Security Symposium*, 2013. Presented at FSE 2013 as an invited talk by Dan Bernstein. Full version of the research paper and relevant results are available online at `http://www.isg.rhul.ac.uk/tls/`.
2. Scott R. Fluhrer, Itsik Mantin, and Adi Shamir. Weaknesses in the key scheduling algorithm of RC4. In Serge Vaudenay and Amr M. Youssef, editors, *Selected Areas in Cryptography*, volume 2259 of *Lecture Notes in Computer Science*, pages 1–24. Springer, 2001.

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. IEEE Computer Society. 802.11i$^{\text{TM}}$ – IEEE standard for information technology – telecommunications and information exchange between systems – local and metropolitan area networks –specific requirements, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, Amendment 6: Medium Access Control (MAC) Security Enhancements, July 2004.
5. Takanori Isobe, Toshihiro Ohigashi, Yuhei Watanabe, and Masakatu Morii. Full plaintext recovery attack on broadcast RC4. In *Fast Software Encryption (FSE)*, 2013. To appear.
6. Subhamoy Maitra and Goutam Paul. New form of permutation bias and secret key leakage in keystream bytes of RC4. In Kaisa Nyberg, editor, *FSE*, volume 5086 of *Lecture Notes in Computer Science*, pages 253–269. Springer, 2008.
7. Subhamoy Maitra, Goutam Paul, and Sourav Sen Gupta. Attack on broadcast RC4 revisited. In Antoine Joux, editor, *FSE*, volume 6733 of *Lecture Notes in Computer Science*, pages 199–217. Springer, 2011.
8. Itsik Mantin. Analysis of the stream cipher RC4. Master's thesis, The Weizmann Institute of Science, Israel, 2001. Available online at `http://www.wisdom.weizmann.ac.il/~itsik/RC4/RC4.html`.
9. 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.
10. 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.
11. Andrew Roos. A class of weak keys in the RC4 stream cipher. Two posts in sci.crypt, message-id `43u1eh$1j3@hermes.is.co.za` and `44ebge$llf@hermes.is.co.za`, 1995. Available online at `http://www.impic.org/papers/WeakKeys-report.pdf`.
12. 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.
13. 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*, 2013. To appear. Published online in December 2012. DOI: 10.1007/s00145-012-9138-1.
14. Pouyan Sepehrdad. *Statistical and Algebraic Cryptanalysis of Lightweight and Ultra-Lightweight Symmetric Primitives*. PhD thesis No. 5415, École Polytechnique Fédérale de Lausanne (EPFL), 2012. Available online at `http://lasecwww.epfl.ch/~sepehrdad/Pouyan_Sepehrdad_PhD_Thesis.pdf`.
15. Pouyan Sepehrdad, Serge Vaudenay, and Martin Vuagnoux. Statistical attack on RC4 - distinguishing WPA. In Kenneth G. Paterson, editor, *EUROCRYPT*, volume 6632 of *Lecture Notes in Computer Science*, pages 343–363. Springer, 2011.