Some Results on RC4

Abstract. In this paper, we have given a theoretical analysis of the key scheduling algorithm (KSA) of RC4, where the nonlinear operation is swapping among the permutation bytes. We have come up with explicit formulae for the probabilities with which the psuedo random index j and hence the permutaion bytes are biased to the secret key at any stage before the first $l+1$ rounds of the KSA, where $l =$ keylength. Theoretical proofs of these formulae had been left open since Roos' work (1995)[27]. But in 2008 Gautam Paul and Subhamoy Maitra[35] gave the first theoretical proof of the Roos' bias. In their work the analysis was based on the assumption that the pseudo random index j is uniformly distributed over $\mathbb{Z}_N$ at each round of KSA. In reality, this is actually not the case. The only thing that is truely random in the whole of RC4 are the key bytes. In this paper, we have done the analysis based on the assumption that only the key bytes are distributed uniformly over $\mathbb{Z}_N$. The formulae that we got gives a better approximation of the expermental probabilities than result in [35].

Keywords: Bias, Cryptanalysis, Keystream, Key leakage, Permutation, RC4, Stream cipher.

1 Introduction

RC4 is a synchronous stream cipher. RC4 was designed by Ron Rivest of RSA Security in 1987. While it is officially termed "Rivest Cipher 4", the RC acronym is alternatively understood to stand for "Ron's Code", thus RC4 = Ron's code #4 or Rivest's code # 4 RC4 was initially a trade secret, but in September 1994 a description of it was anonymously posted to the Cypherpunks mailing list which invoked the external analysisis of RC4. The name "RC4" is trademarked, however. RC4 is often referred to as "ARCFOUR" or "ARC4"(meaning Alleged RC4, because RSA has never officially released the algorithm), to avoid possible trademark problems. RC4 is one of the most widely-used software stream cipher and is used in popular protocols such as Secure Sockets Layer (SSL) and Transport Layer Security (TLS) (to protect Internet traffic), Wired Equivalent Privacy (WEP) and Wi-Fi Protected Access (WPA) (to secure wireless networks). It is also used in Microsoft Windows, Lotus Notes, Apple AOCE, Oracle Secure SQL etc. Though a variety of other stream ciphers have been proposed after RC4, it is still the most popular stream cipher algorithm due to its simplicity, ease of implementation, speed and efficiency. The algorithm can be stated in less than ten lines, yet after two decades of analysis its strengths and weaknesses are of great interest to the community.

The RC4 algorithm uses a state array or a S-Box S = $(S[0], \ldots, S[N-1])$ of length N. Typically, N = 256. S is initialized as the identity permutation, i.e., S[y] = y for $0 \le y \le N-1$. A secret key k of size l bytes (typically, $5 \le l \le 16$) is used to scramble this permutation.

The RC4 cipher has two components, namely, the Key Scheduling Algorithm (KSA) and the Pseudo-Random Generation Algorithm (PRGA). The KSA takes as input the secret key k and the state array S and turns into a random permutation S of $0, 1, \ldots, N-1$. The random permutation obtained after the completion of KSA forms the input to the PRGA which then uses this permutation to generate pseudo-random keystream bytes. The keystream output byte z is bitwise XOR-ed with the message byte to generate the ciphertext byte which is then send through the channel. On the other end the receiver has in his disposal a similar synchronised RC4 running wiht the same key k. The receiver upon receiving the cipher text z again bitwise XOR's with the keystream byte generated at his/her end to get back the message byte.

The index i in both the algorithm above is normally termed as the deterministic index of RC4 whereas the index j is termed as the pseudo-random index of RC4.

```
KSA()
Initialization:
    for i from 0 to N-1
        S[i] := i
    endfor
    j := 0
Scrambling:
    for i from 0 to N-1
        j := (j + S[i] + key[i mod l]) mod N
        swap(&S[i],&S[j])
    endfor
```

```
PRGA()
Initialization:
    i := 0
    j := 0
Keystream Generation Loop:
    while GeneratingOutput:
        i := (i + 1) mod N
        j := (j + S[i]) mod N
        swap(&S[i],&S[j])
        output S[(S[i] + S[j]) mod N]
    endwhile
```

2 A Different Probabilistic Approach For The Analysis Of The Key Scheduling Algorithm

2.1 Notations

Let, S_{y+1} be the permutation and j_{y+1} be the value of the pseudo-random index after the $(y+1)^{th}$ round of the RC4 KSA, i.e, the round when the deterministic index i takes the value y, where $y \in \{0, 1, \cdots, N-1\}$. Thus, S_N denotes the final state after the completion of RC4 KSA. We denote the initial state by S_0 and the initial value 0 of j by j_0. Further let, $\mathbb{Z}_N$ denote the set $\{0, 1, \cdots, N-1\}$.

2.2 Our Work

In the literatures that we have went through all the probability analysis that is done on the KSA of RC4 is done based on the assumption that the pseudo-random index j can take all possible N values at each round of the KSA independently, according to the uniform distribution. But in reality this is not the case for RC4. This assumtion has inherent drawbacks as it considers events and hence their probabilities, in cases where those events actually never occur. Thus increasing the search space as one tries to implement such kind of theoretical biases.

In this section we discuss some of the previously known result but without the assumption of the pseudo-random index j being uniformly distributed. Instead we calculate the probabilities based on the fact that the key bytes are uniformly distributed over

$\mathbb{Z}_N = \{0, 1, \cdots, N-1\}$(where we identify the congruence class $\underline{i}$ with i). Since the only thing that is truely random in the whole of RC4 are the key bytes. Which in effect means that one can manipulate upto the l^{th} round of KSA, by choosing different key bytes. After which everything is deterministic. Thus the idea behind this discussion is to come out with some results which are soley dependent on the key bytes and thus giving a better probabilistic analysis of RC4.

Theorem 1. *During KSA,*

1. $S_y[y] = y$ *can only happen if till the* $(y+1)^{th}$ *round the index* j *does not hit* y *and*
2. *if* $S_y[y] \neq y$ *then* $S_y[y] < y$.

Proof. 1. The other possibility is that the index j hits y first, say in the $(\alpha+1)^{th}$ round $\alpha < y$ and after that it is swapped again in an intermediate round, i.e, between $(\alpha+1)^{th}$ round and $(y+1)^{th}$ round. This can happen more that once.

Let, $S_\alpha[\alpha] = \beta$, then after $(\alpha+1)^{th}$ round

$$S_{\alpha+1}[\alpha] = y,\ S_{\alpha+1}[y] = \beta.$$

Now, after the $(\alpha+1)^{th}$ round the index i takes values greater than α. So, to have after any $(\gamma+1)^{th}$ $(\alpha < \gamma < y)$ round $S_{\gamma+1}[y] = y$, j should hit α in the $(\gamma+1)^{th}$. In that case after $(\gamma+1)^{th}$ round

$$S_{\gamma+1}[\alpha] = S_\gamma[\gamma]$$

$$S_{\gamma+1}[\gamma] = y.$$

Now this can happen more than once where after each such step $S_{\gamma+1}[\gamma] = y$.. So the only way $S_y[y]$ may be equal to y is when $\gamma = y-1$ but in that case $S_y[y-1] = y$. Thus, leading to the conclusion that $S_y[y] \neq y$. Hence the theorem.

2. Suppose if possible $S_y[y] = A > y$. Now before $(y+1)^{th}$ round the value of the deterministic index i was $< y$ which implies that there must have been an index $\alpha < y$ such that $S_\alpha[\alpha] = A > y > \alpha$ impling that there exists a $\beta < \alpha < y$ such that $S_\beta[\beta] = A > y > \alpha > \beta$. Now this argument can be continued infinitely many times. Thus, impling that the index i is unbounded below which is a contradiction to the fact that minimum value of i is 0. Hence, the theorem. $\square$

Theorem 2. $P(S_y[y] = y) = \left(\frac{N-1}{N}\right)^y$ $(y \leq l)$.

Proof.

$$P(S_y[y] = y) = P(j_1 \neq y, \cdots, j_y \neq y) \text{ [By Theorem 1]}$$

$$P(j_1 \neq y) = P(K[0] \neq y) = \left(\frac{N-1}{N}\right).$$

Now,

$$j_x = \sum_{m=1}^{x-1} S_m[m] + \sum_{m=0}^{x-1} K[m \, mod \, l] = \sum_{m=1}^{x-1} S_m[m] + \sum_{m=0}^{x-2} K[m \, mod \, l] + K[(x-1) \, mod \, l].$$

Let, $\sum_{m=1}^{x-1} S_m[m] + \sum_{m=0}^{x-2} K[m \, mod \, l] = \alpha_x$.

Then "$j_x \neq y \mid j_1 \neq y, \cdots, j_{x-1} \neq y$" $\Rightarrow K[(x-1) \, mod \, l] \not\equiv (y - \alpha_x) \mod N$. Thus,

$$P(j_x \neq y \mid j_1 \neq y, \cdots, j_{x-1} \neq y) = P\Big(K[(x-1) \, mod \, l] \not\equiv (y - \alpha_x) \mod N\Big) = \Big(\frac{N-1}{N}\Big).$$

Therefore,

$$\begin{aligned} P(S_y[y] = y) &= P(j_1 \neq y, \cdots, j_y \neq y) \\ &= P(j_1 \neq y) \cdot P(j_2 \neq y \mid j_1 \neq y) \cdots P(j_y \mid j_1 \neq y, \cdots, j_{y-1} \neq y) \\ &= P(K[0] \neq y) \cdot P(K[1] \not\equiv (y - \alpha_2) \mod N) \cdots \\ &\quad P(K[y-1] \not\equiv (y - \alpha_y) \mod N) \\ &= \Big(\frac{N-1}{N}\Big) \cdot \Big(\frac{N-1}{N}\Big) \cdots \Big(\frac{N-1}{N}\Big) = \Big(\frac{N-1}{N}\Big)^y. \qquad \square \end{aligned}$$

Theorem 3. $P(S_0[0] = 0; \, S_1[1] = 1; \cdots; S_y[y] = y) = \Big(\frac{(N-1)!}{N^y (N-y-1)!}\Big) \, (y \leq l)$.

Proof.

$P(S_0[0] = 0) = 1$ [Since the initial permutation is the identity permutation in KSA]

Now,

$$\begin{aligned} &P(S_0[0] = 0; \, S_1[1] = 1; \cdots; S_y[y] = y) \\ &= P(; \, j_1 \neq 1; \, j_1 \neq 1, j_2 \neq 2; \cdots; j_1 \neq 1, \cdots, j_y \neq y) \text{ [by Theorem 1]} \\ &= P(j_1 \neq 1, \cdots, j_1 \neq y; \, j_2 \neq 2, \cdots, j_2 \neq y; \cdots; j_y \neq y) \end{aligned}$$

Now,

$$P(j_1 \neq 1, \cdots, j_1 \neq y) = P(k[0] \neq 1, \cdots, y) = \Big(\frac{N-y}{N}\Big).$$

Therefore,

$$\begin{aligned} &P(j_x \neq x, \cdots, j_x \neq y \mid j_1 \neq 1, \cdots, j_1 \neq y; \, j_2 \neq 2, \cdots, j_2 \neq y; \cdots; \\ &\qquad j_{x-1} \neq x-1 \cdots j_{x-1} \neq y) \\ &= P(K[x] \not\equiv x - \alpha_{x+1}, \cdots, y - \alpha_{x+1}) = \Big(\frac{N - (y - (x-1))}{N}\Big) \end{aligned}$$

where α_x has the same meaning as in Theorem 2 and $x \le y$. Then from the chain rule we get the required probability $= \left(\frac{(N-1)!}{N^y (N-y-1)!}\right)$ $\square$

Corollary 1. *The event "$j_{y+1} = \frac{y(y+1)}{2} + \sum_{m=0}^{y} K[m \mod l]$" can occur if and only if the event "$S_0[0] = 0;\ S_1[1] = 1; \cdots ; S_y[y] = y$" occurs, where y is such that $\frac{y(y+1)}{2} \le N$ and $\frac{(y+1)(y+2)}{2} > N$.*

Proof. Let, A_y be the event "$S_0[0] = 0;\ S_1[1] = 1; \cdots ; S_y[y] = y$" and E_y the event that "$j_{y+1} = \frac{y(y+1)}{2} + \sum_{m=0}^{y} K[m \mod l]$". Let, $j'_{y+1} = \sum_{m=1}^{y} S_m[m] + \sum_{m=0}^{y} K[m \mod l]$ where atleast one $S_m[m] \ne m$. Now, the other possibility by which E_y can occur is if,

$$j_{y+1} \equiv j'_{y+1} \mod N$$

$$\Rightarrow \sum_{m=1}^{y} S_m[m] \equiv \frac{y(y+1)}{2} \mod N$$

Now, from the second part of Theorem 1 we get $S_m[m] \le m$, which imples that in j'_y

$$\sum_{m=1}^{y} S_m[m] = \sum_{m=1}^{y_m} (i_m)^{n_m}$$

where $\sum_{m=1}^{y_m} n_m = y$ and $i_m \in \{1, \cdots, y\}\ \forall m$. Therefore,

$$\sum_{m=1}^{y_m} (i_m)^{n_m} + \delta N = \frac{y(y+1)}{2}.$$

where $\delta \in \mathbb{N} \cup \{0\}$. Clearly, if $\frac{y(y+1)}{2} \le N$ then $\delta = 0$ in which case $\sum_{m=1}^{y_m} (i_m)^{n_m} \ne \frac{y(y+1)}{2}$, since for atleast one m, $S_m[m] < m$. $\square$

According to the corollary 1 for $N = 256$ and l upto 22 (since $\frac{22.23}{2} = 253 < 256$ and $\frac{23.24}{2} = 276 > 256$) the event "$j_{y+1} = \frac{y(y+1)}{2} + \sum_{m=0}^{y} K[m \mod l]$" can occur if and only if "$S_0[0] = 0;\ S_1[1] = 1; \cdots ; S_y[y] = y$", where $y \le l$. Which implies that in lemma 1 of [35, section 2.1], $P(E_y \mid \overline{A}_y) = 0$ instead of $1/N$, $\forall\, y \le l \le 22$. Also the probability of A_y in lemma 1 of [35, section 2.1] was calculated assuming that j takes values from $\mathbb{Z}_N$ uniformly at random which is not the case in KSA. Hence, the probability of the same event calculated as in Theorem 3 gives a more exact picture of what actually occurs during RC4 KSA.

Experimental Results : We here substantiate our above claims through experimental results, which actually shows that our claim is indeed true. The following experimental results were obtained by running RC4 KSA on one million randomly selected keys of length 16 bytes.

d	**Experimental**	**Theorem 3**	**Mentioned Paper**	**Diff. col 2, col 3**	**Diff. col 2, col 4**
5	0.942002	0.9426899	0.943204346	0.0006899	0.001202346
6	0.918577	0.920595605	0.921403419	0.002018605	0.002826419
7	0.893766	0.895423069	0.896607697	0.001657069	0.002841697
8	0.865494	0.867441098	0.869089214	0.001947098	0.003595214
9	0.834725	0.836945122	0.839143578	0.002220122	0.004418578
10	0.802510	0.804251953	0.807084699	0.001741953	0.004574699
11	0.768166	0.769694252	0.773239341	0.001528252	0.005073341
12	0.732340	0.733614834	0.737941633	0.001274834	0.005601633
13	0.694950	0.696360956	0.701527645	0.001410956	0.006577645
14	0.657209	0.658278716	0.664330183	0.001069716	0.007121183
15	0.618262	0.619707697	0.626673878	0.001445697	0.008411878
16	0.578868	0.580975966	0.588870698	0.002107966	0.010002698

Thus we can see that that the experimental results also seems to sugest that Theorem 3 actually gives a more accurate probability of the event.

Also, $\frac{(N-l)\times\cdots\times(N-1)}{N^i} > \frac{1}{2}$

$$\Rightarrow \log(N-l) + \cdots + \log(N-1) - l\log(N) > -1$$

$$\Rightarrow l\Big(\log(N-1) - \log(N)\Big) > \log(N-l) + \cdots + \log(N-1) - l\log(N) > -1$$

$$\Rightarrow l\Big(\log(N) - \log(N-1)\Big) < 1$$

$$\Rightarrow l < \frac{1}{\Big(\log(N) - \log(N-1)\Big)}$$

Now, for $N = 256$ and $l <= 177$, i.e, for all key of length < 177 the event "$S_0[0] = 0$; $S_1[1] = 1; \cdots ; S_l[l] = l$" occurs with probability $> \frac{1}{2}$.

Thus we have the following corollary.

Corollary 2. *For $N = 256$ and $l \leq 177$, i.e, for all key of length ≤ 177 the event "$S_0[0] = 0$; $S_1[1] = 1; \cdots ; S_y[y] = y$" ($y \leq l$) occurs with probability $> \frac{1}{2}$.* $\square$

Here we also like to bring to the notice that after the l^{th} round the probability of events like "$S_y[y] = y$ where $y > l$" is actually much more than what we find in the literature. For example, consider the event "$S_{l+1}[l+1] = l+1$". now for this event to occur it is sufficient that "$K[0] \neq l+1, k[0] \not\equiv l+1-j_l - S_l[l]; j_2 \neq l+1; \cdots ; j_l \neq l+1$" event occurs thus giving

$$\begin{aligned} P(S_{l+1}[l+1] = l+1) &= \left(\frac{N-1}{N}\right)^{l}; \text{ if } j_l = S_l[l] = 0 \text{ or } j_l + S_l[l] = N \\ &= \left(\frac{N-2}{N}\right)\left(\frac{N-1}{N}\right)^{l-1}; \text{ otherwise} \end{aligned}$$

which is much more than the $\left(\frac{N-1}{N}\right)^{l+1}$ which is what we get if we consider j to be uniformly distributed over $\mathbb{Z}_N$.

References

1. M. Akgun, P. Kavak and H. Demirci. New Results on the Key Scheduling Algorithm of RC4. Indocrypt 2008.
2. E. Biham and Y. Carmeli. Efficient Reconstruction of RC4 Keys from Internal States. FSE 2008, pages 270-288, vol. 5086, Lecture Notes in Computer Science, Springer Berlin / Heidelberg.
3. eSTREAM, the ECRYPT Stream Cipher Project. `http://www.ecrypt.eu.org/stream` [last accessed on July 18, 2008].
4. S. R. Fluhrer, I. Mantin and A. Shamir. Weaknesses in the Key Scheduling Algorithm of RC4. SAC 2001, pages 1-24, vol. 2259, Lecture Notes in Computer Science, Springer-Verlag.
5. J. Golic. Linear statistical weakness of alleged RC4 keystream generator. EUROCRYPT 1997, pages 226-238, vol. 1233, Lecture Notes in Computer Science, Springer-Verlag.
6. G. Gong, K. C. Gupta, M. Hell and Y. Nawaz. Towards a General RC4-Like Keystream Generator. CISC 2005, pages 162-174, vol. 3822, Lecture Notes in Computer Science, Springer-Verlag.
7. J. Hong and P. Sarkar. New Applications of Time Memory Data Tradeoffs. ASIACRYPT 2005, pages 353-372, vol. 3788, Lecture Notes in Computer Science, Springer-Verlag.
8. R. J. Jenkins. ISAAC and RC4. 1996. Available at `http://burtleburtle.net/bob/rand/isaac.html` [last accessed on July 18, 2008].
9. A. Klein. Attacks on the RC4 stream cipher. *Designs, Codes and Cryptography*, pages 269-286, vol. 48, no. 3, September, 2008.
10. L. R. Knudsen, W. Meier, B. Preneel, V. Rijmen and S. Verdoolaege. Analysis Methods for (Alleged) RCA. ASIACRYPT 1998, pages 327-341, vol. 1514, Lecture Notes in Computer Science, Springer-Verlag.
11. LAN/MAN Standard Committee. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, 1999 edition. IEEE standard 802.11, 1999.
12. S. Maitra and G. Paul. New Form of Permutation Bias and Secret Key Leakage in Keystream Bytes of RC4. FSE 2008, pages 253-269, vol. 5086, Lecture Notes in Computer Science, Springer Berlin / Heidelberg.
13. I. Mantin and A. Shamir. A Practical Attack on Broadcast RC4. FSE 2001, pages 152-164, vol. 2355, Lecture Notes in Computer Science, Springer-Verlag.
14. I. Mantin. A Practical Attack on the Fixed RC4 in the WEP Mode. ASIACRYPT 2005, pages 395-411, volume 3788, Lecture Notes in Computer Science, Springer-Verlag.
15. I. Mantin. Predicting and Distinguishing Attacks on RC4 Keystream Generator. EUROCRYPT 2005, pages 491-506, vol. 3494, Lecture Notes in Computer Science, Springer-Verlag.
16. I. Mantin. Analysis of the stream cipher RC4. Master's Thesis, The Weizmann Institute of Science, Israel, 2001.
17. A. Maximov. Two Linear Distinguishing Attacks on VMPC and RC4A and Weakness of RC4 Family of Stream Ciphers. FSE 2005, pages 342-358, vol. 3557, Lecture Notes in Computer Science, Springer-Verlag.
18. A. Maximov and D. Khovratovich. New State Recovering Attack on RC4. CRYPTO 2008, pages 297-316, vol. 5157, Lecture Notes in Computer Science, Springer.
19. I. Mironov. (Not So) Random Shuffles of RC4. CRYPTO 2002, pages 304-319, vol. 2442, Lecture Notes in Computer Science, Springer-Verlag.
20. New European Schemes for Signatures, Integrity, and Encryption. Available at `https://www.cosic.esat.kuleuven.be/nessie` [last accessed on September 19, 2008].

21. G. Paul, S. Rathi and S. Maitra. On Non-negligible Bias of the First Output Byte of RC4 towards the First Three Bytes of the Secret Key. Proceedings of the International Workshop on Coding and Cryptography (WCC) 2007, pages 285-294. Extended version available at Designs, Codes and Cryptography 49:1-3, December 2008.
22. G. Paul and S. Maitra. Permutation after RC4 Key Scheduling Reveals the Secret Key. SAC 2007, pages 360-377, vol. 4876, Lecture Notes in Computer Science, Springer Berlin / Heidelberg.
23. G. Paul and S. Maitra. RC4 State Information at Any Stage Reveals the Secret Key. IACR Eprint Server, eprint.iacr.org, number 2007/208, June 1, 2007. This is an extended version of [22].
24. G. Paul, S. Maitra and R. Srivastava. On Non-Randomness of the Permutation after RC4 Key Scheduling. Applied Algebra, Algebraic Algorithms, and Error Correcting Codes (AAECC) 2007, pages 100-109, vol. 4851, Lecture Notes in Computer Science, Springer Berlin / Heidelberg.
25. S. Paul and B. Preneel. Analysis of Non-fortuitous Predictive States of the RC4 Keystream Generator. INDOCRYPT 2003, pages 52-67, vol. 2904, Lecture Notes in Computer Science, Springer-Verlag.
26. S. Paul and B. Preneel. A New Weakness in the RC4 Keystream Generator and an Approach to Improve the Security of the Cipher. FSE 2004, pages 245-259, vol. 3017, Lecture Notes in Computer Science, Springer-Verlag.
27. A. 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 at `http://groups.google.com/group/sci.crypt.research/msg/078aa9249d76eacc?dmode=source` [last accessed on July 18, 2008].
28. RSA Lab Report. RSA Security Response to Weaknesses in Key Scheduling Algorithm of RC4. Available at `http://www.rsa.com/rsalabs/node.asp?id=2009` [last accessed on July 18, 2008].
29. E. Tews, R. P. Weinmann and A. Pyshkin. Breaking 104 bit WEP in less than 60 seconds. IACR Eprint Server, eprint.iacr.org, number 2007/120, April 1, 2007 [last accessed on July 18, 2008].
30. V. Tomasevic, S. Bojanic and O. Nieto-Taladriz. Finding an internal state of RC4 stream cipher. *Information Sciences*, pages 1715-1727, vol. 177, 2007.
31. Y. Tsunoo, T. Saito, H. Kubo, M. Shigeri, T. Suzaki and T. Kawabata. The Most Efficient Distinguishing Attack on VMPC and RC4A. SKEW 2005. Available at `http://www.ecrypt.eu.org/stream/papers.html` [last accessed on July 18, 2008].
32. Y. Tsunoo, T. Saito, H. Kubo and T. Suzaki. A Distinguishing Attack on a Fast Software-Implemented RC4-Like Stream Cipher. *IEEE Transactions on Information Theory*, page 3250-3255, vol. 53, issue 9, September 2007.
33. S. Vaudenay and M. Vuagnoux. Passive-Only Key Recovery Attacks on RC4. SAC 2007, pages 344-359, vol. 4876, Lecture Notes in Computer Science, Springer Berlin / Heidelberg.
34. B. Zoltak. VMPC One-Way Function and Stream Cipher. FSE 2004, pages 210-225, vol. 3017, Lecture Notes in Computer Science, Springer-Verlag.
35. Goutam Paul and Subhamoy Maitra. On Non-negligible Bias of the First Output Byte of RC4 towards the First Three Bytes of the Secret Key, Cryptography and Communications, 22 December, 2008.