STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LV**, Number 1, 2010

SOME COMBINATORIAL ASPECTS OF THE KSAm-LIKE ALGORITHMS SUITABLE FOR RC4 STREAM CIPHER

BOGDAN CRAINICU, FLORIAN MIRCEA BOIAN

Abstract. RC4 remains one of the most widely used stream cipher. In order to face the main critical weaknesses, a number of proposals for modifying RC4 algorithm have been advanced. In this paper we analyze some combinatorial aspects regarding the randomness of a variant of the Key-Scheduling Algorithm (KSA), called KSAm, proposed by Crainicu and Boian in [2] as a better protection against Initialization Vectors (IVs) weakness of Wired Equivalent Privacy (WEP) cryptosystems. Based on a model presented by Mironov in [19], we calculate the sign of the entries' permutation of the internal state table S after KSAm, which provides a negligible advantage of guessing a particular bit. Then, we analyze the probability of the event where a particular initial value follows a linear forward movement through the vector S, with possible undesirable consequences in predicting the value during that movement.

1. Introduction

RC4 is a stream cipher which was designed by Ron Rivest in 1987 for RSA Security. RC4 was kept as a trade secret until an alleged copy of it was anonymously posted to the Cypherpunks mailing list in 1994.

Because of its simplicity and speed, RC4 is one of the most widely used stream cipher; for example, it is used in the SSL/TLS (Secure Socket Layer/ Transport Layer Security) standards, WEP (Wired Equivalent Privacy), WPA (Wi-Fi Protected Access), and it can be also found in email encryption products.

There were discovered many significant weaknesses of RC4 and RC4-based WEP implementations: weak IVs/keys [1, 4, 8, 9, 10, 13, 14, 21, 22, 26, 28], invariance weakness [4], bias in the second output [16], related key attack [4,

Received by the editors: January 2, 2010.

2010 *Mathematics Subject Classification.* 68P25, 94A60, 11T71.

1998 *CR Categories and Descriptors.* G.2.1 [**Mathematics of Computing**]: Combinatorics – *Permutation and combinations*; G.3 [**Mathematics of Computing**]: Probability and Statistics – *Random number generation*; E.3 [**Data**]: Data Encrypting – *Code breaking.*

Key words and phrases. KSA, KSAm, RC4, sign of permutation, randomness of state table.

7], state recovery attack [11, 20, 23, 27], distinguishing attack [3, 5, 16-19, 24, 25], biased distribution of RC4 initial permutation [24, 28].

In order to address its critical weaknesses, a number of proposals for modifying RC4 algorithm have been advanced: Paul and Preneel present in [25] a new pseudorandom bit generator called RC4A, Zoltak proposes in [29] the VMPS stream cipher, and Gong, Gupta, Hell and Nawaz also propose in [6] a new 32/64-bit RC4-like keystream generator.

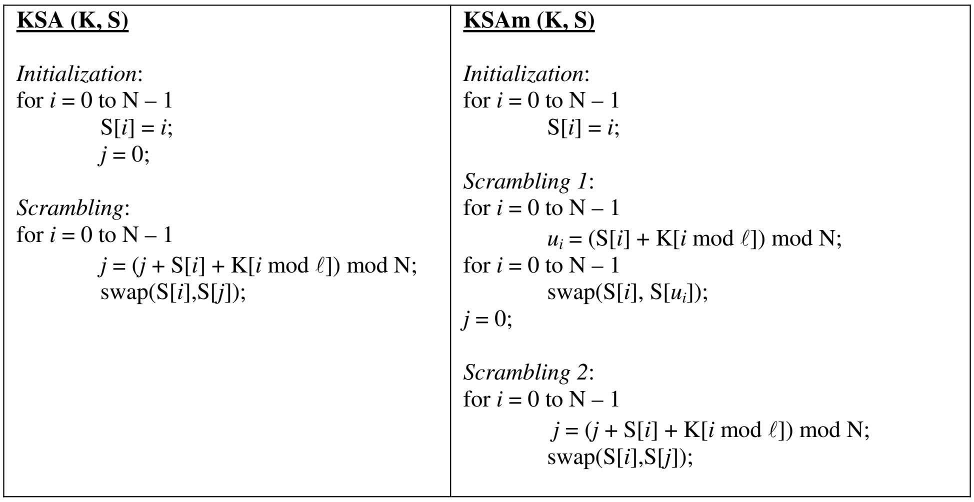
Crainicu and Boian proposed in [2] a modified version of KSA, called KSAm, whose primary goal was to face the weakness exhibited by Fluhrer, Mantin and Shamir (FMS) in [4], where certain IVs leaks information about the secret key in WEP mode of operation. The authors demonstrate that the attacker has no possibilities to manipulate KSAm permutation in order to reach the FMS resolved condition. Based on the Roos' experimental observation [26], there is a weaker probabilistic correlation between the first three words of the secret key and the first three entries of the state table after KSAm, which causes a negligible bias of the first word of the $\mathrm{RC4}_{KSAm}$ (RC4 with KSAm as Key-Scheduling Algorithm) output stream towards the sum of the first three words of the secret key. The effect of this negligible bias can be easily avoided by discarding only the first word from the $\mathrm{RC4}_{KSAm}$ output stream.

In this paper, we examine two combinatorial properties of the KSAm in its normal mode of operation, and not from the perspective of a particular implementation. Firstly, based on a model of a shuffling technique presented by Mironov in [19], where an idealized RC4 stream cipher is involved, we comute the sign of the permutation of S after KSAm, whose values help to predict a bit b with a probability of 0.91% over a random guess. This advantage is too small to be feasible in an attack. We also analyze the state table entries during the KSAm steps, with special focus on calculating the probability of a linear advance movement of an initial value from a particular state table entry during KSAm. The results prove that it is very unlikely to find a location in S during such movement where that value may be predicted with a probability significantly greater than 1/N.

2. KSAm

Crainicu and Boian suggest in [2] a modified version of the original KSA, called KSAm (Fig. 1), for addressing the FMS weakness of WEP-like cryptosystems, mainly when IV precedes the secret key.

The KSAm takes the secret key and initializes a vector of indices $u_0, u_1, \dots,$ u_{N-1}; the values of indices u_i are not necessarily unique within the vector of indices, and they are kept secret. Then, it swaps the two values of S pointed to



```
KSA (K, S)

Initialization:
for i = 0 to N – 1
        S[i] = i;
        j = 0;

Scrambling:
for i = 0 to N – 1
        j = (j + S[i] + K[i mod ℓ]) mod N;
        swap(S[i],S[j]);
```

```
KSAm (K, S)

Initialization:
for i = 0 to N – 1
        S[i] = i;

Scrambling 1:
for i = 0 to N – 1
        u_i = (S[i] + K[i mod ℓ]) mod N;
for i = 0 to N – 1
        swap(S[i], S[u_i]);
j = 0;

Scrambling 2:
for i = 0 to N – 1
        j = (j + S[i] + K[i mod ℓ]) mod N;
        swap(S[i],S[j]);
```

Figure 1. KSA vs KSAm [2]

by i and u_i, so that the *Scrambling 1* stage of KSAm ends with a secret state, which is different from the identity permutation with a very high probability. The rest of operations (*Scrambling 2*) remain the same as in the original KSA: it applies the scrambling rounds $N = 2^n$ times, stepping i across S, updating j by adding the previous value of $j, S[i]$ and the next word of the key.

In fact, KSAm comprises a family of key scheduling algorithms, where *Scrambling 1* sequence tries to follow the Knuth's observation [12]: instead of swapping $S[i]$ with a random entry, it must be swapped with an entry randomly chosen from $S[i]$ to $S[N-1]$ (the implementation of this concept remains still problematic due to the randomness of the secret key K).

At a glance, the first observation is that there are now two different scrambling processes, both of them based on the same secret key. Even if the computation/running time of KSAm is almost twice as long as that of KSA, the additional time is insignificant (the software implementation remains very fast).

The security of KSAm comes also from its huge internal state. The internal state of RC4$_{KSA}$ is approximately 1700 bits for 8-bits words. Instead, KSAm provides a much larger size and, as result, it is much harder to reconstruct its internal state. Crainicu and Boian present in [2] the formula for calculating L$_{RC4-KSAm}$, which represents the size of the RC4$_{KSAm}$' internal state (the values of indices u_i are not necessarily unique; therefore, the number of all possibilities of distributing 2^n elements into 2^n cells where repetitions are allowed is $(2^n)^{2^n}$) [2]:

$$L_{RC4-KSAm} = \log_2(2^n! \times (2^n)^{2^n} \times (2^n)^2)] = [\log_2(2^n!) + (n \times 2^n) + 2n]$$
$$L_{RC4-KSAm,n=8} \approx 3748 \text{ bits}$$

Beside KSA, KSAm needs only additional 256 bytes of memory for the indices u_i (for $N = 8$), which represents a negligible amount of supplementary memory.

3. On Randomness of KSAm

Two independent scrambling processes are involved with KSAm; therefore, after running consecutively both of them, each element of the state table will be swapped at least twice (possibly with itself).

Based on a series of significant studies on the original KSA [5, 16, 17, 19, 20, 23, 24], two of the most important approaches for analyzing KSAm are to deduce the probability of a linear advance movement of a value b which each step along the locations of vector S, and also to calculate the probability of a value b to end up in any location a (including the probability of identity permutation).

3.1. **The sign of the permutation S after KSAm.**

Mironov calculates in [19] the limiting distribution for the two possible values (+1 and −1) of the sign of S after KSA:

$$P(sign(S) = (-1)^N) = \frac{1}{2}\left(1 + e^{-2}\right)$$
$$P(sign(S) = (-1)^{N-1}) = \frac{1}{2}\left(1 - e^{-2}\right)$$

Therefore, Mironov demonstrates that it is possible to predict the sign of the permutation S after KSA with probability $1 - \frac{1}{2} \cdot e^{-2}$, and he shows that this value of about 6.7% over a random guess becomes also the advantage of guessing the bit b correctly.

Next, we compute the sign of the permutation S after KSAm. Two scrambling sequences are involved in KSAm, and therefore we have $2N$ rounds. At each round and regardless of what scrambling sequence is about, the swap process changes the sign of the permutation only if $i \neq j$. This happens with probability $\left(1 - \frac{1}{N}\right)$. If $i = j$, with probability $\frac{1}{N}$, the values of S pointed by i and j remain unchanged. The probability that $i \neq j$ during all $2N$ rounds of KSAm is $\left(1 - \frac{1}{N}\right)^{2N}$, and the probability that $i = j$ during all $2N$ rounds of KSAm is $\left(\frac{1}{N}\right)^{2N}$. The sign of the identity permutation is +1.

Based on these observations, the probability that the sign of S is changed an even number of times and at the end of KSAm is negative, is:

$$P(sign(S) = (-1)^{2N}) = \left(1-\frac{1}{N}\right)^{2N} + C_{2N}^{2} \cdot \left(1-\frac{1}{N}\right)^{2N-2} \cdot \frac{1}{N^2} +$$
$$C_{2N}^{4} \cdot \left(1-\frac{1}{N}\right)^{2N-4} \cdot \frac{1}{N^4} + \ldots + C_{2N}^{N} \cdot \left(1-\frac{1}{N}\right)^{N} \cdot \frac{1}{N^N} +$$
$$C_{2N}^{N+2} \cdot \left(1-\frac{1}{N}\right)^{N-2} \cdot \frac{1}{N^{N+2}} + \ldots + C_{2N}^{2N} \cdot \left(1-\frac{1}{N}\right)^{0} \cdot \frac{1}{N^{2N}} =$$
$$\left(1-\frac{1}{N}\right)^{2N} \cdot \left[1 + \frac{2N\cdot(2N-1)}{2!\cdot N^2} \cdot \left(1-\frac{1}{N}\right)^{-2} + \right.$$
$$\left. \frac{2N\cdot(2N-1)\cdot(2N-2)\cdot(2N-3)}{4!\cdot N^4} \cdot \left(1-\frac{1}{N}\right)^{-4} + \ldots + \frac{1}{N^{2N}}\right] \to$$
$$e^{-2} \cdot \left(1 + \frac{4}{2!} + \frac{16}{4!} + \frac{64}{6!} + \ldots\right) =$$
$$\frac{e^{-2}}{2} \cdot \left(2 + \frac{8}{2!} + \frac{32}{4!} + \frac{128}{6!} + \ldots\right) \xrightarrow[N\to\infty]{} \frac{e^{-2}}{2}(e^2 + e^{-2}) = \frac{1}{2} \cdot \left(1 + e^{-4}\right)$$

The probability that the sign of S is changed an odd number of times and at the end of KSAm is negative, is:

$$P(sign(S) = (-1)^{2N-1}) = 1 - P(sign(S) = (-1)^{2N}) = \frac{1}{2} \cdot \left(1 - e^{-4}\right).$$

After KSAm, the sign of S can be predicted with an very small advantage of $\frac{e^{-4}}{2} \approx 0,0091$ over the random guess, which means approximately 0.91%. Mironov obtains in [19] approximately the same result by running consecutively two times the original KSA.

Even if we found a small bias towards the sign of the permutation after KSAm, the value $\frac{e^{-4}}{2}$ is totally useless for attacking RC4 based on KSAm. As further precaution, discarding only the first three words of the output of RC4 ensures the security of the algorithm.

3.2. **Probability of a linear advance movement of an initial value from a particular state table entry during KSAm.** We define the minimum probability of a given initial entry $S_0[a] = a$ as the probability that this entry remains unchanged during each step of one of the two scrambling processes of KSAm. Thus, the minimum probability of the identity permutation is defined as the multiplication of all minimum probabilities corresponding to each initial entry $S_0[a] = a$.

We refine the Theorem 1 from [2]:

Theorem 1. *The minimum probability of a given initial entry $S_0[a] = a$ after N steps is:*

$$P(S_N[a] = a) = \frac{1}{N} \cdot \left(1 - \frac{1}{N}\right)^{N-1} \tag{1}$$

Proof. At a some point, the index i touches the value a. In this round, with probability $1/N$, $i = j = a$, and therefore $S[a]$ will be swapped with itself. For the rest of the $(N - 1)$ rounds we have $i \neq a$, and $j \neq a$ with probability $\left(1 - \frac{1}{N}\right)$.

The minimum probability of the identity permutation, which means that all N entries of table S remain unchanged after completion of one of the two scrambling process is:

$$P(S_N = \text{identity\_permutation}) = \left[\frac{1}{N} \cdot \left(1 - \frac{1}{N}\right)^{N-1}\right]^N \tag{2}$$

□

Result 1. [19]: The probability of the value b to end up in location a in S at the end of the KSA round (the distribution of KSA outputs) is:

$$P[S_N[a] = b] = \begin{cases} \frac{1}{N}\left[\left(1 - \frac{1}{N}\right)^{N-a-1} + \left(1 - \frac{1}{N}\right)^{b}\right] & \text{if } a < b \\ \\ \frac{1}{N}\left[\left(1 - \frac{1}{N}\right)^{b} + \left(1 - \left(1 - \frac{1}{N}\right)^{b}\right)\ \left(1 - \frac{1}{N}\right)^{N-a-1}\right] & \text{if } a \geq b \end{cases}$$

For example, the minimum probability for event $S_N[0] = 0$ is $\frac{1}{N} \cdot \left(1 - \frac{1}{N}\right)^{N-1}$, and according to Result 1, the probability for event $S_N[0] = 0$ is $\frac{1}{N}$.

Theorem 2. *Assuming that both indices u_i and j take values independently and uniformly at random at each round of the two scrambling processes of KSAm, the probability of the value b to end up in location $a + 1$ in S in the round a of Scrambling 1 or Scrambling 2, where $S_0[0] = b$, is:*

$$P(S_a[(a + 1) \bmod N] = b) = \frac{1}{N}, a \in [1, N] \text{ and } b \in [0, N - 1] \tag{3}$$

Proof. At each round, taking into account that the value of i is known, we can analyze the probability of the event $S_a[(a + 1) \bmod N] = b$, which depends on the value taken by u_i or j. The value b, during the a rounds, must not remain behind a location pointed by $i(i \in [0, (a + 1) \bmod N])$ nor to reach a position after the $[(a + 1) \bmod N]^{th}$ location in S, because, in these situations, there are no possibilities to manipulate the value b so that it ends up in location $[(a+1) \bmod N]$. Following the steps of either *Scrambling 1* or *Scrambling 2*, we have:

$S_1[2] = b$ if $j_1 = 2$ (the initial condition is $S_0[0] = b$) $\Rightarrow$

$P(S_1[2] = b) = P(j_1 = 2) = \dfrac{1}{N}$;

$S_2[3] = b$ if $< j_1 = 1$ and $j_2 = 3 >$ or $< j_1 = 3$ and $j_2 \neq 3 > \Rightarrow$

$P(S_2[3] = b) = P(j_1 = 1) \cdot P(j_2 = 3) + P(j_1 = 3) \cdot P(j_2 \neq 3) =$

$\dfrac{1}{N^2} + \dfrac{1}{N} \cdot \left(1 - \dfrac{1}{N}\right) = \dfrac{1}{N}$;

$S_3[4] = b$ if $< j_1 = 1$ and $j_2 = 2$ and $j_3 = 4 >$ or $< j_1 = 1$ and $j_2 = 4$ and $j_3 \neq 4 >$ or

$< j_1 = 2$ and $j_2 \neq 2$ and $j_3 = 4 >$ or $< j_1 = 4$ and $j_2 \neq 4$ and $j_3 \neq 4 > \Rightarrow$

$P(S_3[4]) = P(j_1 = 1) \cdot P(j_2) = 2) \cdot P(j_3 = 4) + P(j_1 = 1) \cdot P(j_2 = 4) \cdot P(j_3 \neq 4) +$

$P(j_1 = 2) \cdot P(j_2 \neq 2) \cdot P(j_3 = 4) + P(j_1 = 4) \cdot P(j_2 \neq 4) \cdot P(j_3) \neq 4) =$

$\dfrac{1}{N^3} + \dfrac{2}{N^2} \cdot \left(1 - \dfrac{1}{N}\right) + \dfrac{1}{N} \cdot \left(1 - \dfrac{1}{N}\right)^2 = \dfrac{1}{N}$;

$$\begin{aligned}
&\ldots \\
&P(S_4[5] = b) = \frac{1}{N^4} + \frac{3}{N^3} \cdot \left(1 - \frac{1}{N}\right) + \frac{3}{N^2} \cdot \left(1 - \frac{1}{N}\right)^2 + \\
&+\frac{1}{N} \cdot \left(1 - \frac{1}{N}\right)^3 = \frac{1}{N} \\
&P(S_5[6] = b) = \frac{1}{N^5} + \frac{4}{N^4} \cdot \left(1 - \frac{1}{N}\right) + \frac{6}{N^3} \cdot \left(1 - \frac{1}{N}\right)^2 + \\
&+\frac{4}{N^2} \cdot \left(1 - \frac{1}{N}\right)^3 + \frac{1}{N} \cdot \left(1 - \frac{1}{N}\right)^4 = \frac{1}{N}; \\
&\ldots
\end{aligned}$$

$$P(S_a[a+1]=b) = \frac{1}{N} \cdot \left(1-\frac{1}{N}\right)^{a-1} + \frac{(a-1)}{1! \cdot N^2} \cdot \left(1-\frac{1}{N}\right)^{a-2} +$$
$$\frac{(a-1)\cdot(a-2)}{2! \cdot N^3} \cdot \left(1-\frac{1}{N}\right)^{a-3} + \frac{(a-1)\cdot(a-2)\cdot(a-3)}{3! \cdot N^4} \cdot \left(1-\frac{1}{N}\right)^{a-4} +$$
$$\frac{(a-1)\cdot(a-2)\cdot(a-3)\cdot(a-4)}{4! \cdot N^5} \cdot \left(1-\frac{1}{N}\right)^{a-5} + \ldots + \frac{1}{N^a} = \frac{1}{N}$$

□

For *Scrambling 1*, $b = 0$, and for *Scrambling 2*, $b \in [0, 255]$. Applying the Result 1, where the initial state is the identity permutation, for $b = 0$ and $a = 1$, we obtain $P[S_N[1] = 0] = \frac{1}{N}$, and applying the Theorem 2 for $a = N$, we have the same result: $P(S_N[(N+1)\text{mod}N] = 0) = \frac{1}{N}$.

Theorem 2 can be adapted for all entries of the initial permutation S_0. The significance of Theorem 2 lies in the fact that the event that a particular value b follows a linear path through the vector S, and consequently ends up in an expected location after *Scrambling1*, has a probability around $\frac{1}{N}$. This result has also to be considered in the context of starting *Scrambling 2* with a state table S which is different from the identity permutation with high probability.

4. Conclusions

We investigated KSAm [2] from the point of view of shuffling algorithm presented by Mironov in [19]. We calculated as well the probability of the sign of the permutation S, but after KSAm, finding a value which may help in predicting that sign with an advantage of 0.91% over a random guess. Mironov obtains in [19] about the same result by running consecutively two times the original KSA, but KSAm benefits from a much larger size of the internal state and the running of two different scrambling processes. The mentioned advantage of 0.91% over a random guess, biased towards the sign of the permutation after KSAm, is though too small, so that an attack against RC4$_{KSAm}$ could not rely on it. As precaution, an additional measure for thwarting against this weakness consists in discarding only the first three words of the output of RC4$_{KSAm}$.

The second part of our analysis is focused on calculating the probability of a particular event, namely, a linear advance movement of the state table entry $S_0[0] = b$ during KSAm rounds. The value obtained is about $\frac{1}{N}$, which demonstrates that such event happens randomly. The result can be extended to the others entries of the initial state table S.

References

[1] A. Bittau, *Additional weak IV classes for the FMS attack*, Department of Computer Science, University College London, 2003. Available: http://www.cs.ucl.ac.uk/staff/a.bittau/sorwep.txt.

[2] B. Crainicu, F. M. Boian, *KSAm – An Improved RC4 Key-Scheduling Algorithm for Securing WEP*, in Novel Algorithms and Techniques in Telecommunications and Networking, Springer, Netherlands 2010, ISBN 978-90-481-3661-2, pp. 391-396.

[3] S. Fluhrer, D. McGrew, *Statistical analysis of the alleged RC4 keystream Generator*, in. Proc. 7th International Workshop, FSE 2000, New York, Lecture Notes in Computer Science, Vol. 1978, Springer-Verlag, 2001, pp. 66-71.

[4] S. Fluhrer, I. Mantin, A. Shamir, *Weaknesses in the key scheduling algorithm of RC4*, in Proc. 8th Annual International Workshop, SAC 2001, Toronto, Lecture Notes in Computer Science, Vol. 2259, Springer-Verlag, 2001 pp. 1-24.

[5] J. Dj. Goli, *Linear statistical weakness of alleged RC4 keystream generator*, in. Proc. International Conference on the Theory and Application of Cryptographic Techniques, EUROCRYPT '97, Konstanz, Lecture Notes in Computer Science, Vol. 1233, Springer-Verlag, 1997, pp. 226-238.

[6] G. Gong, K. C. Gupta, M. Hell, Y. Nawaz, *Towards a General RC4-like Keystream Generator*, in Proc. First SKLOIS Conference*, CISC 2005*, Beijing, Lecture Notes in Computer Science, Vol. 3822, Springer-Verlag, 2005, pp. 162-174.

[7] A. L. Grosul, D. S. Wallach, *A related key cryptanalysis of RC4*, Technical Report TR-00-358, Department of Computer Science, Rice University, 2000. Available: www.weizmann.ac.il/mathusers/itsik/RC4/Papers/GrosulWallach.ps

[8] D. Hulton, *Practical exploitation of RC4 weaknesses in WEP environments*, 2001. Available: http://www.datastronghold.com/security-articles/hacking-articles/practical-exploitation-of-rc4-weaknesses-in-wep-environments.html

[9] KoreK, *Need security pointers*, 2004. Available: http://www.netstumbler.org/showthread.php?postid=89036#post89036.

[10] KoreK, *Next generation of WEP attacks?*, 2004. Available http://www.netstumbler.org/showpost.php?p=93942&postcount=35

[11] L. R. Knudsen, W. Meier, B. Preneel, V. Rijmen, S. Verdoolaege, *Analysis Methods for (Alleged) RC4*, in Proc. International Conference on the Theory and Application of Cryptology and Information Security, ASIACRYPT'98, Beijing, Lecture Notes in Computer Science, Springer-Verlag, Vol.1514, 1998, pp.327–341.

[12] E. Knuth, *The Art of Computer Programming*, Third edition, Volume 2, Addison-Wesley, 1997.

[13] K. Kobara, H. Imai, *Key-Dependent Weak IVs and Weak Keys in WEP – How to Trace Conditions Back to Their Patterns –*, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E89-A, No. 8, 2006, pp. 2198-2206.

[14] K. Kobara, H. Imai, *IVs to Skip for Immunizing WEP against FMS Attack*, IEICE Transactions on Communications, Vol.E91–B, No.1, 2008, pp. 218-227.

[15] I. Mantin, *The Security of the Stream Cipher RC4*, Master Thesis, The Weizmann Institute of Science, 2001.

[16] I. Mantin, A. Shamir, *A practical attack on broadcast RC4*, in Proc. 8th International Workshop, FSE 2001, Yokohama, Lecture Notes in Computer Science, Springer-Verlag, Vol. 2355, 2002, pp. 87-104.

[17] I. Mantin, *Predicting and Distinguishing Attacks on RC4 Keystream Generator*, in. Proc. 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, EUROCRYPT 2005, Aarhus, Lectures Notes in Computer Science, Vol. 3494, Springer-Verlag, 2005, pp. 491-506.

[18] I. Mantin, *A Practical Attack on the Fixed RC4 in the WEP Mode*, in Proc. 11th International Conference on the Theory and Application of Cryptology and Information Security, ASIACRYPT 2005, Chennai, Lecture Notes in Computer Science, Springer-Verlag, Vol. 3788, 2005, pp. 395-411.

[19] I. Mironov, *(Not So) Random Shuffles of RC4*, in Proc. 22nd Annual International Cryptology Conference, Advances in Cryptology, CRYPTO 2002, Santa Barbara, Lecture Notes in Computer Science, Springer-Verlag, Vol. 2442, 2002, pp. 304–319.

[20] S. Mister, S. E. Tavares, *Cryptanalysis of RC4-like Ciphers*, in Proc. 5th Annual International Workshop, SAC 1998, Kingston, Lecture Notes in Computer Science, Springer-Verlag, Vol.1556, 1999, pp. 131–143.

[21] T. Ohigashi, Y. Shiraishi, M. Morii, *Most IVs of FMS Attack-Resistant WEP Implementation Leak Secret Key Information*, in Proc. 2005 Symposium on Cryptography and Information Security, Maiko, Vol. 4, 2005, pp. 1957–1962.

[22] T. Ohigashi, Y. Shiraishi, M. Morii, *FMS Attack-Resistant WEP Implementation Is Still Broken – Most IVs Leak a Part of Key Information –* , in Proc. International Conference, CIS 2005, Xi'an, Lecture Notes in Computer Science, Springer-Verlag, Vol. 3802, 2005, pp. 17-26.

[23] T. Ohigashi, Y. Shiraishi, M. Morii, *New Weakness in the Key-Scheduling Algorithm of RC4*, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E91-A, No. 1, 2008, pp. 3-11.

[24] S. Paul, B. Preneel, *Analysis of Non-fortuitous Predictive States of the RC4 Keystream Generator*, in Proc. 4th International Conference on Cryptology in India, INDOCRYPT 2003, New Delhi, Lecture Notes in Computer Science, Springer-Verlag, Vol. 2904, 2002, pp. 52-67.

[25] S. Paul, B. Preneel, *A New Weakness in the RC4 Keystream Generator and an Approach to Improve the Security of the Cipher*, in Proc. 11th International Workshop, FSE 2004, Delhi, Lecture Notes in Computer Science, Springer-Verlag, Vol. 3017, 2004, pp. 245–259.

[26] A. Roos, *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.

[27] Y. Shiraishi, T. Ohigashi, M. Morii, *An improved Internal-State Reconstruction Method of a Stream Cipher RC4*, in Proc. IASTED International Conference on Communication, Network, and Information Security, CNIS 2003, New York, 2003, pp. 132-135.

[28] D. Wagner, *My RC4 weak keys*, Post in sci.crypt, message-id 447o1l$cbj@cnn.princeton.edu, 1995. Available: http://www.cs.berkeley.edu/˜daw/my-posts/my-rc4-weak-keys

[29] B. Zoltak, *VMPC One-Way Function and Stream Cipher*, in Proc. 11th International Workshop, FSE 2004, Delhi, Lectures Notes in Computer Science, Vol. 3017, Springer-Verlag, 2004, pp. 210–225.

"Petru Maior" University of Tîrgu-Mureş
E-mail address: cbogdan@upm.ro