On a new properties of random number sequences ,a randomness test and a new RC4's key scheduling algorithm.

Samir Bouftass[1] and Abdelhak Azhari[2]

[1]Crypticator.Inc , e-mail : crypticator@gmail.com.
[2]University Hassan II , e-mail : aazhari2001@yahoo.fr.

June 16, 2014

Abstract

In this paper, we introduce the concept of the derivative of sequence of numbers and define new statistical indices by which we discoverd new properties of randomly generated number sequences. We also build a test for pseudo random generators based on these properties and use it to confirm the weakness of RC4 key scheduling algorithm that has been reported in the litterature. In this rescpect we publish a new RC4's key scheduling algorithm that don't have this weakness.

Keywords : Randomness test, Statistical indices , RC4 key scheduling.

1 Introduction :

Randomness is a fundamental scientific and phisophical concept, it reflects our brains limits to apprehend the complexities of the universe and thence to predict the events occuring in it.
In the field of cryptography , the main objective of a encryptor was always been producing a cipher text , a sequence of symbols or numbers indistinguishable from corresponding sequences generated by a true random process such as the game of lottery or a radioactive decay process.
That means that the more we understand randomness or know properties of true random processes or number sequences, the more it helps cryptographers by devising new tests and use them to analyze and test the security of new or old cryptographic schemes

In this paper we publish a new test of randomness based on newly discoverd properties of random numbers sequences, use it to perform 5 analysis on a key schedule algorithm of a stream cipher namely RC4, and validate it by revealing some of this algorithm's weaknesses already reported in the litterature [2][3][4]. We also propose a new KSA for RC4 that don't have these weaknesses.

2 Definitions :

2.1 Derivative of a sequence of numbers :

Definition :

Let Sn be a sequence of n + 1 natural numbers [s(0) ... s(i) ,s(i + 1) ... s(n)]
while $0 \leq s(0...n) < M$ and M being the smallest natural number greater than $s(0...n)$.

The first derivative of Sn is defined as the sequence of natural numbers

Sn_1 : [$|s(1) - s(0)|...|s(i+1) - s(i)|...|s(n) - s(n-1)|$]

2.2 Interval Average Indices (IAIs) :

Definition :

Let Sn be a sequence of n + 1 natural numbers [$s(0)$ ... $s(i)$ $s(i+1)$ ... $s(n)$]
while $0 \leq s(0...n) < M$ and M being the smallest natural number greater than $s(0...n)$.

Interval Average Index IAI of Sn base M is defined as :

$IAI_M(Sn) = (|s(1) - s(0)| + .. + |s(i+1) - s(i)| + .. + |s(n) - s(n-1)|)/n.$
The length of the derivative of Sn is the length minus one of this last.

The nth+1 derivative of Sn is the first derivative of the nth derivative of Sn.

The mean of the first derivative of Sn is the Interval Average Index IAI base M of this last.

If Sn_n is the nth derivative of Sn, the derivative order of Sn_n relative to Sn is n.

2.3 Derivative sequences Means :

Let Sn be a sequence of n + 1 natural numbers [$s(0)$ ... $s(i)$ $s(i+1)$ ... $s(n)$]
while $0 \leq s(0...n) < M$ and M being the smallest natural number greater than $s(0...n)$.

We have experimentaly observed that If Sn is a randomely choosen sequence of n natural numbers and n increases :

The mean of the first derivative of Sn , Or Interval Average Index Base M of Sn ($IAI_M(Sn)$) converge to a constant value $C = M/3 \pm \epsilon$.

If $M = 2$ then $\epsilon = -(M \times 0.25)$.

If $M > 2$ then $\epsilon \leq |(M \times 0.10)|$.

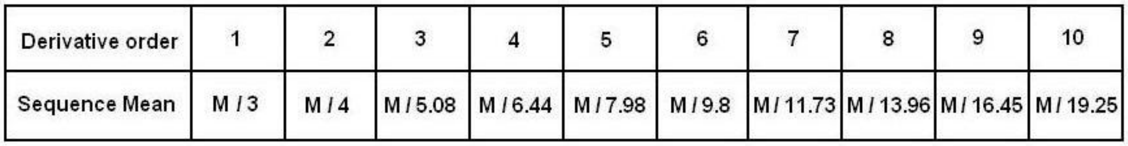
If M increases then $\epsilon \to 0$.

We have equaly experimentaly observed that

The mean of the nth derivative ($n > 1$) of a randomely choosen sequence of natural numbers converge to a constant value $C = M * d \pm \epsilon$.

If M increases then $\epsilon \to 0$.

The table below (figure 1) shows the average means of nth derivatives ($0 \leq n < 11$) of a randomely choosen sequence of 1000000 natural numbers $s(0...999999) < M$.



Derivative order	1	2	3	4	5	6	7	8	9	10
Sequence Mean	M / 3	M / 4	M / 5.08	M / 6.44	M / 7.98	M / 9.8	M / 11.73	M / 13.96	M / 16.45	M / 19.25

Figure 1: Average means of nth derivatives of a random sequence of numbers (n = 0 ... 10)

3 A Test for Pseudo random number generators :

1 - Set value N and value L .

2 - Repeat N Times.

2 - 1 Choose randomely a sequence RSn of L bytes.

2 - 2 Compute IAI_{256} index of RSn.

2 - 3 Produce with the PRNG to be tested a sequence $PrSn$ of L bytes.

2 - 4 Compute IAI_{256} index of $PrSn$.

3 - The test is successful, if the statistical distributions of IAI_{256} indices of the N pseudo randomly produced sequences coincides with those of N Randomely choosen sequences.

4 RC4 key scheduling analysis :

RC4 key scheduling [1] weakness is reported by many authors [2][3][4]. In this section we will use our test for Prngs to show this weakness.

Below is described RC4 algorithm :

Key scheduling algorithm :

Input : Secrete key K composed by l bytes K[0],..,K[i], ..,K[l-1].
Output : S a SBox (8 X 8).

```
for i = 0 to 255
    S[i] : i.

j : 0
for i = 0 to 255
    j : ( j + S[i] + K[i mod l ] ) mod 256
    swap S[i] and S[j]
```

4.1 Analysis 1 :

In this analysis we intialised 10000 RC4's SBoxs with 10000 randomely choosen key.
We constructed then 256 byte sequences whose lenght is 10000.
Each sequence of bytes Sb(i) ($0 \leq i < 256$) is filled with the iths elements of our 10000 SBoxs. (See Figure 2).

If RC4 key scheduling algorithm is strong we couldn't distinguish these 256 sequences of 10000 bytes from a randomely choosen byte sequences of the same length.

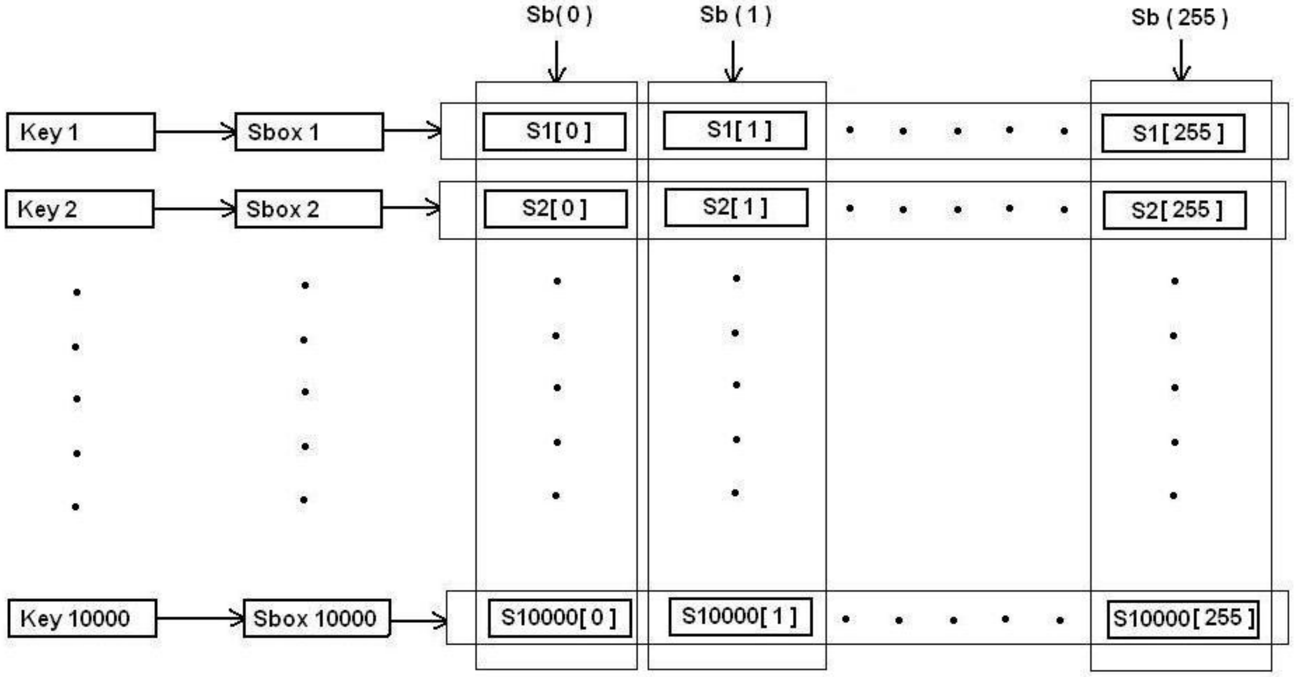


Figure 2: Analysis 1

As random source we have used true random number sequences downloaded from www.random.org. our experiments shows us that :

IAI_{256} indices of random byte sequences whose length is 10000 are included between 83 and 87.

IAI_{256} indices of 78 to 88 from the 256 sequences constructed from 10000 RC4's Sboxs as described above are outside the interval [83 - 87].

This result means that RC4 key scheduling algorithm is inclined to produce Sboxs thats biased toward certain values.

4.2 Analysis 2 :

In this analysis we have intialised 10000 RC4's SBox with 10000 randomely choosen key.
We then have constructed 256 byte sequences whose length is 10000.

After that, each sequence of bytes Sb(i) ($0 \leq i < 256$) is filled with the iths elements of our 10000 Sboxs xored with the first key byte K[0] corresponding to the Sboxs (See Figure 3).

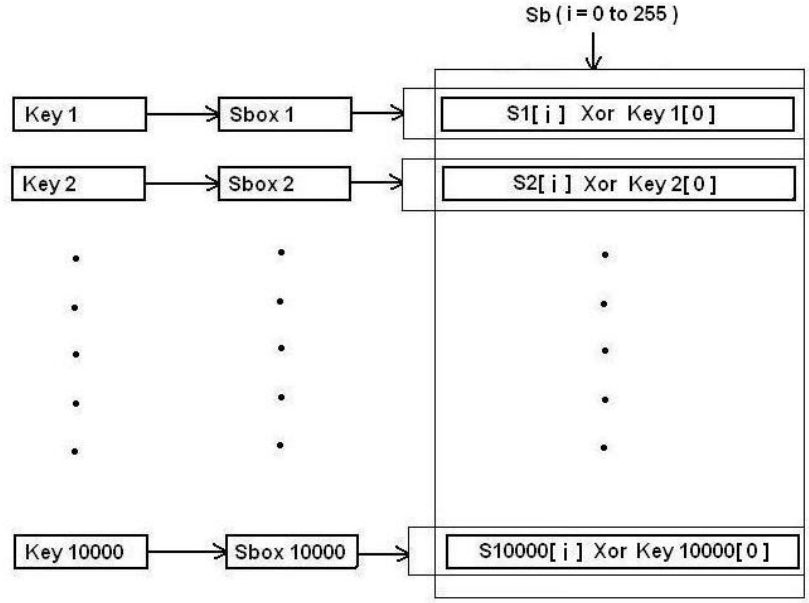


Figure 3: Analysis 2

We have found that the IAI_{256} of sequence Sb(0) is between 92 and 95 then outside the interval [83 - 87], implying that there is a correlation between the first byte of secret keys K[0] and first byte of corresponding RC4's Sboxs S[0].

Next we xored the byte sequences elements with corresponding second keys bytes. IAI_{256} of sequence Sb(1) is outside the interval [83 - 87] that means that S[1] (the second byte of RC4's Sboxs) correlates with the xor sum of the first and second bytes of corresponding secret key K[0] xor K[1].

These correlations equations are showen below :

Equation : $Sbox[0] \oplus K[0] \; = 0$, holds with probability of 1/3 instead of 1/256.

Equation : $Sbox[1] \oplus (K[0] \oplus K[1]) \; = 2^n - 1$, holds with probability P.

If $(n < 8)$, $P = 4.5/256$ instead of 1/256.

If $(n = 8)$, $P = 12.12/256$ instead of 1/256.

We repeated this procedure for the next key bytes and found that :

If $(n < 10)$ is inferior to 10, S[i] the ith elements of RC4s' SBox correlates with $\oplus_0^i$K[i], the xor sum of the i first bytes of corresponding secret key.

4.3 Analysis 3 :

We repeated the protocole of analysis 2 but this time we xored the elements of our bytes sequences with sum modulo 256 of corresponding k bytes we got these correlation equations :

Equation : $Sbox[1] \oplus (K[0] + K[1]) = 1$, holds with probability of 45/256 instead of 1/256.

Equation : $Sbox[1] \oplus (K[0] + K[1]) = 3$, holds with probability of 23/256 instead of 1/256.

Equation : $Sbox[1] \oplus (K[0] + K[1]) = 7$, holds with probability of 12/256 instead of 1/256.

Equation : $Sbox[1] \oplus (K[0] + K[1]) = 15$, holds with probability of 6.5/256 instead of 1/256.

Equation : $Sbox[1] \oplus (K[0] + K[1]) = 31$, holds with probability of 3.5/256 instead of 1/256.

We repeated this procedure for the next key bytes and found that :

If $(n < 30)$, S[i] the ith elements of RC4s' SBox correlates with $\sum_0^i$K[i], the sum modulo 256 of the i first bytes of corresponding secret key.

4.4 Analysis 4 :

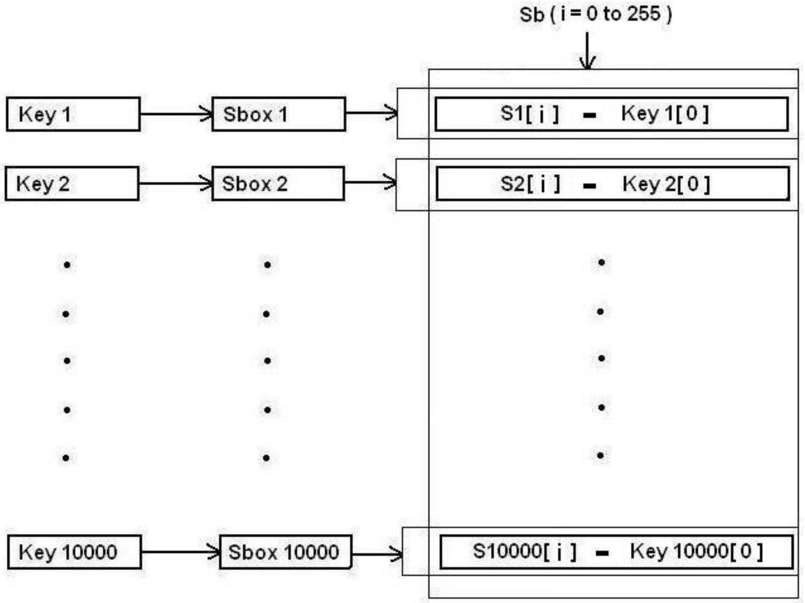


Figure 4: Analysis 4

We repeated the protocole of analysis 2 but this time we substracted the elements of our bytes sequences with sum modulo 256 of corresponding k bytes (see figure 4) we got these correlation equations :

Equation : $Sbox[0] - K[0] \ = 0$, holds with probability of 93/256 instead of 1/256.

Equation : $Sbox[1] - (K[0] + K[1]) \ = 1$, holds with probability of 91/256 instead of 1/256.

Equation : $Sbox[2] - (K[0] + K[1] + K[2]) \ = 3$, holds with probability of 89/256 instead of 1/256.

Equation : $Sbox[3] - (K[0] + K[1] + K[2] + K[3]) \ = 6$, holds with probability of 88/256 instead of 1/256.

Equation : $Sbox[4] - (K[0] + K[1] + K[2] + K[3]) \ = 10$, holds with probability of 86/256 instead of 1/256.

We repeated this procedure for the next key bytes and found that :

Equations : $Sbox[i] - \sum_0^i K[i] = C_i$. ($C_i$ is a constant). holds with probabilities much higher than what suposed to be : 1/256.

5 A new key scheduling algorithm for RC4 :

In our expriments we used our new randomness test to validate what is already reported in the litterature [2][3][4] namely the fact that RC4 key scheduling algorithm is weak and produces Sboxs whose elements are correlating with certain secret key bytes.

In our opinion, RC4 current key scheduling algorithm consists of a unique perumtation encryption stage of the state : the Sbox. We think that a additional substitution stage would strength it. In other hand In each permutation , only one byte of the secret key is used.
That leads us to the key scheduling algorithm described below :

The substitution stage consists in xoring the Sboxs bytes with secret key bytes sum modulo 256, after its initialization to identity permutation.

The new RC4 Key scheduling algorithm :

Input : Secrete key K composed by l bytes K[0],..,K[i], ..,K[l-1].
Output : S , a SBox (8 X 8).

```
k : 0
for i = 0 to l
     k : ( k + K[i] ) mod 256
for i = 0 to 255
     S[i] : i ⊕ k.
j : 0
for i = 0 to 255
     j : ( (j ⊕ k) + S[i] + ( j ⊕ K[i mod l ]) ) mod 256
     swap S[i] and S[j]
```

We performed the 4 analysis described above on this new key scheduling algorithm and found that IAI_{256} of all the sequences Sb(0..255) are included in the interval [83 - 87] proving that this new KSA don't have the weaknesses of RC4's KSA.

5.1 Analysis 5 :

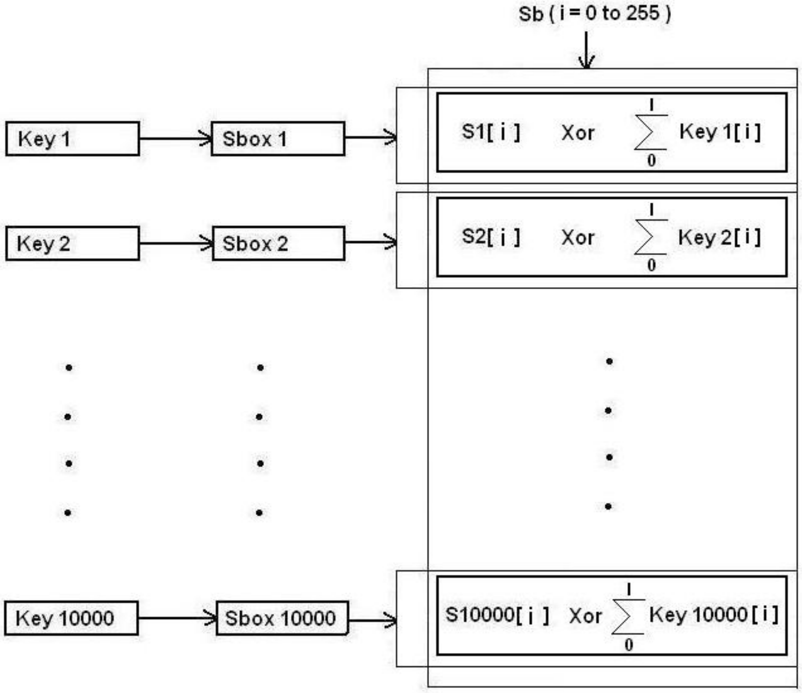

Sb (i = 0 to 255)
Key 1
Sbox 1
S1[i] Xor $\sum_{0}^{l}$ Key 1[i]
Key 2
Sbox 2
S2[i] Xor $\sum_{0}^{l}$ Key 2[i]
Key 10000
Sbox 10000
S10000[i] Xor $\sum_{0}^{l}$ Key 10000[i]

Figure 5: Analysis 5

In this analysis we used the new key scheduling algorithm to intialise 10000 SBoxs with 10000 randomely choosen key. We constructed 256 byte sequences whose length is 10000. We then filled, each sequence of bytes Sb(i) ($0 \leq i < 256$) with the iths elements of our 10000 Sboxs xored with the sum of key bytes corresponding to the Sboxs (See Figure 5).
IAI_{256} indices of 30 to 40 from the 256 sequences constructed from 10000 RC4's Sboxs as described above are outside the interval [83 - 87].
implying that the elements of the Sboxs biased toward secret key bytes sum, which is explainable by the fact that we xored the first with the last during the inialisation.

To remove this bias we modified our KSA :

Input : Secrete key K composed by l bytes K[0],..,K[i], ..,K[l-1].
Output : S a SBox (8 X 8).

```
k : 0
for i = 0 to l
    k : ( k + K[i] ) mod 256
for i = 0 to 255
    S[i] : i ⊕ k
j : 0
for i = 0 to 255
    j : ( (j ⊕ k) + S[i] + ( j ⊕ K[i mod l ]) ) mod 256
    swap S[i] and S[j]
    if (i mod 64) = 0
        for w = 0 to l
            K[w] : S[K[w]]
```

We performed all the five analysis on this version of proposed KSA for RC4 and found that IAI_{256} of all the sequences Sb(0..255) are included in the interval [83 - 87] proving that this new KSA don't have the weaknesses of RC4's original KSA.

6 Conclusion :

In this paper, we introduce the concept of the derivative of sequence of numbers and define new statistical indices by which we discoverd new properties of randomly generated number sequences. We also build a test for pseudo random generators based on these properties and use it to perform 5 analysis on RC4's KSA. These analyses help us to illucidate the cause of RC4's KSA weakness that has been reported in the literrature [2][3][4], which is that RC4's KSA is equivalent to merly one stage of permutation encryption of a identity permutation Sbox. Yet It is Known that permutation alone without subsequent or precedent substitution is cryptographically weaker.

In this respect we produced a new KSA for RC4 that dont have the original RC4's KSA weakness.

References

[1] R.L. Rivest *The RC4 Encryption Algorithm, RSA Data Security,. Inc.,(1992)*

[2] Scott Fluhrer, Itsik Mantin, Adi Shamir *Weaknesses in the key scheduling Algorithm of RC4 , Cisco System, Inc.,The Weizmann Institute (2001)*

[3] Pouyan Sepehrdad, Serge Vaudenay, Martin Vuagnoux *Discovery and Exploitation of New Biases*, (2010), Springer Verlag , New York .

[4] Goutam Paul, Subhamoy Maitra *RC4 stream cipher and its variants*, (2012), CRS Press Taylor and Francis, New York .