Some Security Results of the RC4+ Stream Cipher

Subhadeep Banik and Sonu Jha

***Abstract*—RC4+ stream cipher was proposed by Maitra et. al. at Indocrypt 2008. It was claimed by the authors that this new stream cipher is designed to overcome all the weaknesses reported on the alleged RC4 stream cipher. In the design specifications of RC4+, the authors make use of an 8-bit design parameter called pad which is fixed to the value `0xAA`. The first Distinguishing Attack on RC4+ based on the bias of its first output byte was shown by Banik et. al. in Indocrypt 2013. In this paper, it was also mentioned that the distinguishing attack would still hold if the pad used in RC4+ is fixed to any even 8-bit constant other than `0xAA`. Therefore, the question that arises is whether the design of RC4+ can be protected by fixing the pad parameter to some constant odd value. In this paper, we try to answer this very question. We show that the design is still vulnerable by mounting a distinguishing attack even if the pad is fixed to some constant 8-bit odd value. Surprisingly we find that if the value of the pad is made equal to `0x03`, the design provides maximum resistance to distinguishing attacks. Lastly we return to the original cipher i.e. in which pad is set to `0xAA` and unearth another bias in the second output byte of the cipher, thereby showing that practical implementations of this cipher should discard the use of the first two output bytes for encryption.**

***Index Terms*—Distinguishing Attacks, RC4, RC4+, Stream Ciphers.**

I. Introduction

THE alleged RC4 stream cipher is the most widely used software stream cipher in different popular protocols. Apart from its popularity in the commercial uses, it has also become one of the most involved topics of research for cryptologists. RC4 only requires byte manipulations and hence it is ideal for software implementation. Its simplistic design allows faster encryption in software. Several years of thorough research on the cryptanalysis of the alleged RC4 showed many vulnerabilities and shortcomings of this stream cipher. In [3], a practical attack on broadcast RC4 was also demonstrated which was enough to compromise the security of many popular protocols which used the RC4 encryption scheme. As a result, many researchers tried to focus on designing RC4 like stream ciphers with introducing additional security layers to minimize the reported shortcomings of RC4. Many stream ciphers have been proposed by researchers to fulfill the objective. RC4A [6], GGHN [5], VMPC [7], etc. are such proposed stream ciphers to name a few. Nevertheless, all of the above mentioned stream ciphers have had some reports of distinguishing attacks [8]–[10] against them. In [1], a new stream cipher named RC4+ was introduced. It is a modified version of RC4 with a complex 3-phase key schedule and a more complex output function. The RC4+ stream cipher was primarily designed to over come the weaknesses and shortcomings of RC4 stream cipher. The authors claimed that while being marginally slower than RC4 in software, the RC4+ stream cipher would wipe out all the weaknesses like distinguishing [3] and state recovery attacks [4] on the RC4 stream cipher. The physical structure of the RC4+ stream cipher is same as RC4. The first layer of Key Scheduling Algorithm(KSA) (given in Table I) of the RC4+ stream cipher is similar to KSA of RC4. The second and third layer is described in Table II. However the Pseudo-Random Keystream Generation Algorithm(PRGA) (given in Table III) of RC4+ differs from the PRGA phase of RC4. A detailed description of the RC4+ stream cipher is given in Section II.

TABLE I
Initialization and KSA Layer 1 Scrambling

procedure Initialize(S)	**procedure** Scrambling(S, K)
$i \leftarrow 0$;	$i \leftarrow 0$;
while $i \neq N$ **do**	**while** $i \neq N$ **do**
$S[i] \leftarrow i$;	$j \leftarrow (j + S[i] + K[i])$;
$i \leftarrow i + 1$;	SWAP($S[i], S[j]$);
end while	$i \leftarrow i + 1$;
$j \leftarrow 0$;	**end while**
end procedure	**end procedure**

A. *Existing attacks on RC4+*

In [2], a distinguishing attack on RC4+ was shown, where the authors proved that the first output byte produced by RC4+ stream cipher is negatively biased towards 1. The probability of the first output byte being equal to 1 is around $\frac{1}{N} - \frac{1}{2N^2}$. Based on this observation, they mounted a distinguishing attack on RC4+ which required around 2^{26} output keystreams produced by PRGA of the RC4+ stream cipher. In the same paper, the authors also mounted a differential fault attack on RC4+.

B. *Our Contribution and Organization of the Paper*

The distinguishing attack on RC4+ mentioned in [2] holds also when even pads other than the proposed pad `0xAA` are used in the PRGA. This fact was mentioned explicitly in [2]. However, the question, "Can the design resist the distinguishing attack if the design parameter(pad) is fixed to some 8-bit odd value?", still remains unanswered. In other words, can the design be fixed by simply replacing the even pad with an odd one? In this paper we mount a distinguishing attack on RC4+ in case when odd pads are used in the PRGA. In Section III, we will show that the first output byte produced by RC4+ in case of odd pads (except a special case of pad being equal to `0x03`) is positively biased towards 1. The probability of the first output byte being equal to 1 is around $\frac{1}{N} + \frac{1}{2N^2}$. The required amount of output keystreams produced

S. Banik is Post-Doctoral fellow at DTU Compute, Technical University of Denmark, Lyngby, Denmark. email: subb@dtu.dk
S. Jha is Research Trainee from Kolkata, India. email: jhasonu1987@yahoo.com

by RC4+ to mount the attack in case of odd pads are also 2^{26}. In Section IV we will discuss why it is not possible to find a distinguisher in the special case when the pad used in PRGA is equal to `0x03` in the similar attack scenario used in case of other odd pads. In addition, we will introduce another attack scenario in which it is possible to find a distinguisher bias in case of pad `0x03`. In Section V we will discuss a bias present in the second output byte of the original version of RC4+. In fact, we will prove that the second output byte Z_2 is positively biased towards 0 and the probability of Z_2 being equal to 0 is $\frac{1}{N} + \frac{1}{N^3}$. We will conclude our paper in Section VI.

TABLE II
IV AND ZIG-ZAG SCRAMBLING

```
procedure MIX IV(S, K, V)
    i ← N/2 − 1;
    while i ≠ 0 do
        j ← (j + S[i]) ⊕ (K[i]+
        V[i]);
        SWAP(S[i], S[j]);
        i ← i − 1;
    end while
    i ← N/2;
    while i ≠ N do
        j ← (j + S[i]) ⊕ (K[i]+
        V[i]);
        SWAP(S[i], S[j]);
        i ← i + 1;
    end while
end procedure
```

```
procedure ZIG-ZAG(S, K)
    y ← 0;
    while y ≠ N do
        if y ≡ 0  mod 2 then
            i = y/2;
        else
            N/2 − (y+1)/2;
        end if
        j ← (j + S[i] + K[i]);
        SWAP(S[i], S[j]);
        y ← y + 1;
    end while
end procedure
```

II. Description of the RC4+ Stream Cipher

As in case of RC4, RC4+ too consists of a permutation S of $N = 256$ elements. The elements of S comes from the integer ring $\mathbb{Z}_{256}$. It also consists of two index pointers i and j. Each of these pointers has the size of 1 byte. RC4+ has a three-layer key scheduling algorithm. The initialization part and the basic scrambling in the first layer is similar to the KSA of RC4. Table I describes the initialization and first layer basic scrambling in KSA where S is initialized to the identity permutation and mixed with the Secret Key K of size l bytes where typically $l = 16$. All the addition operations are performed in $\mathbb{Z}_{256}$ and $\oplus$ denotes the bitwise XOR. In the second layer of KSA, the permutation S is further scrambled using an IV of size l bytes. Finally in the third layer, a zig-zag scrambling is performed on the permutation S. The description of second and third layer of KSA is given in Table II. The array V used in the table is of length N and is defined as

$$V[i] = \begin{cases} IV[\frac{N}{2} - 1 - i] & \text{if } \frac{N}{2} - l \leq i \leq \frac{N}{2} - 1 \\ IV[i - \frac{N}{2}] & \text{if } \frac{N}{2} \leq i \leq \frac{N}{2} + l - 1 \\ 0 & \text{otherwise} \end{cases}$$

The PRGA routine of RC4+ has a slight deviation from the simplistic PRGA structure of RC4. The designers proposed a bit different PRGA in order to protect the cipher against the strong second output bias showed in [3] and the permutation recovery attack of [4]. To protect the cipher design against the well known aforementioned attacks, the designers choose to make the output keystream byte functions of a few other locations of the permutation array S. Table III provides the exact details of the PRGA routine of RC4+ where $\ll$ and $\gg$ represents the left and right bitwise shifts respectively. The term p denotes the design parameter which can take the values from $\{0, 1, \cdots, 255\}$. Note that the pad p used in the PRGA by the designers of the RC4+ encryption scheme is `0xAA`.

TABLE III
RC4+ PRGA ROUTINE

```
procedure GENERATE KEYSTREAM(PERMUTATION S)
    i ← 0; j ← 0;
    while Keystream is required do
        i ← i + 1;
        j ← j + S[i];
        SWAP(S[i], S[j]);

        t ← S[i] + S[j];
        t' ← (S[i ≫ 3 ⊕ j ≪ 5] + S[i ≪ 5 ⊕ j ≫ 3]) ⊕ p;
        t'' ← j + S[j];

        Z_i = (S[t] + S[t']) ⊕ S[t''];
    end while
end procedure
```

III. Description of bias for odd pads $p \neq$ `0x03`

In this section we will show that RC4+ is not secure even if the design parameter p is changed to any fixed 8-bit odd constant except `0x03`. We will show that the first output byte Z_1 is still biased positively towards 1. With the help of the following theorems, we will prove that $\Pr(Z_1 = 1) = \frac{1}{N} + \frac{1}{2N^2}$. Let S_0 denote the initial state of the PRGA of RC4+.

Theorem 1. *Let S_0 be a random permutation on the set $\{0, 1, \ldots, 255\}$. Furthermore, let the pad used in the PRGA is fixed to some odd constant denoted by p where p ≠ 0x03. If $S_0[1] = 1$ and $S_0[2]$ is even, then the first output byte Z_1 can take the value 1 for exactly two values of $S_0[32]$.*

Proof: According to the description of the PRGA given in Table III, initially $i = j = 0$. After the increment operations take place, the new i, j values change as $i = 0 + 1 = 1$ and $j = 0 + S_0[i] = 0 + 1 = 1$. Since both the indices i and j are equal after the increment operations, the following swap operation doesn't bring any change to the array S_0. Now the calculation for t, t' and t'' are done as follows.

$$t = S_0[i] + S_0[j] = S_0[1] + S_0[1] = 2. \quad (1)$$

$$\begin{aligned} t' &= (S_0[i \gg 3 \oplus j \ll 5] + S_0[i \ll 5 \oplus j \gg 3]) \oplus p. \\ &= (S_0[1 \gg 3 \oplus 1 \ll 5] + S_0[1 \ll 5 \oplus 1 \gg 3]) \oplus p. \\ &= 2 \cdot S_0[32] \oplus p. \end{aligned} \quad (2)$$

$$t'' = j + S_0[j] = 1 + S_0[1] = 2. \quad (3)$$

Finally we have

$$Z_1 = (S_0[2] + S_0[t']) \oplus S_0[2]. \quad (4)$$

Let $\mathbb{E}_{256}$ be the set of even numbers in $\mathbb{Z}_{256}$. Consider the function $f : \mathbb{E}_{256} \times \mathbb{Z}_{256} \to \mathbb{Z}_{256}$ defined as $f(x, y) = (x + y) \oplus x$. By constructing a truth table for f it can be easily verified that $f = 1$ if and only if $y = 1$. This tells us that for Z_1 to be equal to 1, we must have $S_0[t'] = 1$, and since $S_0[1]$ has already been fixed to 1, this requires that $t' = 1$.

Since $p \neq \texttt{0x03}$ is an odd 8-bit constant, let $p = 2k + 1$, where $k \in \mathbb{Z}_{256} \setminus \{\texttt{0x01}\}$. Since S_0 is a permutation on $\mathbb{Z}_{256}$, and coupled with the fact that $S_0[1] = 1$ and $S_0[2]$ is even, it can be deduced that $S_0[32]$ can take total of 254 values out of which 127 are odd and 127 are even. We need

$$\begin{aligned} t' = 2 \cdot S_0[32] \oplus (2k+1) = 1 &\Leftrightarrow 2 \cdot S_0[32] = (2k+1) \oplus 1 \\ &\Leftrightarrow 2 \cdot S_0[32] = 2k \end{aligned} \tag{5}$$

Thus it can be seen that for the two values of $S_0[32]$ equal to k and $128+k$, t' evaluates to 1. Hence it follows that Z_1 can take the value 1 for exactly two values of $S_0[32]$. Since $Z_1 = 1$ for 2 out of 254 values $S_0[32]$ can take, $\Pr[Z_1 = 1|E] = \frac{2}{254} \approx \frac{2}{N}$, where the event E denotes the event "$S_0[1] = 1$ and $S_0[2]$ is even".

■

Theorem 2. *Let S_0 be a random permutation on the set $\{0, 1, \ldots, 255\}$. Furthermore, let the pad used in the PRGA is fixed to some odd constant denoted by p where $p \neq 3$. The probability that $Z_1 = 1$ is given as $Pr[Z_1 = 1] = \frac{1}{N} + \frac{1}{2N^2}$.*

Proof: Let the event E denote: "$S_0[1] = 1$ and $S_0[2]$ is even". Then we have $\Pr[E] = \frac{\frac{N}{2} \cdot (N-2)!}{N!} \approx \frac{1}{2N}$. From Theorem 1, we have $\Pr[Z_1 = 1|E] \approx \frac{2}{N}$. We have $\Pr[Z_1 = 1|E^c] = \frac{1}{N}$ (verified experimentally on 2^{20} random permutations) by following the standard randomness assumptions. Therefore the final probability comes down to

$$\begin{aligned} Pr[Z_1 = 1] &= Pr[Z_1 = 1|E] \cdot Pr[E] \\ &\quad + Pr[Z_1 = 1|E^c] \cdot Pr[E^c] \\ &= \frac{2}{N} \cdot \frac{1}{2N} + \frac{1}{N} \cdot (1 - \frac{1}{2N}) \\ &= \frac{1}{N} + \frac{1}{2N^2}. \end{aligned} \tag{6}$$

■

The following Theorem 3 from [3] indicates the number of output samples required to reliably distinguish two distributions X and Y. It is stated as follows.

Theorem 3. *X and Y being two distributions, if the probability of occurrence of the event υ in the distributions X and Y is p_1 and $p_1(1 + p_2)$ respectively, then for small p_1 and p_2, $\mathcal{O}(\frac{1}{p_1 \cdot p_2^2})$ samples are sufficient for distinguishing X from Y with a constant probability of success.*

Distinguishing RC4+ from Random Sources: Using Theorem 3, let X be the probability distribution of Z_1 in an ideal random stream, and Y be the probability distribution of Z_1 in the streams produced by RC4+. Let the event υ denote $Z_1 = 1$. The probability of occurrence of the event υ in the distribution X is $\frac{1}{N}$ and in Y is $\frac{1}{N} + \frac{1}{2N^2} = \frac{1}{N}(1 + \frac{1}{2N})$. Then we have $p_1 = \frac{1}{N}$ and $p_2 = \frac{1}{2N}$. Therefore the number of output samples required to reliably distinguish the two distributions is about $\frac{1}{p_1 \cdot p_2^2} = N \cdot 2^2 \cdot N^2 = 2^{26}$.

IV. Description of the bias when pad is 0x03

In this section we analyze the security of the RC4+ stream cipher if the pad used as the design parameter is fixed to $p = \texttt{0x03}$. The following Theorem 4 shows that first output byte Z_1 of RC4+ is not biased towards 1 for the event E described in Theorems 1 and 2 in case when the pad used as the design parameter of RC4+ is set to 0x03.

Theorem 4. *Let S_0 be a random permutation on the set $\{0, 1, \ldots, 255\}$. Furthermore, let the pad p used in the PRGA is fixed to $p = \mathit{0x03}$. If the event E denotes $S_0[1] = 1$ and $S_0[2]$ is even, then*

1) *The first output byte Z_1 can take the value 1 for only one value of $S_0[32]$.*
2) *The probability that $Z_1 = 1$ is $Pr[Z_1 = 1] = \frac{1}{N}$.*

Proof:

1) According to Theorem 1,

$$t' = 2 \cdot S_0[32] \oplus \texttt{0x03} = 1. \tag{7}$$

This can hold for $S_0[32] = 1$ and $S_0[32] = 129$. Since S_0 is injective, $S_0[32] \neq 1$. This implies that $t' = 1$ only if $S_0[32] = 129$.

2) The final probability of Z_1 being equal to 1 is given as follows

$$\begin{aligned} Pr[Z_1 = 1] &= Pr[Z_1 = 1|E] \cdot Pr[E] \\ &\quad + Pr[Z_1 = 1|E^c] \cdot Pr[E^c] \\ &= \frac{1}{N} \cdot \frac{1}{2N} + \frac{1}{N} \cdot (1 - \frac{1}{2N}) \\ &= \frac{1}{N}. \end{aligned} \tag{8}$$

■

In Theorem 4 we showed that the first output byte Z_1 of RC4+ is not biased towards 1 when pad $p = \texttt{0x03}$. For all the other odd pads p other than 0x03, we proved with the help of Theorems 1 and 2, that the first output byte Z_1 is positively biased. However, Z_1 remains free of any biases in case of pad being equal to 0x03. This provides us with the motivation to investigate biases in the subsequent output bytes. With the help of the following theorems we will show that the second output byte Z_2 is negatively biased towards 0 and 2 for pad $p = \texttt{0x03}$.

Theorem 5. *Let S_0 be a random permutation on the set $\{0, 1, \ldots, 255\}$. Let the pad used in the PRGA is fixed to $p = \mathit{0x03}$. If $S_0[1] = 0$ and $S_0[2] = 2$, then the second output byte Z_2 can never take the value 0. Furthermore, if $S_0[4] \equiv 0 \mod 4$ or $S_0[4] \equiv 1 \mod 4$, then Z_2 can never take the value 2.*

Proof: We refer to the PRGA routine shown in Table III of RC4+. The index bytes i and j are set to 0 initially. Suppose $S_0[0] = e$, where e is any value other than 0 and 2. In the first round of PRGA routine, i and j are incremented as follows

$$i = 0 + 1 = 1. \tag{9}$$

$$j = 0 + S_0[i] = 0 + S_0[1] = 0 + 0 = 0. \tag{10}$$

The swap operation of the PRGA makes $S_0[0] = 0$, $S_0[1] = e$ and $S_0[2] = 2$. In the second round of PRGA, the index values i and j change as follows,

$$i = 1 + 1 = 2. \tag{11}$$

$$j = 0 + S_0[2] = 0 + 2 = 2. \tag{12}$$

Now the subsequent swap operation doesn't change the values of $S_0[0]$, $S_0[1]$ and $S_0[2]$. In the next operations, t, t' and t'' are updated as follows

$$t = S_0[i] + S_0[j] = 2 \cdot S_0[2] = 4. \tag{13}$$

$$\begin{aligned} t' &= (S_0[i \gg 3 \oplus j \ll 5] + S_0[i \ll 5 \oplus j \gg 3]) \oplus \texttt{0x03}. \\ &= (S_0[2 \gg 3 \oplus 2 \ll 5] + S_0[2 \ll 5 \oplus 2 \gg 3]) \oplus \texttt{0x03}. \\ &= 2 \cdot S_0[64] \oplus \texttt{0x03}. \end{aligned} \tag{14}$$

$$t'' = j + S_0[j] = 2 + 2 = 4. \tag{15}$$

Now we have

$$Z_2 = (S_0[4] + S_0[t']) \oplus S_0[4]. \tag{16}$$

Suppose $Z_2 = 0$, then

$$(S_0[4] + S_0[t']) \oplus S_0[4] = 0 \implies S_0[t'] = 0. \tag{17}$$

Since S_0 is a permutation, hence injective, we have $S_0[t'] = S_0[0] = 0$. This implies $t' = 0$. Therefore

$$2 \cdot S_0[64] \oplus \texttt{0x03} = 0. \tag{18}$$

Now in the above equation, the L.H.S. can never be 0. This gives rise to a contradiction. Hence Z_2 can never take the value 0.

Now suppose $Z_2 = 2$, then

$$(S_0[4] + S_0[t']) \oplus S_0[4] = 2. \tag{19}$$

Furthermore, assume $S_0[4] \equiv 0 \mod 4$ or $S_0[4] \equiv 1 \mod 4$, then,

$$S_0[4] + S_0[t'] = S_0[4] + 2 \implies S_0[t'] = 2. \tag{20}$$

Since $S_0[2] = 2$, $S_0[t'] = S_0[2] = 2$. It implies $t' = 2$. Therefore,

$$2 \cdot S_0[64] \oplus \texttt{0x03} = 2. \tag{21}$$

In the above equation, L.H.S. can never be 2. Hence a contradiction and Z_2 can never take the value 2.

∎

Theorem 6. *Let S_0 be a random permutation on the set $\{0, 1, \ldots, 255\}$. Then following the results of Theorem 5, the probabilities of $Z_2 = 0$ and $Z_2 = 2$ are given as $\frac{1}{N} - \frac{1}{N^3}$ and $\frac{1}{N} - \frac{1}{2N^3}$.*

Proof: Let E_1 denote the event "$S_0[1] = 0$ and $S_0[2] = 2$". The probability of event E_1 can be given as $\Pr[E_1] = \frac{(N-2)!}{N!} \approx \frac{1}{N^2}$. From Theorem 5, we have $\Pr[Z_2 = 0 | E_1] = 0$. By standard randomness assumptions, the probability $\Pr[Z_2 = 0 | E_1^c] = \frac{1}{N}$ (verified by computer experiments using 2^{20} random keys). Therefore the final probability is given as

$$\begin{aligned} Pr[Z_2 = 0] &= Pr[Z_2 = 0 | E_1] \cdot Pr[E_1] \\ &\quad + Pr[Z_2 = 0 | E_1^c] \cdot Pr[E_1^c] \\ &= 0 \cdot \frac{1}{N^2} + \frac{1}{N} \cdot (1 - \frac{1}{N^2}) \\ &= \frac{1}{N} - \frac{1}{N^3}. \end{aligned} \tag{22}$$

Let E_2 denote the event "$S_0[1] = 0$, $S_0[2] = 2$ and $S_0[4] \equiv 0 \mod 4$ or $S_0[4] \equiv 1 \mod 4$". Then $\Pr[E_2] = \frac{(\frac{N}{2}-1) \cdot (N-3)!}{N!} \approx \frac{1}{2N^2}$. From Theorem 5, we have $\Pr[Z_2 = 2 | E_2] = 0$. The probability $\Pr[Z_2 = 2 | E_2^c] = \frac{1}{N}$ (verified by computer experiments using 2^{20} random keys). Then

$$\begin{aligned} Pr[Z_2 = 2] &= Pr[Z_2 = 2 | E_2] \cdot Pr[E_2] \\ &\quad + Pr[Z_2 = 2 | E_2^c] \cdot Pr[E_2^c] \\ &= 0 \cdot \frac{1}{2N^2} + \frac{1}{N} \cdot (1 - \frac{1}{2N^2}) \\ &= \frac{1}{N} - \frac{1}{2N^3}. \end{aligned} \tag{23}$$

∎

Distinguishing RC4+ from Random Sources(pad `0x03`): Using Theorem 3, let X be the probability distribution of Z_2 in an ideal random stream, and Y be the probability distribution of Z_2 in the streams produced by RC4+. Let the events υ_1 and υ_2 denote $Z_2 = 0$ and $Z_2 = 2$. The probability of occurrence of the event υ_1 in the distribution X is $\frac{1}{N}$ and in Y is $\frac{1}{N} - \frac{1}{N^3} = \frac{1}{N}(1 - \frac{1}{N^2})$. Then we have $p_1 = \frac{1}{N}$ and $p_2 = -\frac{1}{N^2}$. Therefore the number of output samples required to reliably distinguish the two distributions is about $\frac{1}{p_1 \cdot p_2^2} = N \cdot N^4 = N^5 = 2^{40}$. Similarly, the number of output samples required to reliably distinguish the two distributions for the event υ_2 is 2^{42}.

V. Biases in the Original Cipher

Dropping the first output byte could easily settle down the insecurity issue based on the attack mentioned in [2] on the first output byte of RC4+. In this section we will prove that the second output byte Z_2 produced in the PRGA of RC4+ is positively biased towards 0 when the pad used in the design is 0xAA. The probability of Z_2 being equal to 0 is $\frac{1}{N} + \frac{1}{N^3}$.

Theorem 7. *Let S_0 be a random permutation on the set $\{0, 1, \ldots, 255\}$. Let the pad used in the PRGA is fixed to $p =$ 0xAA. If $S_0[1] = 0$ and $S_0[2] = 2$, then the second output byte Z_2 can take the value 0 for 2 values of $S_0[64]$.*

Proof: In the first iteration of the PRGA given in Table III, initially $i = j = 0$. Thereafter, the values of i and j are updated as $i = 0 + 1 = 1$ and $j = 0 + S_0[i] = S_0[1] = 0$. Let $S_0[0] = e$, where e can be anything except 0 and 2. After the following swap operation, $S_0[0] = 0$ and $S_0[1] = e$. In the second round of the PRGA, i and j are updated as follows,

$$i = 1 + 1 = 2. \tag{24}$$

$$j = 0 + S_0[2] = 0 + 2 = 2. \tag{25}$$

Therefore no swapping takes place in the subsequent swap operation. In the next steps, t, t' and t'' are updated as follows,

$$t = S_0[i] + S_0[j] = 2 \cdot S_0[2] = 4. \tag{26}$$

$$\begin{aligned} t' &= (S_0[i \gg 3 \oplus j \ll 5] + S_0[i \ll 5 \oplus j \gg 3]) \oplus \texttt{0xAA}. \\ &= (S_0[2 \gg 3 \oplus 2 \ll 5] + S_0[2 \ll 5 \oplus 2 \gg 3]) \oplus \texttt{0xAA}. \\ &= 2 \cdot S_0[64] \oplus \texttt{0xAA}. \end{aligned} \tag{27}$$

$$t'' = j + S_0[j] = 2 + 2 = 4. \tag{28}$$

Now if $Z_2 = 0$, then

$$(S_0[4] + S_0[t']) \oplus S_0[4] = 0 \implies S_0[t'] = 0. \tag{29}$$

The injective property of the permutation S_0 implies $S_0[t'] = S_0[0] = 0$. This implies $t' = 0$. Therefore,

$$2 \cdot S_0[64] \oplus \texttt{0xAA} = 0. \tag{30}$$

It is evident in the above equation that for exactly 2 values of $S_0[64]$, i.e. 85 and 213, t' will evaluate to 0. Furthermore, if the event "$S_0[1] = 0$ and $S_0[2] = 2$" is denoted by E_1, then $Pr[Z_2 = 0|E_1] \approx \frac{2}{N}$. ■

Theorem 8. *Let S_0 be a random permutation on the set $\{0, 1, \ldots, 255\}$. The probability of $Z_2 = 0$ is given as $\frac{1}{N} + \frac{1}{N^3}$.*

Proof: Let E_1 denote the event "$S_0[1] = 0$ and $S_0[2] = 2$". The probability of event E_1 can be given as $\Pr[E_1] = \frac{(N-2)!}{N!} \approx \frac{1}{N^2}$. From Theorem 7, we have $\Pr[Z_2 = 0|E_1] = \frac{2}{N}$. By standard randomness assumptions, the probability $\Pr[Z_2 = 0|E_1^c] = \frac{1}{N}$ (verified by computer experiments using 2^{20} random keys). Therefore the final probability is given as

$$\begin{aligned} Pr[Z_2 = 0] &= Pr[Z_2 = 0|E_1] \cdot Pr[E_1] \\ &\quad + Pr[Z_2 = 0|E_1^c] \cdot Pr[E_1^c] \\ &= \frac{2}{N} \cdot \frac{1}{N^2} + \frac{1}{N} \cdot (1 - \frac{1}{N^2}) \\ &= \frac{1}{N} + \frac{1}{N^3}. \end{aligned} \tag{31}$$

■

According to Theorem 3 and following the results of Theorems 7 and 8, we conclude that around $N \cdot (N^2)^2 = N^5 = 2^{40}$ output samples are sufficient to reliably mount a distinguishing attack on RC4+ based on the bias present in the second output byte Z_2.

Based on the results shown in our paper and [2], it is evident that in the practical implementations of RC4+, discarding the use of the first and the second output bytes i.e. Z_1 and Z_2, is necessary. Future works in this direction includes the investigations of biases present in subsequent output bytes. In the light of the results discussed in this paper, the pad $p = \texttt{0x03}$ seems to make the design most resistant to distinguishing attacks. Therefore, based on the results and scenarios presented by us, we think that the safest use of the cipher is from the 3$^{\text{rd}}$ byte onwards with $p = \texttt{0x03}$.

VI. Conclusion

In this paper we focus on the security of the stream cipher RC4+ against distinguishing attacks based on the biases of its output bytes. As the bias present in the first output byte Z_1 of this stream cipher proved in [2], we prove that the second output byte Z_2 is as well biased positively towards 0. In addition, we also analyze the security of the cipher if odd pads are used as the design parameter other than the pad 0xAA used in the original cipher proposed in [1]. We show that the cipher is still vulnerable to distinguishing attacks if the pads are changed into an 8-bit constant odd value. In our analysis, we find that the stream cipher RC4+ provides maximum resistance to distinguishing attacks if the pad used as the design parameter is made equal to 0x03.

References

[1] S. Maitra and G. Paul. *Analysis of RC4 and Proposal of Additional Layers for Better Security Margin*. In INDOCRYPT 2008, LNCS, Vol. 5365, pp. 27-39.

[2] S, Banik, S, Sarkar and R. Kacker. *Security Analysis of RC4+ Stream Cipher*. In INDOCRYPT 2013, LNCS, Vol. 8250, pp. 297-307.

[3] I. Mantin and A. Shamir. *A Practical Attack on Broadcast RC4*. In FSE 2001, LNCS, Vol. 2355, pp. 152-164.

[4] A. Maximov and D. Khovratovich. *New State Recovery Attack on RC4*. In CRYPTO 2008, LNCS, Vol. 5157, pp. 297-316.

[5] G. Gong, K.C. Gupta, M. Hell and Y.Nawaz. *Towards a General RC4-like Keystream Generator*. In CISC 2005, LNCS, Vol. 3822, pp. 162-174.

[6] S. Paul and B. Preneel. *A New Weakness in the RC4 Keystream Generator and an Approach to Improve the Security of the Cipher*. In FSE 2004, LNCS, Vol. 3017, pp. 245-259.

[7] B. Zoltak. *VMPC One-Way Function and Stream Cipher*. In FSE 2004, LNCS, Vol. 3017, pp. 210-225.

[8] A. Maximov. *Two Linear Distinguishing Attacks on VMPC and RC4A and Weakness of RC4 Family of Stream Ciphers*. In FSE 2005, LNCS, Vol. 3557, pp. 342-358.

[9] S. Paul and B. Preneel. *On the (In)security of Stream Ciphers Based on Arrays and Modular Addition*. In ASIACRYPT 2006, LNCS, Vol. 4284, pp. 69-83.

[10] Y. Tsunoo, T. Saito, H. Kubo, M. Shigeri, T. Suzaki and T. Kawabata. *The Most Efficient Distinguishing Attack on VMPC and RC4A*. In SKEW 2005. Available at http://www.ecrypt.eu.org/stream/papers.html.