# Predicting and Distinguishing Attacks on RC4 Keystream Generator

Itsik Mantin

NDS Technologies, Israel
imantin@nds.com

**Abstract.** In this paper we analyze the statistical distribution of the keystream generator used by the stream ciphers RC4 and RC4A. Our first result is the discovery of statistical biases of the digraphs distribution of RC4/RC4A generated streams, where digraphs tend to repeat with short gaps between them. We show how an attacker can use these biased patterns to distinguish RC4 keystreams of $2^{26}$ bytes and RC4A keystreams of $2^{26.5}$ bytes from randomness with success rate of more than 2/3. Our second result is the discovery of a family of patterns in RC4 keystreams whose probabilities in RC4 keystreams are several times their probabilities in random streams. These patterns can be used to predict bits and words of RC4 with arbitrary advantage, e.g., after $2^{45}$ output words a single bit can be predicted with probability of 85%, and after $2^{50}$ output words a single byte can be predicted with probability of 82%, contradicting the unpredictability property of PRNGs.

**Keywords**: RC4, Stream ciphers, Cryptanalysis, Distinguishing attacks, Predicting attacks.

## 1 Introduction

RC4 is the most widely used stream cipher in software applications. Among numerous applications it is used to protect Internet traffic as part of the SSL and is integrated into Microsoft Windows. It was designed by Ron Rivest in 1987 and kept as a trade secret until it leaked out in 1994. RC4 has a secret internal state which is a permutation of all the $N = 2^n$ possible $n$ bits words, associated with two indices in it, when in practical applications $n = 8$, and thus RC4 has a huge state of $log_2(2^8! \times (2^8)^2) \approx 1700$ bits.

In this paper we explore several classes of RC4 states, and analyze their statistical properties and cryptanalytic significance. RC4 was already proven to contain many patterns with unique statistical behavior and many correlations between the output words and the internal state. The main innovation of our work is in moving the focus of RC4 analysis from consecutive sequences of rounds to non-consecutive ones. We show classes of RC4 partial states that cause unique behavior of the output stream, when the unique patterns are in correlations between output words in distant rounds.

Our first result is based on analysis of RC4 1-states (partial states are defined later), which cause digraphs to repeat with short gaps in RC4 output stream, e.g., $ABAB$, $ABCAB$, $ABCDAB$, etc, when the relative bias (the ratio between the additional probability and the original probability) of these patterns is approximately $1/N$ for a zero gap ($ABAB$) and gradually decreases when the gap length increases. We show how these patterns can be used to mount a distinguisher of RC4 streams from randomness that requires only $2^{26}$ output words, about a third of the data needed by the best known distinguisher from [4] for the same success rate of 2/3. In addition, we show that these patterns appear also in RC4A with the same biased probability as in RC4, and describe a slightly less efficient distinguishing algorithm for RC4A.

Our second result is based on new analysis of RC4 predictive states, which are partial states (usually small) that suffice for determining keystream output for several rounds. We define a recyclability property for these states and show that recyclable predictive states have relatively high probability to repeat in shifts of $N$ rounds, creating the same predicted output pattern repetitively. We prove that some of the known predictive states are indeed recyclable and use these observations to extend the significance of every recyclable predictive state from a single short biased pattern to a family of patterns that occur with various probabilities and with various relative biases. The probabilities are lower than the one of the original pattern, whereas the relative biases are significantly larger than the one of the original pattern and can grow to arbitrary values, allowing the attacker to predict output bits with arbitrary advantage. In addition, we discuss how the recyclable states can be used to construct state recovery attacks on RC4 internal state.

The rest of the paper is organized in the following way: In section 2 we describe RC4 and previous results about its security. In Section 3 we present the digraphs repetition pattern, analyze its statistical properties in RC4/RC4A generated streams and discuss how the cryptanalyst can exploit these properties. In Section 4 we define recyclable states and discuss their availability and their cryptanalytic usability. We summarize our work in Section 5.

## 2 RC4 and Its Security

### 2.1 Description of RC4

RC4 consists of 2 parts (described in Figure 1): A key scheduling algorithm KSA which turns a random key (whose typical size is 40-256 bits) into an initial permutation $S$ of $\{0, \ldots, N-1\}$, and an output generation part PRGA which uses this permutation to generate a pseudo-random output sequence.

The PRGA initializes two indices $i$ and $j$ to 0, and then loops over four simple operations which increment $i$ as a counter, increment $j$ pseudo randomly, exchange the two values of $S$ pointed to by $i$ and $j$, and output the value of $S$ pointed to by $S[i] + S[j]$[1].

[1] Here and in the rest of the paper all the additions are carried out modulo $N$

| KSA($K[0 \ldots \ell-1]$) | PRGA($K$) |
|---|---|
| Initialization: | Initialization: |
| For $i = 0 \ldots N-1$ | $i = 0$ |
| $S[i] = i$ | $j = 0$ |
| $j = 0$ | $S = KSA(K)$ |
| Scrambling: | Generation loop: |
| For $i = 0 \ldots N-1$ | $i = i + 1$ |
| $j = j + S[i] + K[i \bmod \ell]$ | $j = j + S[i]$ |
| $Swap(S[i], S[j])$ | Swap($S[i], S[j]$) |
| | Output $z = S[S[i] + S[j]]$ |

**Fig. 1.** The Key Scheduling Algorithm and the Pseudo-Random Generation Algorithm

### 2.2 Previous Attacks on RC4

Cryptanalysis of RC4 is divided into two main parts, analysis of the initialization of RC4 and analysis of the keystream generation. The first part focuses on the KSA, the PRGA initialization and the integration of both, whereas the last focuses on the internal state and the round operation of the PRGA.

The simplicity of the initialization part and the major difference between the amount of hidden information between this part and the keystream generation part attracted a lot of attention in the cryptographic community and indeed numerous significant weaknesses were discovered in this part of many types, including classes of weak keys ([18]), patterns that appear twice and three times the expected probability ([5]), propagation of key patterns through the KSA to the initial permutation and through the PRGA initialization to the prefix of the stream ([6]), modes of operation that allow related key attacks ([17]), partial message recovery ([5]) and full key recovery attacks ([6]) with practical time complexities, statistical biases in different prefixes of the generated stream ([6] and [9]) and analysis of the biased distribution of RC4 initial permutation ([7] and [16]).

The weaknesses that were discovered in [6] where the most distructive ones, as they were translated to practical attack on the usage of RC4 in the security protocols (WEP) of the international standard for wireless LAN communication 802.11b. The discovery of this attack affected the trust of cryptographers and security designers in RC4 and the common practice for using RC4 today includes hardening of the initialization process by truncating some prefix of the keystream (RSA and Ron Rivest recommendation is $N$ words). This hardening neutralizes most of the attacks and weaknesses that were discovered in RC4 initialization.

The analysis of the keystream generation part was far less successful in mounting severe attacks, but still several interesting weaknesses where discovered. Golić ([1]) and Fluhrer and McGrew ([4]) designed distinguishers of RC4 streams from random streams that require $2^{44.7}$ and $2^{30.6}$ keystream words respectively. Several classes of RC4 partial states were defined and analyzed in [4], [5] and [8] as such that create unique patterns in the output stream and allow a viewer of the output stream to recover parts of the internal state with more than trivial probability (chapter 2 of [16] contains an overview of these classes). The cycles structure of RC4 state progression was also analyzed in [3] and [15], where the last describes short cycles that are unreachable by RC4. [2] and [3] describe state recovery attacks with complexity that is less than the square root of an exhaustive search over all possible states. However, due to the hugeness of the state (1700 bits for $n = 8$), these attacks are completely impractical as they require more than $2^{700}$ steps.

Two variants of RC4 were recently proposed, both slightly more complex than the original RC4 and are claimed to be more secure than it. RC4A ([9]) was designed by Paul and Preneel and works with two RC4 tables (we describe the algorithm in more details in Section 3.4). The generation stage of RC4A is slightly more efficient than RC4's, but the initialization stage requires at least twice the effort of RC4 initialization. VMPC ([10]) was designed by Zoltak and includes several changes to the initialization ($j$ not initialized to 0), the round operation (different progression for $j$) and the output generation stage (different calculation of the output index). Maximov described in [13] distinguishers for both variants, requiring $2^{54}$ data for VMPC and $2^{58}$ data for RC4A.

The trend of side-channel attacks had not skipped RC4 and efficient fault attacks were described in [11] and [12].

## 3 The Digraph Repetition Bias

In this section we describe a special scenario that occurs when the value 1 is used to update the index $j$, analyze the expression of this scenario on the statistical behavior of RC4 output streams and exploit these observations to mount a distinguishing attack on RC4.
We use the notations $i_t, j_t$ and $S_t$ for the indices $i$ and $j$ and the permutation $S$ after round $t$.

### 3.1 The Queue Model

The behavior of RC4 permutation was described in [16] as a unique queue with some interesting properties. The queue has $N$ ordered elements (permutation elements) and when one reaches its turn, it is used to update the index $j$ through the function $j \leftarrow j + S[i]$. However, instead of going to the end of the line, the updating element selects a pseudo-random location in the queue (pointed to by $j$), pushes itself to this location and sends the deprived element that was there to the end of the line. One of the properties of this special queue is that values that were recently used (to update $j$) are likely to be used again in less than $N$ rounds, whereas values that were not used for $N$ rounds must have been pushed back at least once and thus have lower probability to be used.

The permutation itself changes in two locations within every round and in $2k$ locations (possibly with repetitions) within sequences of $k$ rounds due to the swapping, where half of these changes are in predicted locations and the other half are in pseudo-random locations. Lemma 1 measures the effect of these changes on the permutation entries.

**Lemma 1.** *Let $\mathcal{I}$ be a set of $r$ permutation locations. Suppose that RC4 is in a state where the predictable course of the index $i$ in the next $k$ rounds does not visit $\mathcal{I}$. Then the probability of the permutation $k$ rounds later to have the same values in $\mathcal{I}$ is approximately $e^{-kr/N}$.*

*Proof.* The index $i$ does not reach any of the indices in $\mathcal{I}$ and the index $j$ progresses in a pseudo-random manner and will reach each of the the $r$ positions in each of the $k$ rounds with probability $1/N$. Thus failing in these $kr$ trials results with having the set $\mathcal{I}$ untouched. The probability of this event is $(1-1/N)^{kr} \approx e^{-kr/N}$. □

### 3.2 The Digraph Repetition Scenario

Consider the situation where the value 1 is used to update $j$, i.e. (in "queue terms"), 1 reaches the head of the line. We will denote the round where that happens as round $r$ and the pair of rounds $[r-1, r]$ as the *origin pair*. Notice that this situation occurs quite often, approximately once in every $N$ rounds. Suppose now that the pseudo-random location selected by 1 after the $j$ update ($j_r$) is not very far from the head of the line, making it unlikely that this element will be moved from there before being used again to update $j$. We denote the distant which 1 passes in the swap by the gap $g = j_r - i_r$. And lastly, suppose that when $i$ reaches the new location of 1, $j$ points exactly to the original location of 1. We denote this round and the previous one as the *end pair* $[r+g-1, r+g]$. Our observation is that when that happens, a unique equivalence occurs between the origin pair of rounds and the end pair of rounds, making it likely that they produce the same pair of output words. We formalize this observation in Lemma 2.

**Lemma 2.** *Suppose that $S_{r-1}[i_r] = 1$ and let $g = j_{r-1} - i_{r-1}$ (the gap). If $j_{r+g-1} = i_{r-1}$ then the probability of the digraph that is outputted in rounds $[r+g-1, r+g]$ to be identical to the digraph outputted in rounds $[r-1, r]$ is bounded from below by $e^{(8-8g)/N}$.*

*Proof.* We first prove that when everything "goes right" the digraphs are identical and then bound the probability that anything "goes wrong". In Figure 2 we track the internal state during rounds $r, \ldots, r+g$ and show that the same digraphs are outputted in the origin pair and the end pair.

This scenario assumes that the permutation values in locations that affect the swaps and the selection of output words in both pairs of rounds, remain unchanged during the whole process (except for the swaps in the origin and end pairs). The locations $i_{r-1}$, $i_r$, $i_{r+g-1}$ and $i_{r+g}$ cannot be reached by $i$ but can be changed by $j$ in any of the $g-2$ intermediating rounds. By using Lemma 1 we get that the probability of 4 elements to survive $g-2$ rounds is $e^{-4(g-2)/N}$. The locations of the output words $S_{r-1}[i_{r-1}] + S_{r-1}[i_{r+g-1}]$ and $S_r[i_r] + S_r[i_{r+g}]$ are arbitrary and may fall in the course of $i$ between the round pairs. Each of these entries need to remain unchanged for $g$ rounds (the first in rounds $r, \ldots, r+g-1$

| $i$ | $j$ | S | | | | | $S[i]$ | $S[j]$ | $S[i]+S[j]$ | *Output* |
|---|---|---|---|---|---|---|---|---|---|---|
| | | $i_r-1$ | $i_r$ | $\ldots$ | $i_r+g-1$ | $i_r+g$ | | | | |
| $i_r-2$ | $i_r-1+g-A$ | $A$ | 1 | $\ldots$ | $B$ | $C$ | / | / | / | / |
| $i_r-1$ | $i_r-1+g$ | $B$ | 1 | $\ldots$ | $A$ | $C$ | $B$ | $A$ | $B+A$ | $Z_1$ |
| $i_r$ | $i_r+g$ | $B$ | $C$ | $\ldots$ | $A$ | 1 | $C$ | 1 | $C+1$ | $Z_2$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\ddots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $i_r+g-2$ | $i_r-1-B$ | $B$ | $C$ | $\ldots$ | $A$ | 1 | / | / | / | / |
| $i_r+g-1$ | $i_r-1$ | $A$ | $C$ | $\ldots$ | $B$ | 1 | $B$ | $A$ | $B+A$ | $Z_1$ |
| $i_r+g$ | $i_r$ | $A$ | 1 | $\ldots$ | $B$ | $C$ | $C$ | 1 | $C+1$ | $Z_2$ |

**Fig. 2.** The Digraph Repetition Scenario

and the second in rounds $r+1,\ldots,r+g$) and has a probability of $g/N$ to fall in the $i$-affected area. Thus the probability of these two indices to remain in place is $(1-\frac{g}{N})^2 \cdot e^{-2g/N}$. For small $g/N$ we can use the approximation $e^{-\varepsilon} \approx 1-\varepsilon$ to get $e^{-4g/N}$ and thus the overall probability of all indices to be in place is $e^{(8-8g)/N}$. $\square$

**Theorem 1.** *For small values of $G$ the probability of the pattern $AB\mathcal{S}AB$ in RC4 streams where $S$ is a $G$-word string is* $(1+e^{(-4-8G)/N}/N)\cdot 1/N^2$.

*Proof.* Notice that $g$ from Lemma 2 actually represents the shift between the digraphs and the real gap between the digraphs is $G=g-2$.

Let $E_S$ (source event) be the event where the conditions of Lemma 2 are satisfied, i.e., $S_{r-1}[i_r]=1$, $j_{r-1}=i_{r+G+1}$ and $j_{r+G+1}=i_{r-1}$ and let $E_T$ (target event) be the event where there is equality between the output digraph of the origin pair and the end pair. The probability of $E_S$ is $N^{-3}$ and we use it to calculate the probability of $E_T$.

$$\begin{aligned}\mathbb{P}[E_T] &= \mathbb{P}[E_T|E_S]\cdot\mathbb{P}[E_S]+\mathbb{P}[E_T|\overline{E_S}]\cdot\mathbb{P}[\overline{E_S}]\\ &\geq e^{(-8-8G)/N}/N^3+1/N^2\cdot(1-1/N^3)\\ &= 1/N^2\cdot(1+e^{(-8-8G)/N}/N)+negl(1/N^2)\end{aligned}$$

$\square$

The relative bias is $b_{DBL}(G)=\frac{e^{-8/N}}{N}\cdot(e^{-8/N})^G = C_1\cdot C_2^G$ for the constants $C_2=e^{-8/N}\approx 0.97$ and $C_1=C_2/N$.

For the sake of simplicity the we made many heuristic assumptions during the analysis. Therefore, we carried out RC4 simulations to support the analysis and put the analytically calculated biases against the simulation result (see Figure 3)[2].

[2] In the simulation we used $2^{16}$ bitstreams of size $2^{24}$ each, ignoring cross-dependencies between the examined events and regarding every digraph with every gap as an independent event, meaning that every bitstream digraph was used for 64 experiments as the ending digraph and 64 experiments as the beginning digraph.

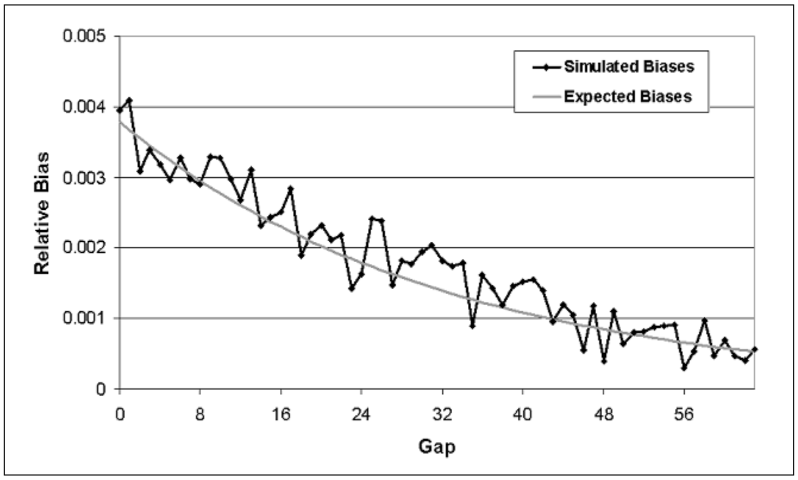


**Fig. 3.** Experimental vs. analytic results for biases of digraph repetitions.

### 3.3 Digraph Repetition Distinguisher

In [4] Fluhrer and McGrew used information theoretic means to estimate the number of samples that are required for their distinguisher. They showed that this amount is inversely proportional to the discrimination between these distributions. Lemma 3 estimates the discrimination between two distributions that differ in a set of independent events.

**Lemma 3.** *Let $\mathcal{X}$ and $\mathcal{Y}$ be two distributions and suppose that the independent events $\{E_i : 1 \le i \le k\}$ occur with probabilities $p_{\mathcal{X}}(E_i) = p_i$ in $\mathcal{X}$ and $p_{\mathcal{Y}}(E_i) = (1 + b_i)p_i$ in $\mathcal{Y}$. Then the discrimination of the distributions is $\sum_i p_i b_i^2$.*

We prove this lemma in Appendix A.1.

Let us calculate this sum for the biases of digraphs repetitions with different gaps of at most $M$. The probability in random distributions is $1/N^2$ for all events and the relative bias in RC4 streams is $b_{DBL}(G)$.

$$\sum_{0 \le G \le M} p \cdot b_{DBL}^2(G) = p \cdot \sum_{0 \le G \le M} (C_1 \cdot C_2^G)^2 = p\, C_1^2 \cdot \sum_{0 \le G \le M} (C_2^2)^G$$

this is a partial sum of a geometric sequence and for $M \ge N/4$ the tail of the sum is very small

$$= p\, C_1^2 \cdot \frac{1 - C_2^{2M}}{1 - C_2^2} \approx N^{-4} \cdot \frac{e^{-16/N}}{1 - e^{-16/N}} \approx \frac{N^{-4}}{1 - (1 - 16/N)} = \frac{1}{16N^3}$$

The biases used by Fluhrer's distinguisher are in $7N$ events that has trivial probability $1/N^3$ and relative bias of $1/N$, and when summing them according to Lemma 3 we get an induced discrimination of $\sum pb^2 = 7/N^4$. Thus for a given success rate, a distinguisher based on our observations solely, requires $102/N$ (less than half for $N = 256$) of the samples that are required by one that is based on Fluhrer's biases and a combined distinguisher requires $\frac{102}{N+102}$ (less than one third for $N = 256$) of this number.

**Lemma 4.** *The number of samples that is required for distinguishing two distributions that have discrimination $D$ with success rate $1 - \alpha$ (for both directions) is $(1/D) \cdot (1 - 2\alpha) \cdot log_2 \frac{1-\alpha}{\alpha}$*

We prove this lemma in Appendix A.2. We use lemma 4 to estimate the numbers of samples that are required for different success rates, and get that less than $2^{29}$ samples are required for success rate of 90%, less than $2^{28}$ samples are required for success rate of 80% and less than $2^{26}$ samples are required for success rate of 2/3.

### 3.4 Applicability to RC4A

We divide the RC4 round into three parts

1. The *State update* part includes the changes in $i$ and $j$ and the swap of the corresponding values in $S$.
2. The *Output Index Calculation* part is the calculation of $S[i] + S[j]$
3. The *Emission* part is the selection of the output value from the permutation.

RC4A uses two instances of RC4 tables with cross references between them. Every round comprises two semi-rounds, in each of which an output word is emitted, and thus every round results with two output words. During the initialization stage both tables are initialized to pseudo-random permutations (we omit detailed description of this part) and during the generation stage both tables are involved in the generation of every output word; In the first semi-round the *state update* and the *output index calculation* are done in one table, whereas the *emission* part is done in the other table, namely the output word is taken from the other table (in the the calculated index). In the second semi-round, the roles of the tables change and the output is taken from the first table. Notice that on every such round both RC4A tables progress identically to a standard RC4 table and thus the state progression is similar to RC4. We will use this property to show that the statistics of digraphs repetition in RC4A is similar to its statistics in RC4.

Before getting to the fine details, let us give some intuition for why the digraph repetition patterns are biased also in RC4A. Most of the weaknesses of RC4 keystream generation rely on certain scenarios that are known to occur during RC4 progression. We refer to these scenarios as *special*. Most of the special scenarios rely on a small number of values that appear in certain locations of the RC4 table, which are used during the scenario. Scenarios that are based on a large number of values are less frequent and thus in order to maximize the probability of occurrence, a typical special scenario minimizes the number of starting conditions (known entries) by re-using the known entries for all three parts of the round; *State update* is crucial for assuring the correct progression of the scenario. *Output index calculation* and *emission* are important for external expression of the scenario, sometimes resulting with biases in the keystream distribution. In RC4A the emission part is done in the second table and thus the same values cannot be used for both parts 1 and 3 or 2 and 3. Thus the expression of most of the special scenarios vanishes in RC4A and their induced weaknesses disappear.

However, the digraph repetition scenario is an exceptional one, since the permutation entries that guarantee the occurrence of the scenario are used only for *state update* and *output index calculation*. The expression of the scenario does not depend on any particular output value and thus the separation of the *emission* part does not affect the expression of the scenario. The output *indices* at close pairs of rounds are identical with high probability and the only requirement from the output values is to remain in the same locations during the intermediating rounds. This requirement is fully satisfied in RC4A since both tables evolve in the same rate as RC4 tables and therefore the digraph repetition scenario occurs in RC4A with similar statistics as in RC4.

The scenario in RC4A is as follows. We use permutations $P_1$ and $P_2$ and same symbols as in Figure 2. In addition, we ignore every second semi-round (which does not affect $P_1$).

1. Round $t$: The index $i$ reaches the position before the value 1 in $P_1$ ($i_r - 1$). Output of the first semi-round is the value from $P_2$ in location $B + A$ ($Z_1$).
2. Round $t + 1$: The index $i$ reaches the value of 1 in $P_1$ ($i_r$). Output of the first semi-round is the value from $P_2$ in location $C + 1$ ($Z_2$).
3. $g - 2$ rounds pass. Locations $B + A$ and $C + 1$ remain unchanged in $P_2$. Locations $t$, $t + 1$, $t + g$ and $t + g + 1$ remain unchanged in $P_1$.
4. Round $t + g$: The index $i$ reaches again the position before the value 1 in $P_1$ ($i_r + g - 1$). Output of the first semi-round is the value from $P_2$ in location $B + A$ (again $Z_1$). In the second semi-round some arbitrary value is emitted.
5. Round $t + g + 1$: The index $i$ reaches the value of 1 in $P_1$ ($i_r + g$). Output is the value from $P_2$ in location $C + 1$ (again $Z_2$).

Thus when omitting the even output words and concentrating in the sub-stream of odd words, or alternatively concentrating in sub-stream of even words) the same pattern $Z_1 Z_2 \underbrace{* \ldots *}_{g} Z_1 Z_2$ occurs with the same biased probability as in RC4 streams. When summing up the biases of these patterns in both sub-streams, we get a similar induced discrimination of $\frac{1}{16N^3}$.

**Corollary 1.** *There exists an algorithm that distinguishes RC4A streams from randomness with success probability of* $2/3$ *by analyzing less than* $2^{26.5}$ *samples.*

The difference between the distinguishers is due to the unavailability of Fluhrer's biases in RC4A. The best RC4A distinguisher that was described in the literature ([13]) requires $2^{59}$ output words, which is more than billion times the data for our distinguisher.

# 4 Recycling Attacks

## 4.1 Partial States and Fortuitous States

A significant part of published RC4 analysis is based on classification of RC4 states and partial states according to their cryptanalytic significance. We recall the definition of partial states and predictive states from [5].

**Definition 1.** *A $d$-state is a partially specified RC4 state, that includes $i$, $j$ and $d$ (not necessarily consecutive) elements of $S$.*

We use the notation $\{i_0, j_0, [x_0, x_1, \ldots, x_{d-1}]\}$ to specify the $d$-state $i = i_0$, $j = j_0$, $S[i_0 + 1, \ldots, i_0 + d] = [x_0, \ldots, x_{d-1}]$.

**Definition 2.** *Let $D$ be a $d$-state and suppose that for a positive $b$ there exist a sequence of $b$ words that are output (in certain shifts) by every RC4 state that is compliant with $D$. Then $D$ is said to be $b$-predictive.*

It was shown in [5] how $b$-predictive states can be used to attack RC4 by revealing parts of the internal state and constructing efficient distinguishing algorithms, in particular when $b = d$ (it was proved in [8] that $b \leq d$). A fortuitous state of order $d$ is a $d$-predictive $d$-state in which the predicted $d$ outputs are emitted immediately after the occurrence of the state. In Figure 4 we demonstrate predictiveness by presenting a family of $N - 1$ fortuitous states of order two, $\{x - 1, 0, [-1, x + 1]\}$ for every $x \neq 2$.

| Round | $i$ | $j$ | $S[x]$ | $S[x+1]$ | $S[i]$ | $S[j]$ | $S[i]+S[j]$ | *Output* |
|---|---|---|---|---|---|---|---|---|
| $x-1$ | $x-1$ | $x+2$ | $-1$ | $x+1$ | / | / | / | / |
| $x$ | $x$ | $x+1$ | $x+1$ | $-1$ | $x+1$ | $-1$ | $x$ | $x+1$ |
| $x+1$ | $x+1$ | $x$ | $-1$ | $x+1$ | $x+1$ | $-1$ | $x$ | $-1$ |

**Fig. 4.** A Family of Fortuitous States of order 2

## 4.2 Recyclable States

A very interesting property of the scenario presented in Figure 4 is the fact that the swap on the second round reverts the swap from the first round, leaving the permutation in the same situation as it was at the beginning of the scenario. We formalize this behavior by defining *recyclable* states and then discuss how the cryptanalyst can exploit this property.

**Definition 3.** *For some $d$, let $D$ be a $d$-state with $i = i^*$ and let $I$ be the permutation consecutive interval that begins in $i^* + 1$, contains the $d$ permutation entries of $D$ and is the minimal interval that satisfies these requirements (thus $|I| >= d$). Suppose that in every $D$-compliant state, the permutation $S$ after $i$ leaves $I$ satisfies again the permutation constraints of $D$. Then $D$ is said to be recyclable.*

At the point where $i$ leaves $I$, the permutation entries that are specified by $D$ have the same values that they had when $D$ occurred and they have a very high probability of more than $e^{-d}$ to remain in place during the next $N-i$ rounds until the index $i$ completes a full traversal of $S$. Combining in the $1/N$ probability of the index $j$ to be also in place at that time, we get an overall probability of at least $e^{-d}/N$ for another $D$-compliant state, exactly $N$ rounds after the first one.

Moreover, $D$ remains recyclable and the second occurrence of $D$ causes a third occurrence with the same probability of $e^{-d}/N$ and so forth. We formalize this observation in Theorem 2.

**Theorem 2.** *Let $D$ be a recyclable RC4 $d$-state and suppose that the state of round $t$ has the index $i$ as specified by $D$ and that the rest of the state is distributed uniformly. Then the probability of the states in rounds $t, t+N, t+2N, t+3N, \ldots, t+(k-1)N$ to be $D$-compliant is bounded from below by $e^{-d(k-1)}/N^{d+k}$.*

*Proof.* Since a $D$-compliant requires (except for $i$) $d$ specific permutation elements and specific index $j$, the probability of the state in round $t$ to be $D$-compliant is $N^{-1-d}$. The probability of round $t+N$ to have $D$-compliant state (given that round $t$ had one) is at least $e^{-d}/N$ and the same situation holds for round $t+2N, t+3N, \ldots, t+(k-1)N$ and thus the overall probability of the whole sequence to be $D$-compliant is at least $N^{-1-d} \cdot (e^{-d}/N)^{k-1} = e^{-d(k-1)}/N^{d+k}$. □

### 4.3 Availability of Recyclable Fortuitous States

We enumerated the fortuitous states of different orders to check whether and how many recyclable fortuitous states exist. The results are assorted where for some orders we found few states (9 states of order 3 and 100 states of order 5), whereas for other orders we found that most of the fortuitous states are recyclable. 512 out of 516 fortuitous states of order 2 are recyclable, more than 500 of them from the family $\{x, x+3, [-1, x+1 \textit{ or } x+2]\}$. 4011 out of 6540 fortuitous states of order 4 are recyclable, more than 4000 of them from the family $\{x, x+6, [-3, 1, x+4 \textit{ or } x+5 \textit{ or } x+6 \textit{ or } x+7, x \textit{ or } x+1 \textit{ or } x+2 \textit{ or } x+3]\}$. There are no recyclable fortuitous states of order 6.

Due to time limitations, we were not able to complete the research on the availability of recyclable states of high order. Our intuition and some simulations we made lead us to the conjecture that recyclable fortuitous states of high order are rare or unavailable at all. However, when such states occur they tend to come in families, e.g., recyclable fortuitous states of an even order $2d$ tend to come in large families of about $N \cdot (2d)^d$ members (as described above for $d = 1, 2$), where the index $i$ is "free" and the second half of the permutation constraints have some "freedom" (each entry has $2d$ alternatives).

### 4.4 Recycling Attacks

When the state in question is predictive, the recyclability property becomes extremely important since the occurrence of these states are expressed in the output stream.

#### Statistical Analysis of RC4 Streams

**Theorem 3.** *Let $D$ be a recyclable $b$-predictive $d$-state and suppose that the predicted outputs of $D$ are the values $z_0, \ldots, z_{b-1}$ in distances $r_0, \ldots, r_{b-1}$ after the occurrence of $D$. Then the following pattern has probability of at least $N^{-dk} \cdot (1 + \frac{N^{dk-d-k}}{e^{d(k-1)}})$.*

$$\underbrace{\overset{r_0}{z_0}, \ldots, \overset{r_1}{z_1}, \ldots, \overset{r_{b-1}}{z_{b-1}}, \ldots,}_{N} \underbrace{\overset{r_0+N}{z_0}, \ldots, \overset{r_{b-1+N}}{z_{b-1}}, \ldots,}_{N} \ldots, \underbrace{\overset{r_0+(k-1)N}{z_0}, \ldots, \overset{r_{b-1+(k-1)N}}{z_{b-1}}, \ldots}_{N}$$

*Proof.* Let $E_T$ be the target event in which the output values of the $dk$ rounds in question comply to the pattern. By Theorem 2 the probability for $k$ occurrences of $D$ in $N$-distances is $e^{-d(k-1)} \cdot N^{-d-k}$. We denote this event by $E_S$. When $E_S$ occurs $E_T$ occurs with probability 1 and otherwise $E_T$ has the trivial probability of $N^{-2k}$.

$$\begin{aligned}
\mathbb{P}[E_T] &= \mathbb{P}[E_S] \cdot \mathbb{P}[E_T|E_S] + \mathbb{P}[\overline{E_S}] \cdot \mathbb{P}[E_T|\overline{E_S}] \\
&= e^{-d(k-1)} \cdot N^{-d-k} \cdot 1 + (1 - e^{-d(k-1)} \cdot N^{-d-k}) \cdot N^{-dk} \\
&= N^{-dk} \cdot (1 + \frac{N^{dk-d-k}}{e^{d(k-1)}}) - \underbrace{e^{-d(k-1)} \cdot N^{-d-(d+1)k})}_{negl(N^{-dk})}
\end{aligned}$$

□

Notice that for a fixed $d$ the relative bias is proportional to $\left(\frac{N^{d-1}}{e^d}\right)^k$ and thus can increase to arbitrary values by increasing $k$. However, despite of the hugeness of these relative biases, they can be hardly used by distinguishing algorithms. When $k$ grows, the original probability ($N^{-2k}$) decreases rapidly and the amount of data that is required for the distinguishing increases. However, large relative biases can be used for prediction of output bits and words with large advantages.

**Theorem 4.** *Let $\mathcal{X}$ be a distribution and suppose that a particular bit-pattern $B = b_0 b_1 \ldots b_{k-1}$ occurs with probability $p \cdot (1+b)$ in $\mathcal{X}$ (for $p = 2^{-k}$). In addition, suppose that for a parameter $v < k$ all the other $k$-bit strings with the same $v$-bit prefix as $B$ have trivial probability of $p$. Then there exists an algorithm that once in $\frac{2^k}{2^{k-v}+b}$ samples (on average) of $k$-bit strings from $\mathcal{X}$ predicts the $(k-v)$-bit suffix with success probability $\frac{1+b}{2^{k-v}+b}$.*

We prove this theorem in Appendix B. Thus a $k$-bit pattern that occurs with relative bias $b$ can be used to predict a bit ($v{=}k{-}1$) with success probability

$\frac{1+b}{2+b} = 1/2 + \frac{b}{4+2b}$ once in every $\frac{2^k}{2+b}$ samples on average. The same pattern can be used to predict a byte ($v{=}k{-}8$) with success probability $\frac{1+b}{256+b}$ once in every $\frac{2^k}{256+b}$ samples on average.
We summarize the practical predictions that are derived from the combination of Theorems 3 and 4 in the table in Figure 5.

| $d$ | States | $k$ | Relative Bias | $\mathbb{P}[E_T\|\overline{E_S}]$ | Samples Number | Byte Prediction | Bit Prediction |
|---|---|---|---|---|---|---|---|
| 2 | 512 | 2 | 0.135 | $2^{-40}$ | $2^{29.9}$ $(2^{23})$ | $1.135/N$ | 0.53 |
| | | 3 | 4.7 | $2^{-56}$ | $2^{44.3}$ $(2^{39})$ | $4.63/N$ | 0.85 |
| | | 4 | 162 | $2^{-72}$ | $2^{55.7}$ $(2^{54.3})$ | 0.39 | 0.994 |
| | | 5 | 5628 | $2^{-88}$ | $2^{66.5}$ | 0.956 | $\approx 1$ |
| 3 | 9 | 2 | 12.75 | $2^{-56}$ | $2^{49}$ $(2^{44.8})$ | 0.05 | 0.93 |
| | | 3 | 41586 | $2^{-80}$ | $2^{61.5}$ | $\approx 1$ | $\approx 1$ |
| 4 | 4011 | 2 | 1200 | $2^{-72}$ | $2^{50}$ | 0.825 | 0.999 |
| | | 3 | $2^{28.5}$ | $2^{-104}$ | $2^{63.5}$ | $\approx 1$ | $\approx 1$ |

**Fig. 5.** Predictions of bits and bytes. The samples number are calculated for bit prediction and whenever the samples number for byte prediction is significantly different, we add it in parentheses.

**State Recovery Attack**
Some of the state recovery attacks that are analyzed in [2] work very fast when more than 100 permutation elements are known. Let us present a way to obtain these 100 elements within relatively small effort using recyclable fortuitous states. A recyclable fortuitous state of order $d$ repeats $k$ times with probability $e^{-d(k-1)} \cdot N^{-d-k}$. The attacker can wait for the event where the expression pattern of this state repeats $k$ times in shifts of $N$ rounds ($e^{d(k-1)} \cdot N^{d+k}$) and guess that this external pattern stems from the corresponding internal pattern (with probability of almost 1). The amount of data that is required for this event to occur for appropriate selections of $k = d = 10$ is $e^{90}N^{20} \approx 2^{290}$. In that case, the attacker learns 100 permutation elements with probability 1. An equivalent attack that is based on occurrences of fortuitous states of order 100 get the hundred elements in a single state, but requires $2^{800}$ data in order to obtain them.

The attacks from [2] were tested in situations where the known elements are in a single permutation whereas in our case the known entries are actually $k$ times the same $d$ entries in $k$ relatively close permutations. However, many situations are impossible under the constraints that the same values appear in the same entries during the $kN$ rounds, e.g., the index $j$ cannot touch these values (except for the fortuitous scenario), and the attacker can rule out many impossible branches and converge faster to the correct solution. Still the $2^{290}$ data complexity is a potential one, depending heavily of efficient ways to exploit these $dk$ values. Another question that remains open at the moment regards the availability of recyclable fortuitous state of high order. There are such states of order 5, but there are none of order 6.

## 5 Summary

In this paper we presented new families of unique statistical patterns of the keystream generator of RC4. The families are large and the patterns are more frequent and more biased than previously known ones, and allow stronger attacks and new attacks on RC4. These patterns were hidden for a long period mainly due to the fact that they are spread over distant rounds and thus could not be accidentally detected. However, the slow evolution of the permutation preserves many permutation elements with high probability along large number of rounds and we cannot rule out the possibility that this direction of research was not yet extracted.

## A Statistical Biases and Distinguishing Algorithms

### A.1 Combining the Effect of Independent Events

We prove here Lemma 3. The discrimination of two distributions is given by the expression $\sum_s p_\mathcal{X} \lg \frac{p_\mathcal{X}(s)}{p_\mathcal{Y}(s)}$ where the sum is over all the possible strings. For a single event $E$ with probabilities $p$ and $p(1+b)$ the discrimination is (we use $q \overset{def}{=} 1-p$)

$$p \lg \frac{p}{p(1+b)} + q \lg \frac{q}{q-pb} = p \lg(1 - \frac{b}{1+b}) + q \lg(1 + \frac{pb}{q-pb})$$

We use the log approximation to get

$$\approx -p\frac{b}{1+b} + q\frac{pb}{q-pb} \approx pb^2$$

The following claim measures the effect of combining several distributions on the discrimination of the overall super-distribution.

*Claim.* Let $\mathcal{X}$ and $\mathcal{Y}$ be two distributions over the domain $\mathcal{S}$ and let $\mathcal{X}'$ and $\mathcal{Y}'$ be two distributions over the domain $\mathcal{S}'$. Let $\mathcal{X}\mathcal{X}'$ be the distribution (over $\mathcal{S} \times \mathcal{S}'$) that is created by concatenating the distributions $\mathcal{X}$ and $\mathcal{X}'$, and let $\mathcal{Y}\mathcal{Y}'$ be the equivalent distribution for $\mathcal{Y}$ and $\mathcal{Y}'$. Then $D(\mathcal{X}\mathcal{X}', \mathcal{Y}\mathcal{Y}') = D(\mathcal{X}, \mathcal{Y}) + D(\mathcal{X}', \mathcal{Y}')$.

*Proof.*

$$\begin{aligned}
D(\mathcal{X}\mathcal{X}', \mathcal{Y}\mathcal{Y}') &= \sum_{s\in\mathbf{S}, s'\in\mathbf{S}'} p_{\mathcal{X}\mathcal{X}'}(s,s') \lg \frac{p_{\mathcal{X}\mathcal{X}'}(s,s')}{p_{\mathcal{Y}\mathcal{Y}'}(s,s')} = \\
&= \sum_{s\in\mathbf{S}} \sum_{s'\in\mathbf{S}'} p_{\mathcal{X}}(s) p_{\mathcal{X}'}(s') \lg \frac{p_{\mathcal{X}}(s) p_{\mathcal{X}'}(s')}{p_{\mathcal{Y}}(s) p_{\mathcal{Y}'}(s')} = \\
&= \sum_{s\in\mathbf{S}} \sum_{s'\in\mathbf{S}'} p_{\mathcal{X}}(s) p_{\mathcal{X}'}(s') \left( \lg \frac{p_{\mathcal{X}}(s)}{p_{\mathcal{Y}}(s)} + \lg \frac{p_{\mathcal{X}'}(s')}{p_{\mathcal{Y}'}(s')} \right) = \\
&= \left( \sum_{s'\in\mathbf{S}'} p_{\mathcal{X}'}(s') \right) \cdot \left( \sum_{s\in\mathbf{S}} p_{\mathcal{X}}(s) \lg \frac{p_{\mathcal{X}}(s)}{p_{\mathcal{Y}}(s)} \right) + \\
&\quad + \left( \sum_{s\in\mathbf{S}} p_{\mathcal{X}}(s) \right) \cdot \left( \sum_{s'\in\mathbf{S}'} p_{\mathcal{X}'}(s') \lg \frac{p_{\mathcal{X}'}(s')}{p_{\mathcal{Y}'}(s')} \right) = \\
&= D(\mathcal{X}, \mathcal{Y}) + D(\mathcal{X}', \mathcal{Y}')
\end{aligned}$$

□

By applying Claim A.1 recursively we can get a generalization of this claim, saying that the overall discrimination of a combination of several independent distributions is the sum of the discriminations of these distributions. The combination of independent events for the same distribution (as described in Lemma 3) is equivalent to the combination of independent distributions and thus the discrimination can be summed over the events to get $\sum_i p_i b_i^2$.

### A.2 Discriminations, Samples Numbers and Success Rates

We prove here Lemma 4. We denote the discrimination between two distributions by $D$ and the discrimination between $\ell$ samples from these distributions by $L$. From [14], $L$ is on one hand proportional to the number of samples and on the other hand induces a bound for the false positive rate $\alpha$ and the false negative rate $\beta$ by the inequality

$$L = \ell D \geq \beta \lg \frac{\beta}{1-\alpha} + (1-\beta) \lg \frac{1-\beta}{\alpha}$$

For $\beta = \alpha$ we get $\ell \geq (1/D) \cdot (1 - 2\alpha) \cdot log \frac{1-\alpha}{\alpha}$.

## B Statistical Biases and Bit Predictions

We prove here Theorem 4.

*Proof.* Let $E_{PREF}$ and $E_{SUFF}$ be the events where the $v$-bit prefix of $B$ and the $(k-v)$-bit suffix of $B$ occur, and let $E$ be the joint event where $B$ occur (namely, $E = E_{PREF} \wedge E_{SUFF}$). Let $E' = E_{PREF} \wedge \overline{E_{SUFF}}$. The algorithm analyzes the output stream of $\mathcal{X}$ and locates occurrences of $E_{PREF}$. Notice that the bias of $E$ affects the probability $E_{PREF}$ in the following manner:

$$\mathbb{P}[E_{PREF}] = \mathbb{P}[E] + \mathbb{P}[E'] = p(1+b) + (2^{k-v} - 1)p = \frac{2^{k-v} + b}{2^k}$$

In such occurrences the algorithm predicts that $E_{SUFF}$ will occur with the following success probability

$$\begin{aligned}\mathbb{P}[E_{SUFF}|E_{PREF}] &= \frac{\mathbb{P}[E_{SUFF} \cap E_{PREF}]}{\mathbb{P}[E_{PREF}]} = \\ &= \frac{\mathbb{P}[E]}{2^{-v}(1 + 2^{v-k}b)} = \\ &= \frac{2^{-k}(1+b)}{2^{-v}(1 + 2^{v-k}b)} = \frac{1+b}{2^{k-v} + b}\end{aligned}$$

□

## References


1. Jovan Dj. Golić: Linear Statistical Weakness of Alleged RC4 Key-Stream Generator. EUROCRYPT: Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, EUROCRYPT'97 pp. 226–238
2. Lars R. Knudsen, Willi Meier, Bart Preneel, Vincent Rijmen and Sven Verdoolaege: Analysis Methods for (Alleged) RC4. ASIACRYPT: Advances in Cryptology, International Conference on the Theory and Applications of Cryptology and Information Security, ASIACRYPT'98 pp. 327–341
3. Serge Mister and Stafford E. Tavares: Cryptanalysis of RC4-like Ciphers. SAC: Selected Areas in Cryptography, SAC'98 pp. 131–143
4. Scott R. Fluhrer and David A. McGrew: Statistical Analysis of the Alleged RC4 Keystream Generator. FSE: Fast Software Encryption, FSE'00 pp. 19–30
5. Itsik Mantin and Adi Shamir: A Practical Attack on Broadcast RC4. FSE: Fast Software Encryption, FSE'01
6. Scott R. Fluhrer, Itsik Mantin and Adi Shamir: Weaknesses in the Key Scheduling Algorithm of RC4. SAC: Annual International Workshop on Selected Areas in Cryptography, SAC'01
7. I. Mironov: (Not So) Random Shuffles of RC4. Crypto'02
8. Souradyuti Paul and Bart Preneel: Analysis of Non-fortuitous Predictive States of the RC4 Keystream Generator. Progress in Cryptology - INDOCRYPT 2003: 4th International Conference on Cryptology in India INDOCRYPT'03 pp. 52–67
9. Souradyuti Paul and Bart Preneel: A New Weakness in the RC4 Keystream Generator and an Approach to Improve the Security of the Cipher. Fast Software Encryption: 11th International Workshop, FSE'04 pp. 245–259
10. B. Zoltak: VMPC one-way function and stream cipher. Fast Software Encryption: 11th International Workshop, FSE'04 pp. 210–225
11. Jonathan J. Hoch and Adi Shamir: Fault Analysis of Stream Ciphers. CHES: Cryptographic Hardware and Embedded Systems, CHES'04 pp. 240–253

12. E. Biham, L. Granboulan and P. Nguyen: Impossible and Differential Fault Analysis of RC4. Fast Software Encryption: 12th International Workshop, FSE'05
13. Alexander Maximov: Two Linear Distinguishing Attacks on VMPC and RC4A and Weakness of the RC4 Family of Stream Ciphers. Fast Software Encryption: 12th International Workshop, FSE'05
14. Richard E. Blahut: Principles and Practice of Information Theory. Addisson-Wesley (1983)
15. Hal Finney: an RC4 cycle that can't happen. (1994)
16. Itsik Mantin: The Security of the Stream Cipher RC4. Master Thesis (2001) The Weizmann Institue of Science
17. Alexander L. Grosul and Dan S. Wallach: a Related-Key Cryptanalysis of RC4. Technical Report TR-00-358, Department of Computer Science, Rice University (2000)
18. Andrew Roos: A Class of Weak Keys in the RC4 Stream Cipher. Posted to sci.crypt (1995)