Generalized RC4 Key Collisions and Hash Collisions

Jiageng Chen * and Atsuko Miyaji**

School of Information Science,
Japan Advanced Institute of Science and Technology,
1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan
{jg-chen, miyaji}@jaist.ac.jp

Abstract. In this paper, we discovered that RC4 can generate colliding key pairs with various hamming distances, other than those found by Matsui (with hamming distance one), and by Chen and Miyaji (with hamming distance three). We formalized RC4 colliding key pairs into two large patterns, namely, Transitional pattern and Self-Absorbing pattern, according to the behavior during KSA. The colliding key pairs found in the previous researches can be seen as either subsets of the Transitional pattern or of the Self-Absorbing pattern. We analyzed both patterns and clarified the relations among the probability of key collision, key length and hamming distances which yield the colliding key pairs. Also we show how to make use of the RC4 key collision patterns to find collisions of RC4-Hash function which was proposed in INDOCRYPT 2006. Some concrete experimental results (RC4-Hash collision and RC4 colliding key pairs) are also given in this paper.

1 Introduction

The stream cipher RC4 is one of the most famous ciphers widely used in real world applications such as Microsoft Office, Secure Socket Layer (SSL), Wired Equivalent Privacy (WEP), etc. Due to its popularity and simplicity, RC4 has become a hot cryptanalysis target since its specification was made public on the Internet in 1994 [5]. Various general weaknesses of RC4 have been discovered in some previous works including [6–8], etc. Another popular cryptanalysis direction of RC4 is in the WEP environment. Such works include [9–12], etc.

Our paper focuses on RC4 key collisions, especially the existence of secret key pairs that generate the same initial states after key scheduling algorithm. This is a serious flaw for a stream cipher from the cryptographic point of view, since two encryptions will become the same under two different secret keys. The study of "colliding keys" of RC4 can be dated back to 2000. Grosul and Wallach [1] first pointed out that RC4 can generate near collisions when the key size is close to the full 256 bytes. In [2] first colliding key pairs with hamming

* This author is supported by the Graduate Research Program, JAIST.
** This work is supported by Grant-in-Aid for Scientific Research (B), 20300003.

distance one were discovered, where hamming distance one means that the two keys differ from each other at one position. Later in [3], other colliding key pairs with hamming distance three were found. Note that these researches can also generate "near colliding keys" of RC4, that generate initial states with small hamming distances after key scheduling algorithm. In a sense, these researches mean that we can control key scheduling algorithm. Recently, a new type of attack, which uses such two initial states with small hamming distances, has been proposed [4].

In this paper, we further analyzed the RC4 colliding key behavior, and we discovered that more colliding key pairs with various hamming distances exist in RC4, in addition to the ones found in [2] and [3]. We also found that all currently known RC4 colliding key pairs can be organized into two patterns, according to the behavior during KSA. We analyze these two generalized patterns and formalize the RC4 key collisions. Collision probability is estimated, and we point out that it is mainly affected by key length and hamming distances between the two keys. By making use of the RC4 key collision, we can also find collisions for RC4-Hash, which is built from RC4 [14] using KSA as a compression function.
Structure of the paper. In Section 2, we briefly describe the RC4 algorithm, followed by some previous works on RC4 key collisions. Section 3 shows the formalized RC4 colliding key patterns and how they work. The probability evaluation is given in Section 4, followed by the RC4-Hash Collisions in Section 5. Some experimental results on RC4-Hash Collisions and RC4 key collisions are given in Section 5 and Appendix.

2 Preparation

2.1 Description of RC4

The internal state of RC4 consists of a permutation S of the numbers $0, ..., N-1$ and two indices $i, j \in \{0, ..., N-1\}$. The index i is determined and known to the public, while j and permutation S remain secret. RC4 consists of two algorithms: The Key Scheduling Algorithm (KSA) and the Pseudo Random Generator Algorithm (PRGA). The KSA generates an initial state from a random key K of k bytes as described in Algorithm 1. It starts with an array $\{0, 1, ..., N-1\}$ where $N = 256$ by default. At the end, we obtain the initial state S_{N-1}. Once the initial state is created, it is used by PRGA. The purpose of PRGA is to generate a keystream of bytes which will be XORed with the plaintext to generate the ciphertext. PRGA is described in Algorithm 2. In this paper, we focus only on KSA.

Algorithm 1. KSA

1: **for** $i = 0$ **to** $N - 1$ **do**
2: $S[i] \leftarrow i$
3: **end for**
4: $j \leftarrow 0$
5: **for** $i = 0$ **to** $N - 1$ **do**
6: $j \leftarrow j + S[i] + K[i \bmod l]$
7: swap$(S[i], S[j])$
8: **end for**

Algorithm 2. PRGA

1: $i \leftarrow 0$
2: $j \leftarrow 0$
3: **loop**
4: $i \leftarrow i + 1$
5: $j \leftarrow j + S[i]$
6: swap$(S[i], S[j])$
7: keystream byte $z_i = S[S[i] + S[j]]$
8: **end loop**

2.2 Previous Research on RC4 key collisions

Three important previous studies on RC4 key collisions are [1], [2] and [3]. In [1], the authors pointed out that it's possible for two secret keys with length close to 256 bytes to generate similar internal state after KSA, and thus they will generate similar hundred byte output during PRGA. The reason for this is that for two keys K_1, K_2, if we assume $K_1[i] = K_2[i]$ except when $i = t$, then when t is close to 255, the two internal states will be substantially similar. However, this idea cannot generate strict key collisions, and this result only works for key lengths close to 256.

In [2], RC4 key collision was first discovered. The key pattern is almost the same as in [1], namely, two keys differ at only one byte position ($K_1[i] = K_2[i]$ except $i = t$) and the value difference is $1(K_1[t] = K_2[t] - 1)$. The intuition behind the collision is that from the first time i touches the different position t, the pattern ensures that there are always only two differences in the internal state as the key scheduling process continues. The difference is absorbed when i touches t for the last time. Please refer to [2] for the detailed description.

In [3], colliding key pairs with hamming distance three were first discovered. The key pattern is totally different from [1] and [2], namely, $K_1[d] = K_2[d] - t, K_1[d+1] = K_2[d+1] + t, K_1[d+t+1] = K_2[d+t+1] - t$. This key pattern shows us a more flexible way in which the two keys can differ from each other.

3 Generalized RC4 colliding key pairs

We found out that RC4 can generate many other colliding key pairs with different key relations, other than those found in [2] and [3]. We formalize all the currently known colliding key pairs into two patterns. We describe them in the following section by first giving the key relations, and then explaining how the two keys with these relations can achieve collisions.

3.1 Notation

- K_1, K_2: a secret key pair with some differences between them.
- $S_{1,i}, S_{2,i}$: S-Boxes corresponding to the secret key pair at time i.
- $i, j_{1,i}, j_{2,i}$: internal states of RC4. When $j_{1,i} = j_{2,i}$, we use j_i to denote.

– d: the first index of the key differences.
– h: hamming distances between the two keys (number of different positions where two keys differ from each other).
– k: the lengths (bytes) of the secret keys.
– n: the number of times the key differences appear during KSA. $n = \lfloor \frac{256+k-1-d}{k} \rfloor$.
– $l_1, ..., l_{h-1}$: the intervals between two consecutive key difference indices.
– l: interval between the first and last key difference indices, $l = \sum_{i=1}^{h-1} l_i$.
– Γ: the set of indices at which two keys differ from each other, $|\Gamma| = h$, $\Gamma = \{\gamma_1, ..., \gamma_h\}$ and $d = \gamma_1$.

3.2 Transitional Pattern

Key relations in Transitional pattern: Let $K_2[i] = K_1[i]+1$, $i \in \Gamma$, namely, two keys differ from each other at h places, and the value differences at these positions all equal 1.

Transitional pattern has the property that after the first internal state differences are generated, which is due to the key difference, the internal state differences are transferred to the later indices of the S-Box, and these differences exist before the last key difference comes into play during KSA.

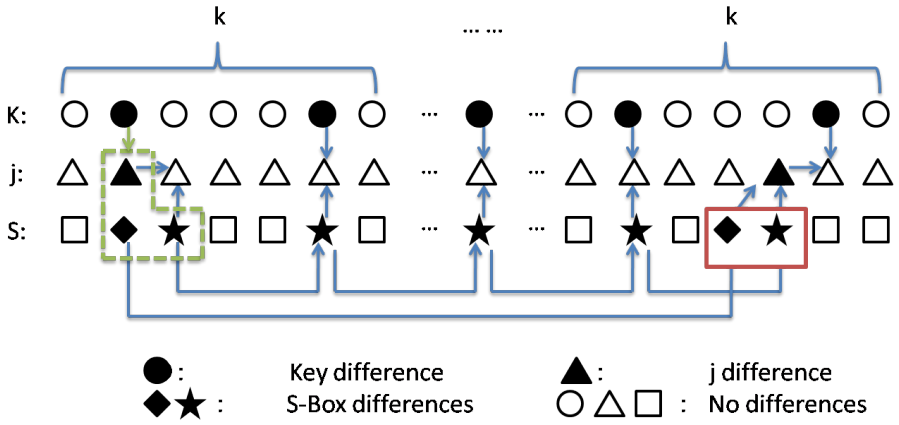


Fig. 1: Transitional Pattern

Figure 1 illustrates the case in which the secret keys are short, so they will appear several times during KSA. When i first touches the key difference, j difference and two S-Box differences are generated. Notice that Transitional pattern requires that one j equal i. Thus the two S-Box differences generated at the beginning are located next to each other, and meanwhile, we require that S-Box value differences also be one. The dotted line area in the figure shows the three internal state differences generated by the first key difference. The next two

j return to the same value, due to the effects of previous j difference (▲) and one S-Box difference(★). Meanwhile, the S-Box difference (★) is transferred to the next key difference index, and this transfer will repeat each time when i touches the next key difference index. The situation for the last appearance of the key is a little bit different. In order to achieve collision, we require that the two S-Box differences ⧫, ★ be in consecutive positions just before the last key difference index. The two S-Box differences are absorbed by each other and generate a j difference(▲). Finally, the last key difference is there to absorb the previous j difference and the internal states become the same.

The colliding key pairs found in [2] demonstrate a special case of this pattern, where the hamming distance between two keys can only be one ($|\Gamma| = h = 1$). In our generalized Transitional pattern, two keys can have various hamming distances as the probability allows. Here we give a more detailed example of a 128-byte colliding key pair with hamming distance three, to show how key collision can be achieved. Two keys differ from each other at indices $1, 4$ and 8.

Table 1: Transitional Pattern, $h = 3, n = 2(k = 128)$

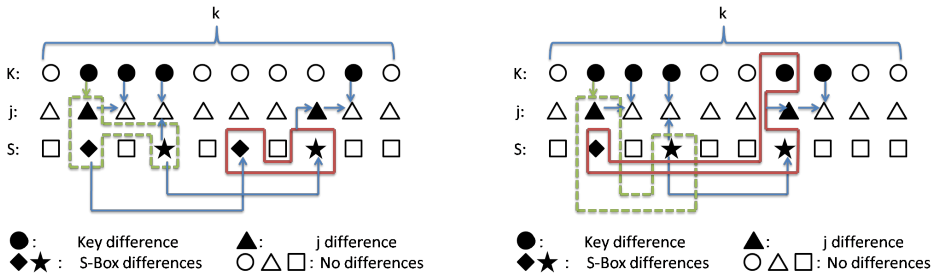
Internal State																				Difference
i	$K_1[i]/K_2[i]$	$j_{1,i}/j_{2,i}$	0	1	2	3	4	5	6	7	8	...	129	130	131	132	133	134	135	136	
0	$K_1[0]$	*		1	2																Same
	$K_2[0] = K_1[0]$	*		1	2																
1	$K_1[1]$	1		1	2																j, S-Box
	$K_2[1] = K_1[1] + 1$	2		2	1																
2	$K_1[2]$	4		1		2															S-Box
	$K_2[2] = K_1[2]$	4		2		1															
4	$K_1[4]$	8		1				2													S-Box
	$K_2[4] = K_1[4] + 1$	8		2				1													
8	$K_1[8]$	129		1									2								S-BoX
	$K_2[8] = K_1[8] + 1$	129		2									1								
129	$K_1[1]$	132		1												2					S-Box
	$K_2[1] = K_1[1] + 1$	132		2												1					
132	$K_1[4]$	135		1															2		S-Box
	$K_2[4] = K_1[4] + 1$	135		2															1		
134	$K_1[6]$	1																1	2		S-Box
	$K_2[6] = K_1[6]$	1																2	1		
135	$K_1[7]$	135																1	2		j
	$K_2[7] = K_1[7]$	134																1	2		
136	$K_1[8]$	*																1	2		Same
	$K_2[8] = K_1[8] + 1$	*																1	2		

The S-Box in Table 1 denotes the state after the swap. Notice that when $i = 134$, the other S-Box difference should be swapped to the index 134, but not

necessarily from index 1, as shown in the example. The first S-Box difference can be touched by j before 134, to be swapped to other positions. As long as this S-Box difference appears in index 134 when $i = 134$, the pattern works.

3.3 Self-Absorbing Pattern

In addition to the above Transitional pattern, we investigate that some of the other RC4 colliding key pairs have the following properties: the internal state differences are generated and absorbed within one key appearance, namely, the differences will not be transferred to the later parts of the S-Box. We can further divide this pattern into two sub-patterns, which are shown in Figures 2(a) and 2(b). Due to the self absorbing property, only one key appearance needs to be illustrated, since the others are the same. The ones found in [3] show a special case of Self-Absorbing pattern 1 ($|\Gamma| = h = 3$).



(a) Self-Absorbing Pattern 1 (b) Self-Absorbing Pattern 2

Fig. 2: Self-Absorbing Pattern

Key relations in Self-Absorbing pattern 1: $K_2[d] = K_1[d] + t$, $K_2[d+1] = K_1[d+1] - t$ and $K_2[i] = K_1[i] + t$ for $i \in \Gamma \setminus \{\gamma_1, \gamma_2\}$. The value difference t is the same for all h different positions.

Figure 2(a) illustrates the case of hamming distance 4 ($h = 4$) and $t = 2$. The first key difference generates three internal differences (dotted line area). In this illustration, the key value difference is $t = 2$, so the interval between two S-Box differences is also required to be t. The second key difference is there to absorb the previous j difference. The third key difference makes the S-Box difference(★) jump to the index just before the last key difference within this key appearance. S-Box difference ♦ should be swapped to the index two intervals from ★ when i touches that index. Then when i touches the S-Box difference ★, two S-Box differences absorb each other and generate a j difference (solid line area). Finally the last key difference is there to absorb the previous j difference, so that the internal states become the same. Table 2 is one example of 128-byte colliding key pair with hamming distance 4. Two keys differ from each other at indices $1, 2, 3$ and 8.

Table 2: Self-Absorbing Pattern 1, $h = 4, n = 2(k = 128)$

Internal State																				Difference
i	$K_1[i]/K_2[i]$	$j_{1,i}/j_{2,i}$	0	1	2	3	4	5	6	7	8	...	129	130	131	132	133	134	135	136	
0	$K_1[0]$	*	1			3															Same
	$K_2[0] = K_1[0]$	*	1			3															
1	$K_1[1]$	1	1			3															j, S-Box
	$K_2[1] = K_1[1] + 2$	3	3			1															
2	$K_1[2]$	*	1			3															S-Box
	$K_2[2] = K_1[2] - 2$	*	3			1															
3	$K_1[3]$	7	1						3												S-Box
	$K_2[3] = K_1[3] + 2$	7	3						1												
5	$K_1[5]$	1						1	3												S-Box
	$K_2[5] = K_1[5]$	1						3	1												
7	$K_1[7]$	7						1	3												j
	$K_2[7] = K_1[7]$	5						1	3												
8	$K_1[8]$	*						1	3												Same
	$K_2[8] = K_1[8] + 2$	*						1	3												
129	$K_1[1]$	129											129		131						j, S-Box
	$K_2[1] = K_1[1] + 2$	131											131		129						
130	$K_1[2]$	*											129		131						S-Box
	$K_2[2] = K_1[2] - 2$	*											131		129						
131	$K_1[3]$	135											129						131		S-Box
	$K_2[3] = K_1[3] + 2$	135											131						129		
133	$K_1[5]$	129															129		131		S-Box
	$K_2[5] = K_1[5]$	129															131		129		
135	$K_1[7]$	135															129		131		j
	$K_2[7] = K_1[7]$	133															129		131		
136	$K_1[8]$	*															129		131		Same
	$K_2[8] = K_1[8] + 2$	*															129		131		

Key relations in Self-Absorbing pattern 2: $K_2[d] = K_1[d] + t$, $K_2[d+1] = K_1[d+1] - t$, $K_2[i] = K_1[i] + t$ for $i \in \Gamma \setminus \{\gamma_1, \gamma_2, \gamma_{h-1}, \gamma_h\}$, $K_2[\gamma_{h-1}] = K_1[\gamma_{h-1}] - t^{'}$, $K_2[\gamma_h] = K_1[\gamma_h] + t^{''}$. For the previous $h-2$ different positions, the value difference t is the same. The last two value differences $t^{'}$ and $t^{''}$, which are determined by the specific key values, can be different values other than t.

Self-Absorbing pattern 2 is almost the same as Self-Absorbing pattern 1, except that in addition to using S-Box differences themselves, it also depends on key differences to absorb the S-Box differences (shown in solid line area in Figure 2(b)) at the final stage. This will allow a more flexible way of how the key value difference can vary, namely, the value difference can choose different values instead of a fixed value, as in the Transitional pattern and Self-Absorbing pattern 1. Table 3 shows a 128-byte colliding key pair example with hamming distance 5. The two keys differ from each other at indices $1, 2, 3, 5$ and 6.

Table 3: Self-Absorbing pattern 2, $h = 5, n = 2 (k = 128)$

Internal State																Difference
i	$K_1[i]/K_2[i]$	$j_{1,i}/j_{2,i}$	0	1	2	3	4	5	6	...	129	130	131	132	133	134	
0	$K_1[0]$	*		1		3											Same
	$K_2[0] = K_1[0]$	*		1		3											
1	$K_1[1]$	1		1		3											j, S-Box
	$K_2[1] = K_1[1] + 2$	3		3		1											
2	$K_1[2]$	*		1		3											S-Box
	$K_2[2] = K_1[2] - 2$	*		3		1											
3	$K_1[3]$	5		1			3										S-Box
	$K_2[3] = K_1[3] + 2$	5		3			1										
5	$K_1[5]$	5		1			3										j
	$K_2[5] = K_1[5] - 2$	1		1			3										
6	$K_1[6]$	*		1			3										Same
	$K_2[6] = K_1[6] + 4$	*		1			3										
129	$K_1[1]$	129									129		131				j, S-Box
	$K_2[1] = K_1[1] + 2$	131									131		131				
130	$K_1[2]$	*									129		131				S-Box
	$K_2[2] = K_1[2] - 2$	*									131		129				
131	$K_1[3]$	133									129				131		S-Box
	$K_2[3] = K_1[3] + 2$	133									131				129		
133	$K_1[5]$	133									129				131		j
	$K_2[5] = K_1[5] - 2$	129									129				131		
134	$K_1[6]$	*									129				131		Same
	$K_2[6] = K_1[6] + 4$	*									129				131		

4 Probability Evaluation

In this section, we evaluate the existence probabilities of RC4 colliding key pairs, and give approximate statistics on the scale and distribution of these keys.

4.1 Transitional Pattern

From the previous analysis, we know that colliding key pairs have the property that the key value difference is fixed at one, and the hamming distance can vary. We divide the whole process into three phases as shown in Figure 1, namely, the starting phase (first appearance of the key), the ending phase (last appearance of the key) and the repeating phase (middle repeating parts).

Starting Phase. First, before i touches d, j can not touch d or $d + 1$ with probability $(\frac{254}{256})^d$. When i touches d, $j_1 = d$ and $j_2 = d + 1$ with probability $\frac{1}{256}$. For each of the other key difference indices, we will pay the probability $\frac{1}{256}$ each, totally $(\frac{1}{256})^{h-1}$. When i is between two consecutive key difference indices, the pattern requires that j does not touch the later key difference index, otherwise i will never be able to touch the later S-Box difference again.

This will add $(\frac{255}{256})^{l_1-2}(\frac{255}{256})^{l_2-1}\cdots(\frac{255}{256})^{l_{h-1}-1}(\frac{255}{256})^{k-l-1} = (\frac{255}{256})^{k-h-1}$ to the total probability cost. Thus, the totally probability in the starting phase is $(\frac{1}{256})^h(\frac{254}{256})^d(\frac{255}{256})^{k-h-1}$.

Repeating Phase. Key will appear $n-2$ times during the repeating phase. For each key, the procedure is as follows. When i touches one key difference index, $\frac{1}{256}$ probability will be paid, $(\frac{1}{256})^h$ in total. When i is between two difference indices, it is not allowed to touch the later one (same as starting phase), this will add probability $(\frac{255}{256})^{k-h}$ in the repeating phase. Thus, the probability that one key appearance must pay is $(\frac{1}{256})^h(\frac{255}{256})^{k-h}$. Since the key will repeat $n-2$ times, the total probability is $(\frac{1}{256})^{h(n-2)}(\frac{255}{256})^{(k-h)(n-2)}$.

Ending Phase. We need probability $(\frac{1}{256})^{h-1}$ for the key difference indices except for the last one, and when i touches the index two intervals before the last key difference, the other S-Box difference needs to be swapped here with probability $\frac{1}{256}$. When i touches the index before the last key difference, we need two S-Box differences to swap with each other with probability $\frac{1}{256}$. And as in the repeating and starting phases, j can not touch the later key difference index when i is between the two. In the ending phase, this probability can be easily calculated as $(\frac{255}{256})^{l_1-1}(\frac{255}{256})^{l_2-1}\cdots(\frac{255}{256})^{l_{h-1}-1} = (\frac{255}{256})^{l-h+1}$. Thus the total probability in the ending phase is $(\frac{1}{256})^{h+1}(\frac{255}{256})^{l-h+1}$. By multiplying the probabilities in the three phases, we get the following theorem.

Theorem 1. *The probability of two keys with relations in Transitional pattern forming a colliding key pair, Prob(trans), is given as follows:*

$$Prob(tran) = \left(\frac{1}{256}\right)^{h\times n+1}\left(\frac{254}{256}\right)^{d}\left(\frac{255}{256}\right)^{(k-h)\times(n-3)+l-h} \approx O\left(\left(\frac{1}{256}\right)^{h\times n}\right)$$

4.2 Self-Absorbing Pattern 1

We only need to evaluate the probability of one key appearance, the other parts just repeat the first key appearance procedure. The value difference t is the same for all different positions in Self-Absorbing pattern 1.

Before i touches index d, we need $S_d[d] + t = S_d[d+t]$ with probability $\frac{255}{256}\times(\frac{254}{256})^{d-1}+\frac{1}{256}$ (Refer to [3] for the proof). When i touches d, we require $j_d = d$ with probability $\frac{1}{256}$. Then for i between $d+1$ and $d+t-1$, index $d+t$ should not be touched with probability $(\frac{255}{256})^{t-1}$. Then we need one of the differences ($\bigstar$ in Figure 2) to appear at indices $\Gamma \bigcup\{\gamma_h - 1\}\setminus\{\gamma_1,\gamma_2,\gamma_3,\gamma_h\}$ when i touches them, and j cannot touch the later key difference position when i is between the consecutive two of them. The probability can be calculated in the same way as in the repeating phase in Transitional pattern, namely, $(\frac{1}{256})^{h-3}(\frac{255}{256})^{l-t-h+3}$. Also we need the S-Box difference ($\blacklozenge$) to be at position $d+l-t-1$ and it cannot be touched when i is between $d+l-t$ and $d+l-1$. So this will give us probability $\frac{1}{256}(\frac{255}{256})^{t-1}$. Finally, when i touches index $d+l-1$, we need $j_{d+l-1} = d+l-1$ with probability $\frac{1}{256}$. By multiplying them together, we get the probability of one key appearance $(\frac{1}{256})^h(\frac{255}{256})^{l+t-h+1}(\frac{255}{256}(\frac{254}{256})^{d-1}+\frac{1}{256})$. Raise this probability to

the power of n due to the n times appearance of the key during KSA, we get the following theorem for the total probability of Self-Absorbing pattern 1.

Theorem 2. *The probability of two keys with relations in Self-Absorbing pattern 1 forming a colliding key pair, Prob(self1), is given as follows:*

$$\begin{aligned} \textit{Prob(self1)} &= \left(\frac{1}{256}\right)^{h\times n}\left(\frac{255}{256}\right)^{n\times(l+t-h+1)}\left(\frac{255}{256}\left(\frac{254}{256}\right)^{d-1}+\frac{1}{256}\right)^{n} \\ &\approx O\left(\left(\frac{1}{256}\right)^{h\times n}\right) \end{aligned}$$

4.3 Self-Absorbing Pattern 2

Self-Absorbing pattern 2 behaves very similarly to Self-Absorbing pattern 1, except at the final stage. Recall Figure 3, which shows that the S-Box differences are absorbed by both key differences and S-Box itself. So in this pattern, we don't need the S-Box difference (⧫) to be t intervals from another difference (★) in the final part. So we only need to cut off the probability for ⧫ to be transferred to the corresponding position in the final stage in Self-Absorbing pattern 1, namely, $\frac{1}{256}(\frac{255}{256})^{t-1}$. So the total probability for Self-Absorbing pattern 2 is as follows.

Theorem 3. *The probability of two keys with relations in Self-Absorbing pattern 2 forming a colliding key pair, Prob(self2), is given as follows:*

$$\begin{aligned} \textit{Prob(self2)} &= \left(\frac{1}{256}\right)^{(h-1)\times n}\left(\frac{255}{256}\right)^{n\times(l-h)}\left(\frac{255}{256}\left(\frac{254}{256}\right)^{d-1}+\frac{1}{256}\right)^{n} \\ &\approx O\left(\left(\frac{1}{256}\right)^{(h-1)\times n}\right) \end{aligned}$$

We can conclude that the probabilities of both Transitional pattern and Self-Absorbing patterns are mainly affected by hamming distance h and length of the secret key. The probability decreases as the hamming distance h becomes larger or the key length k becomes shorter (n becomes larger). Table 4 and 5 gives the probability data for different key lengths and hamming distances according to Theorems 1,2 and 3. According to the previous analysis, we know that Self-Absorbing pattern 1 requires $h \geq 3$, and Self-Absorbing pattern 2 requires $h \geq 5$.

5 Application to RC4-Hash Collisions

In INDOCRYPT 2006, a new hash function name "RC4-Hash" based on RC4 was proposed. It followed the "wide pipe" hash function design principle proposed by Lucks [13] and it was claimed to be as efficient as some widely-used hash

Table 4: Probabilities of colliding key pairs in Transitional pattern

$n(k)$ \ h	1	2	3	4	5
8(32)	2^{-64}	2^{-128}	2^{-192}	2^{-256}	2^{-320}
4(64)	2^{-32}	2^{-64}	2^{-96}	2^{-128}	2^{-160}
2(128)	2^{-16}	2^{-32}	2^{-48}	2^{-64}	2^{-80}
1(256)	2^{-8}	2^{-16}	2^{-24}	2^{-32}	2^{-40}

Table 5: Probabilities of colliding key pairs in Self-Absorbing pattern 1 (Self-Absorbing pattern 2)

$n(k)$ \ h	3	4	5
8(32)	$2^{-192}(-)$	$2^{-256}(-)$	$2^{-320}(2^{-256})$
4(64)	$2^{-96}(-)$	$2^{-128}(-)$	$2^{-160}(2^{-128})$
2(128)	$2^{-48}(-)$	$2^{-64}(-)$	$2^{-80}(2^{-64})$
1(256)	$2^{-24}(-)$	$2^{-32}(-)$	$2^{-40}(2^{-32})$

functions, such as SHA-family and MD-family, while also ruling out all possible generic attacks against those famous hash functions. First hash collision was found in [16] by exploiting the idea of Finney States [15]. We show in this section that totally different from [16], collisions can also be found by making use of previous key collision pattern. First we briefly describe the RC4-Hash algorithm, and then we give the collision analysis. For a more detailed description of the hash function, please refer to [14].

5.1 RC4-Hash

$\{0,1\}^{<2^{64}}$ denotes the set of all messages whose length is at most $2^{64}-1$. l is the output length of the RC4 hash function, $16 \leq l \leq 64$. RC4-Hash function can be described as $\{0,1\}^{<2^{64}} \rightarrow \{0,1\}^{8l}$.

Padding Rule: $\text{pad}(M) = \text{bin}_8(l)||M||1||0^k||\text{bin}_{64}(|M|) = M_1||\cdots||M_t$, where M_t is the last 512-bit block. $\text{bin}_{64}(|M|)$ is the 64-bit binary representation of the number of bits of M. k is the least non-negative integer such that $8+|M|+1+k+64 \equiv 0 \bmod (512)$ and $|M_i| = 512$.

Iteration Phase: Let $(S_0, j_0) = (S^{IV}, 0)$ be an initial value. The compression function C is invoked iteratively as follows:

$$(S_0, j_0) \xrightarrow{M_1} (S_1, j_1) \xrightarrow{M_2} \cdots (S_{t-1}, j_{t-1}) \xrightarrow{M_t} (S_t, j_t)$$

where $(S, j) \xrightarrow{X} (S^*, j^*)$ denotes $C((S, j), X) = (S^*, j^*)$.

Post-Processing: Let (S_t, j_t) be the internal state after the classical iteration. Compute $S_{t+1} = S_0 \circ S_t$ and $j_{t+1} = j_t$. Then compute $HBG_l(OWT(S_{t+1}, j_{t+1}))$.

$C((S,j),X)$	$OWT((S,j))$	$HBG_l((S,j))$
for $i = 0$ **to** 255 $j \leftarrow j + S[i] + X[r(i)]$ Swap(S[i],S[j]); Return (S,j)	Temp1 $= S$ **for** $i = 0$ to 511 $j \leftarrow j + S[i]$ Swap($S[i], S[j]$) Temp2 $= S$ $S =$ Temp1 $\circ$ Temp2 $\circ$ Temp1 Return (S, j)	**for** $i = 1$ to l $j \leftarrow j + S[i]$ Swap($S[i], S[j]$) Out $= S[S[i] + S[j]]$

$\circ$ denotes the composition of the permutations. Function $r : [256] \rightarrow [64]$ reorders the 64-byte message block. There are four r mapping functions(r_0, r_1, r_2, r_3) corresponding to the four iteration processes for each message block. In other words, each message block is reordered three times (r_0 is the identity permutation) during one iteration process. Refer to appendices for S^{IV} and r_i.

5.2 Collisions for RC4-Hash Function

Let's look at the iteration phase carefully. After message is padded, it is cut into 64-byte blocks, and each block is processed by the compression function C four times. The compression function C is actually the KSA in RC4, and the input message block can be seen as a 64-byte secret key, except for two differences. First, the message block is reordered by using r_i functions three times (instead of using the same 64-byte key which appears 4 times during KSA) and second, instead of the identity permutation used at the beginning of KSA, a shuffled S-Box S^{IV} is used as the initial S-Box. The similarities between the compression function and KSA gave us the intuition that we could make use of the RC4 key collision to find collisions for RC4-Hash. Now let's take a look at how these two differences can affect our search. In both Transitional pattern and Self-Absorbing pattern, when i touches the first different position, we need $S_d[d] + t = S_d[d + t]$. This is very easy to achieve when the initial S-Box is an identity permutation (j does not touch index d or index $d + t$ before i touches index d). But still we can make this happen with S^{IV} (Several candidates are available by checking S^{IV} carefully, and we use one of them in the following example). For the transitional pattern, the reordering of the message will not have much effect on finding collisions, because even though the different positions between the two messages change three times, we do not have to pay extra probabilities because there are no restrictions among these different message positions. Thus it works just the same as finding key collisions in the Transitional pattern. However, there are strict relations between the different positions in Self-Absorbing pattern (Self-Absorbing pattern 1: $K_2[d] = K_1[d] + t$, $K_2[d + 1] = K_1[d + 1] - t$ and $K_2[i] = K_1[i] + t$ for $i \in \Gamma \setminus \{\gamma_1, \gamma_2\}$), and the reordering of the message breaks those relations at the later rounds in the compression function, thus making it difficult to find a collision by using this pattern.

Here we give a concrete collision example by making use of the Transitional pattern. Since the initial S-Box S^{IV} is not an identity permutation, we need to first make two consecutive indices have the value difference one. There are several candidates we can use, by examining the S^{IV} carefully, we choose to let $S^{IV}[24] = 53$ appear in index 27 when i touches it, and $S^{IV}[28] = 54$ should not be touched by j before. Then we have two values, 53 and 54, next to each other at indices 27 and 28 when i touches index 27. The four iterations of the 64-byte message block during the compression function C can be seen as a KSA procedure with a 64-byte key. Since the message will be reordered three times, we need to check the mapping function r_i to identify the different positions. According to the Transitional pattern, in order to achieve a collision, two messages

should differ from each other at index 27, and the value difference should be one. According to r_1, r_2 and r_3, the differences between two messages will appear at indices 125, 179 and 213. After i touches 213, the two internal states become the same. Figure 4 describes the above collision by using Transitional pattern during one compression function C (63-byte message plus one padded byte).

From the above analysis, we know the complexity is equal to the key collision complexity in Transitional pattern, which is approximately equal to $O((\frac{1}{256})^{h\times n})$. In RC4-Hash with $k = 64(n = 4)$, we can find a collision with the smallest complexity $O(2^{32})$ when two messages have the smallest hamming distance $h = 1$. Notice that our method results in higher complexity than in [16], where complexity is $O(2^9)$.

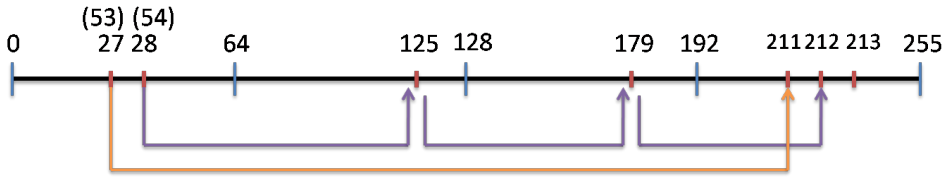


Fig. 3: RC4-Hash collision using Transitional Pattern

Here is the collision pair we found by using the RC4 key collision Transitional pattern with one day computation on an Intel Core i7 CPU PC (only one core was used). Messages and hash outputs are represented in Hexadecimal form. In our example, we set the output length l to be 16 bytes.
Message1(Message2): d8 4b be e4 ac c5 e3 c6 59 16 db b1 c2 7a de c2 62 5e 40 91 2c 7e de 4a f3 55 **8b(8c)** 2c 8f 96 f0 50 f7 54 78 3a 35 f5 ee 7e 76 72 35 83 0a e3 26 b5 06 7f 3b 1e b5 41 1c 1b ec 4e 80 c2 ba 64 9b
Hash Output: 76 54 b9 c6 65 f9 99 83 1b 66 c8 af 5f 0c 68 fa

5.3 Discussion

From the above analysis, we can see that the design of the compression function by modifying KSA using S^{IV} and mapping function r_i cannot eliminate the KSA collision property. Here we propose one method to mitigate the attack by redesigning S^{IV} carefully. Recall in the previous collision example, we need two values with value difference one to be next to each other when i touches the smaller one (Values 53 and 54 at indices 27 and 28 when i touches index 27). If we can prevent this from happening, then we can eliminate the collision. That is to say, if the index of value 54 is greater than 63 in S^{IV}, our Transitional pattern will not work. Generally speaking, design S^{IV} satisfies $|S^{IV}[i]^{-1} - (S^{IV}[i]+1)^{-1}| \geq 64 - \text{MIN}(S^{IV}[i]^{-1}, (S^{IV}[i]+1)^{-1})$ (Transitional pattern) for $0 < i < 64$, where v^{-1} denotes the index of value v. This will eliminate the collisions caused by the RC4 key collision patterns we found.

6 Conclusion

In this paper, we have shown that RC4 can generate many other colliding key pairs with various hamming distances. We analyzed the behavior of these colliding key pairs and formalized them into two patterns, which include the newly discovered colliding key pairs we found, and also the ones found in previous research. We further estimated the collision probabilities for all the RC4 colliding key pairs, and clarified the relations among collision probability, key length and hamming distance. Finally, we showed how the RC4 key collision patterns can be used to find collisions for RC4-Hash which was proposed at INDOCRYPT 2006. We leave how to use the mitigating methods proposed in this paper and [16] to construct secure RC4-Hash function as future work.

References

1. Grosul, A.L., Wallach, D.S.: A Related-Key Cryptanalysis of RC4. Technical Report TR-00-358, Department of Computer Science, Rice University (2000), http://cohesion.rice.edu/engineering/computerscience/tr/ TR Download.cfm?SDID=126
2. Matsui, M.: Key Collisions of the RC4 Stream Cipher. In: Dunkelman, O., Preneel, B. (eds.) FSE 2009. LNCS, vol. 5665, pp. 1.24. Springer, Heidelberg (2009)
3. Chen, J., Miyaji, A.: A New Class of RC4 Colliding Key Pairs With Greater Hamming Distance. In: Kwak, J., Deng, R., Won, Y., Wang, G. (eds.) ISPEC 2010. LNCS, vol.6047, pp. 30.44. Springer, Heidelberg (2010)
4. Miyaji, A., Sukegawa, M.: New Analysis Based on Correlations of RC4 PRGA with Nonzero-Bit Differences. In: *IEICE Trans.*, Fundamentals. vol. E93-A, No.6(2010), pp. 1066-1077.
5. Anonymous: RC4 Source Code. CypherPunks mailing list (September 9, 1994), http://cypherpunks.venona.com/date/1994/09/msg00304.html, http://groups.google.com/group/sci.crypt/msg/10a300c9d21afca0
6. Roos, A.: A Class of Weak Keys in the RC4 Stream Cipher (1995), http://marcel.wanda.ch/Archive/WeakKeys
7. Mantin, I., Shamir, A.: A Practical Attack on Broadcast RC4. In: Matsui, M. (ed.) FSE 2001. LNCS, vol. 2355, pp. 152.164. Springer, Heidelberg (2001)
8. Paul, S., Preneel, B.: A New Weakness in the RC4 Keystream Generator and an Approach to Improve Security of the Cipher. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 245.259. Springer, Heidelberg (2004)
9. Fluhrer, S., Mantin, I., Shamir, A.: Weaknesses in the Key Scheduling Algorithm of RC4. In: Vaudenay, S., Youssef, A.M. (eds.) SAC 2001. LNCS, vol. 2259, pp. 1.24. Springer, Heidelberg (2001)
10. Klein, A.: Attacks on the RC4 Stream Cipher. Designs, Codes and Cryptography 48(3), 269.286 (2008)
11. Tews, E., Weinmann, R.P., Pyshkin, A.: Breaking 104 Bit WEP in Less than 60 Seconds. In: Kim, S., Yung, M., Lee, H.-W. (eds.) WISA 2007. LNCS, vol. 4867, pp. 188.202. Springer, Heidelberg (2007)
12. Vaudenay, S., Vuagnoux, M.: Passive-Only Key Recovery Attacks on RC4. In: Adams, C., Miri, A., Wiener, M. (eds.) SAC 2007. LNCS, vol. 4876, pp. 344.359. Springer, Heidelberg (2007)

13. Lucks, S.: A Failure-Friendly Design Principle for Hash Functions. In: Advances in Cryptology - ASIACRYPT 2005. LNCS, vol,.3788,pp.19-35, Springer, Heidelberg (2005)

14. Chang, D., Gupta, K.C., Nandi, M.: RC4-Hash: A New Hash Function Based on RC4. In: Progress in Cryptology - INDOCRYPT 2006, LNCS, vol.4329, pp. 80-94, Springer, Heidelberg (2006)

15. Finney, H.: An RC4 cycle that can't happen. Newsgroup post in sci.crypt, September 1994.

16. Indesteege, S., Preneel, B.: Collision for RC4-Hash. In: 11th International Conference on Information Security. LNCS, vol. 5222, pp. 355-366. Springer, Heidelberg (2008)

A RC4-Hash (r_i functions and S^{IV})

r1 : 0, 55, 46, 37, 28, 19, 10, 1, 56, 47, 38, 29, 20, 11, 2, 57, 48, 39, 30, 21, 12, 3, 58, 49, 40, 31, 22, 13, 4, 59, 50, 41, 32, 23, 14, 5, 60, 51, 42, 33, 24, 15, 6, 61, 52, 43, 34, 25, 16, 7, 62, 53, 44, 35, 26, 17, 8, 63, 54, 45, 36, **27**, 18, 9.

r2: 0, 57, 50, 43, 36, 29, 22, 15, 8, 1, 58, 51, 44, 37, 30, 23, 16, 9, 2, 59, 52, 45, 38, 31, 24, 17, 10, 3, 60, 53, 46, 39, 32, 25, 18, 11, 4, 61, 54, 47, 40, 33, 26, 19, 12, 5, 62, 55, 48, 41, 34, **27**, 20, 13, 6, 63, 56, 49, 42, 35, 28, 21, 14, 7.

r3 : 0, 47, 30, 13, 60, 43, 26, 9, 56, 39, 22, 5, 52, 35, 18, 1, 48, 31, 14, 61, 44, **27**, 10, 57, 40, 23, 6, 53, 36, 19, 2, 49, 32, 15, 62, 45, 28, 11, 58, 41, 24, 7, 54, 37, 20, 3, 50, 33, 16, 63, 46, 29, 12, 59, 42, 25, 8, 55, 38, 21, 4, 51, 34, 17.

S^{IV}:
145, 57, 133, 33, 65, 49, 83, 61, 113, 171, 63, 155, 74, 50, 132, 248, 236, 218, 192, 217, 23, 36, 79, 72, **53**, 210, 38, 59, **54**, 208, 185, 12, 233, 189, 159, 169, 240, 156, 184, 200, 209, 173, 20, 252, 96, 211, 143, 101, 44, 223, 118, 1, 232, 35, 239, 9, 114, 109, 161, 183, 88, 66, 219, 78, 157, 174, 187, 193, 199, 99, 52, 120, 89, 166, 18, 76, 241, 13, 225, 6, 146, 151, 207, 177, 103, 45, 148, 32, 29, 234, 7, 16, 19, 91, 108, 186, 116, 62, 203, 158, 180, 149, 67, 105, 247, 3, 128, 215, 121, 127, 179, 175, 251, 104, 246, 98, 140, 11, 134, 221, 24, 69, 190, 154, 253, 168, 68, 230, 58, 153, 188, 224, 100, 129, 124, 162, 15, 117, 231, 150, 237, 64, 22, 152, 165, 235, 227, 139, 201, 84, 213, 77, 80, 197, 250, 126, 202, 39, 0, 94, 42, 243, 228, 87, 82, 27, 141, 60, 160, 46, 125, 112, 181, 242, 167, 92, 198, 172, 170, 55, 115, 30, 107, 17, 56, 31, 135, 229, 40, 111, 37, 222, 182, 25, 43, 119, 244, 191, 122, 102, 21, 93, 97, 131, 164, 10, 130, 47, 176, 238, 212, 144, 41, 14, 249, 220, 34, 136, 71, 48, 142, 73, 123, 204, 206, 4, 216, 196, 214, 137, 255, 195, 26, 8, 51, 178, 2, 138, 254, 90, 194, 81, 245, 106, 95, 75, 86, 163, 205, 70, 226, 28, 147, 85, 5, 110.

B RC4 colliding key pairs

B.1 Transitional Pattern, h=3, k=128. $K_1(K_2)$

71, **185(186)**, 1, 63, **192(193)**, 206, 161, 132, **114(115)**, 12, 69, 19, 160, 125, 44, 78, 26, 119, 59, 18, 200, 221, 130, 215, 157, 208, 205, 210, 165, 96, 99, 44, 68, 17, 146, 161, 227, 188, 123, 218, 172, 154, 100, 99, 92, 205, 235, 78, 179, 8, 5, 1, 142, 115, 31, 245, 151, 170, 140, 140, 104, 198, 128, 189, 145, 163, 42, 178, 113, 223, 135, 21, 243, 236, 90, 141, 70, 78, 96, 8, 200, 8, 161, 123, 112, 57, 190, 224, 179, 196, 41, 87, 24, 105, 231, 41, 84, 12, 139, 107, 82, 228, 130, 23, 148, 38, 196, 3, 238, 164, 2, 233, 22, 41, 182, 130, 201, 95, 211, 140, 11, 248, 189, 6, 109, 27, 92, 1.

B.2 Self-Absorbing Pattern 1, h=4, k=128. $K_1(K_2)$

41, **215(217)**, **60(58)**, **197(199)**, 78, 163, 94, 159, **253(255)**, 76, 84, 228, 174, 159, 214, 86, 52, 146, 24, 235, 130, 98, 91, 117, 23, 44, 155, 55, 136, 46, 182, 76, 55, 200, 20, 25, 171, 59, 184, 240, 6, 178, 173, 29, 33, 126, 49, 151, 200, 185, 218, 219, 60, 188, 14, 49, 51, 215, 123, 58, 26, 222, 26, 96, 177, 14, 13, 175, 9, 90, 106, 179, 57, 183, 103, 183, 55, 51, 40, 163, 193, 93, 187, 151, 209, 145, 42, 10, 70, 166, 179, 136, 166, 206, 153, 21, 100, 241, 226, 120, 165, 74, 159, 125, 18, 14, 77, 151, 79, 129, 201, 19, 23, 109, 75, 14, 29, 96, 118, 87, 75, 225, 31, 28, 248, 126, 161, 148.

B.3 Self-Absorbing Pattern 2, h=5, k=128. $K_1(K_2)$

222, **34(36)**, **98(96)**, **157(159)**, 174, **75(73)**, **231(235)**, 9, 221, 154, 135, 215, 175, 166, 27, 58, 91, 226, 252, 225, 7, 164, 124, 198, 65, 132, 222, 132, 205, 184, 196, 21, 86, 41, 124, 121, 115, 138, 108, 2, 26, 137, 55, 224, 46, 92, 109, 63, 15, 156, 104, 144, 101, 3, 41, 224, 98, 15, 185, 198, 152, 226, 148, 111, 2, 136, 35, 69, 159, 211, 250, 47, 130, 40, 200, 19, 97, 205, 250, 226, 34, 243, 45, 120, 86, 175, 52, 157, 145, 214, 138, 107, 182, 50, 247, 20, 121, 20, 144, 40, 172, 236, 150, 77, 196, 200, 158, 198, 44, 206, 73, 90, 169, 64, 152, 1, 82, 163, 192, 235, 246, 24, 121, 185, 234, 158, 48, 200.