New Results in Linear Cryptanalysis of RC5

Ali Aydın Selçuk [*]

Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County
Baltimore, MD, 21250, USA
e-mail: aselcu1@cs.umbc.edu

Abstract. We show that the linear cryptanalytic attack on RC5 that was published by Kaliski and Yin at Crypto'95 does not work as expected due to the failure of some hidden assumptions involved. Then we present new linear attacks on RC5. Our attacks use the same linear approximation as the one used by Kaliski and Yin. Therefore, the plaintext requirement of our attack is around $4w^{2r-2}$ which is impractically high for reasonably high values of w and r. These new attacks has also significances beyond the linear cryptanalysis of RC5 to show how linear cryptanalysis can carry on when the approximation used has a non-zero bias for the wrong key values. We also discuss certain issues about linear cryptanalysis of RC5 that need to be resolved for a better linear attack.

Keywords: Cryptology, cryptanalysis, block ciphers, RC5, linear cryptanalysis.

1 Introduction

RC5 is a secret key block cipher designed by Rivest [5]. Kaliski and Yin [1] published a linear cryptanalytic attack on RC5 at Crypto'95, which still remains as the only general linear attack on RC5 that has been published in the open literature. We are going to refer to this attack as the *K-Y Attack*. In this paper, we show that the K-Y Attack does not work as expected due to the failure of some hidden assumptions involved. Then we present some new attacks. Our attacks are based on the same linear approximation used in the K-Y Attack, but they are different from that attack in the way they use the approximation to recover the secret key.

We first briefly review RC5 and linear cryptanalysis. RC5 has a variable block size, a variable number of rounds, and a variable length secret key. A particular RC5 algorithm is defined by these three parameters and denoted as RC5-$w/r/b$: w, the word size in bits (half of a block is called a *word*); b, the key size in bytes; r, the number of rounds. For the encryption algorithm, we adopt the notation used in [1]. The algorithm is as follows:

[*] This research was done while the author was visiting RSA Laboratories.

```
$L_1 = L_0 + S_0$
$R_1 = R_0 + S_1$
**for** $i = 2$ **to** $2r + 1$ **do**
    $L_i = R_{i-1}$
    $R_i = ((L_{i-1} \oplus R_{i-1}) \lll R_{i-1}) + S_i$
```

In the algorithm, "$+$" denotes addition modulo 2^w, "$\oplus$" denotes bitwise xor, "$\lll$" denotes left rotation. The i^{th} iteration of the for loop is referred as the i^{th} *half-round*. The *first* half-round refers to the two initial equations. L_{i-1}, R_{i-1} denote the left and the right halves of the input and S_i denotes the subkey at the i^{th} half-round. (L_0, R_0) is the plaintext, (L_{2r+1}, R_{2r+1}) is the ciphertext.

Linear cryptanalysis is a kind of statistical correlation attack for block ciphers which was developed by Matsui [4] in 1993. The basic idea of linear cryptanalysis is to find a linear relation, which is called an *approximation*, among the plaintext, ciphertext and key bits, such that the probability of the approximation is different from 1/2. But, if wrong values are substituted for the key bits in the approximation, the approximation will behave randomly (i.e. its probability will be 1/2). The attacker collects plaintext/ciphertext pairs which are encrypted under the same key. Then he tries all possible combinations for the key bits involved in the approximation with all the plaintext/ciphertext pairs he has collected. The correct key bit combination is distinguished by its non-random behavior. Matsui [3] showed that the success probability of the attack is proportional to $N|p - 1/2|^2$ where N is the number of plaintext/ciphertext pairs collected.

Some specific notation used in this paper is as follows. RC5-w/r denotes the RC5 scheme with w bit words and r rounds where each round key is generated independently. $x[i]$ denotes the i^{th} bit of a binary string x, and $x[i \dots j]$ denotes the i^{th} through j^{th} bits of x; n denotes $2r + 1$.

The remainder of the paper is organized as follows. In §2 we discuss the hidden assumptions in the K-Y Attack and explain why they do not hold. In §3–7 we present our attacks and discuss their success rates. In §8 we conclude with some open research problems regarding linear cryptanalysis of RC5 and discuss briefly the factors that make linear cryptanalysis of RC5 harder than linear cryptanalysis of DES-like ciphers.

2 Hidden Assumptions in the K-Y Attack

In this section, we briefly discuss the K-Y Attack and show the hidden assumptions that cause the attack to fail.

2.1 The Attack

Let T denote $S_1[0] \oplus S_3[0] \oplus \cdots \oplus S_{2r-1}[0]$. The approximation used in the K-Y Attack is

$$R_0[0] \oplus L_{2r}[0] = T \qquad (1)$$

which is obtained by combining the half-round approximation

$$R_1[0] = R_0[0] \oplus S_1[0]$$

and the half-round approximation

$$R_i[0] = L_{i-1}[0] \oplus S_i[0]$$

for $i = 3, 5, \ldots, 2r-1$. The probability of this approximation, which we denote by p, is $\frac{1}{2} + \frac{1}{2w^{r-1}}$.

The K-Y Attack was different from all the previously published linear cryptanalytic attacks in the way that it was composed of multiple steps where each step aimed to recover one bit of the round key. The outline of the attack algorithm is as follows:

Step 1: Guess $S_n[0]$ by using the data with $L_n \bmod w = 1$.
Step 2: Guess T by using the data with $L_n \bmod w = 0$.
Step 3: For $i = 1, \ldots, w-1$, guess $S_n[i]$ by using the data with $L_n \bmod w = i$.

The point that is important for us in this algorithm is that, at each step, $L_n \bmod w$ is *fixed* to a certain value.

2.2 Hidden Assumptions

We implemented this attack on RC5-16/2 with $2|p-1/2|^{-2}$ plaintext/ciphertext pairs for each different value of $L_n \bmod w$ (i.e. $w \times 2|p-1/2|^{-2}$ texts in total). The success rate we observed for recovering S_n was around 11–15% as opposed to the 95–99% that was expected by Kaliski and Yin. Even more surprisingly, the success rate did not improve as we increased the amount of data used. These results led us to the following observations.

Let AH_i, for $i = 3, 5, \ldots, 2r-1$, denote the event that the i^{th} half-round approximation $R_i[0] = L_{i-1}[0] \oplus S_i[0]$ holds. The probability of the approximation $P(\mathrm{AH}_i)$ can be calculated as

$$\begin{aligned} P(\mathrm{AH}_i) = {} & P(\mathrm{AH}_i \mid R_{i-1} \bmod w = 0) \cdot P(R_{i-1} \bmod w = 0) \\ & + P(\mathrm{AH}_i \mid R_{i-1} \bmod w \neq 0) \cdot P(R_{i-1} \bmod w \neq 0). \end{aligned}$$

$P(\mathrm{AH}_i \mid R_{i-1} \bmod w = 0)$ is always equal to 1. $P(R_{i-1} \bmod w = 0)$ is equal to $1/w$ and $P(\mathrm{AH}_i \mid R_{i-1} \bmod w \neq 0)$ is equal to $1/2$, hence $P(\mathrm{AH}_i)$ is equal to $1/2 + 1/2w$, *given that the input (or the output) of the i^{th} half-round is uniformly random.*

An important point in the K-Y Attack is that at each step the value of $L_n \bmod w$ is *fixed* to a certain value and it is implicitly assumed that the probability of Approximation (1) does not depend on $L_n \bmod w$. This assumption is based on two other assumptions:

1. The probability $P(R_{i-1} \bmod w = 0)$ does not depend on $L_n \bmod w$;
2. The probability $P(\mathrm{AH}_i \mid R_{i-1} \bmod w \neq 0)$ does not depend on $L_n \bmod w$.

We will refer to these two assumptions as *Assumption 1* and *Assumption 2*, respectively.

2.3 On Assumption 1

We observe that the probability of a zero rotation in the $n-2^{nd}$ half-round, i.e. $P(R_{n-3} \bmod w = 0)$, depends on $L_n \bmod w$, hence Assumption 1 does not hold:

$$\begin{aligned} R_{n-3} &= L_{n-2} \\ &= ((R_{n-1} - S_{n-1}) \ggg R_{n-2}) \oplus R_{n-2} \\ &= ((L_n - S_{n-1}) \ggg R_{n-2}) \oplus R_{n-2}. \end{aligned}$$

Therefore, when $L_n - S_{n-1}$ is fixed, the distribution of R_{n-3} is *not* uniform. Hence, the probability $P(R_{n-3} \bmod w = 0)$, and therefore the probability of Approximation (1), is *not* independent of $L_n \bmod w$.

As an example, let δ denote the difference $L_n - S_{n-1} \bmod w$ and ρ' denote $R_{n-3} \bmod w$. The probability $P(\rho' = 0 \mid \delta = 0)$ for $w = 16$ is $1.56/w$ as opposed to the expected probability $1/w$.

2.4 On Assumption 2

We also observe that Assumption 2, just like Assumption 1, does not hold for the $n-2^{nd}$ half-round; i.e. $P(\mathrm{AH}_{n-2} \mid \rho' \neq 0)$ is *not* independent of $L_n \bmod w$:

First, we observe that the half-round approximation

$$R_{n-2}[0] = L_{n-3}[0] \oplus S_{n-2}[0] \tag{2}$$

can be expressed in terms of *only* R_{n-2}, S_{n-2}, and ρ'. The approximation is

$$\begin{aligned} R_{n-2}[0] &= L_{n-3}[0] \oplus S_{n-2}[0] \\ &= (R_{n-2} - S_{n-2})[\rho'] \oplus R_{n-3}[0] \oplus S_{n-2}[0] \\ &= (R_{n-2} - S_{n-2})[\rho'] \oplus \rho'[0] \oplus S_{n-2}[0]. \end{aligned}$$

Second, we know from Section 2.3 that ρ' (i.e. $R_{n-3} \bmod w$) is equal to $(((L_n - S_{n-1}) \ggg R_{n-2}) \oplus R_{n-2}) \bmod w$. Therefore, we observe that conditions on $L_n - S_{n-1}$ and ρ' together give information about $R_{n-2} \bmod w$.

These two observations imply that when $L_n - S_{n-1}$ and S_{n-2} are fixed, the condition $\rho' \neq 0$ gives information about Approximation (2) and possibly causes the probability $P(\mathrm{AH}_{n-2} \mid \rho' \neq 0)$ to be different from $1/2$. For example, for $w = 16$, $L_n - S_{n-1} = 1$, $S_{n-2} = 0$, the probability $P(\mathrm{AH}_{n-2} \mid \rho' \neq 0)$ is equal to 0.494 as opposed to 0.5.

At this point we should remark that the probability of the approximation does not depend on the top $w - \lg w$ bits of S_{n-2}. This fact is because ρ' and $L_n - S_{n-1}$ give information about only the last $\lg w$ bits of R_{n-2}; therefore $(R_{n-2} - S_{n-2})[\rho']$ has a uniform distribution when $\rho' \geq \lg w$, regardless of S_{n-2}.

2.5 Overall Impact of the Assumptions

In every step of the attack the difference $L_n - S_{n-1}$ is fixed. When $L_n - S_{n-1}$ is fixed, R_{n-3} and R_{n-2} have a non-uniform distribution. This fact has two effects on the probability of the $n - 2^{nd}$ half-round approximation: First, the probability $P(R_{n-3} \bmod w = 0)$ may be different from $1/w$. Second, the probability $P(\mathrm{AH}_{n-2} \,|\, R_{n-3} \bmod w \neq 0)$ may be different from $1/2$.

Table 4 in Appendix A lists the bias of Approximation (2) for different values of $L_n - S_{n-1} \bmod w$ and $S_{n-2} \bmod w$ for $w = 16$. What is particularly important in Table 4 regarding the attack of Kaliski and Yin is the negative entries which correspond to the case $p < 1/2$. Success of Steps 2 and 3 of the attack depends on the assumption that $p > 1/2$ for every single value of $L_n \bmod w$. If the $(i, j)^{th}$ entry of Table 4 is negative, then for $S_{n-2} \bmod w = j$ the attack fails with very high probability at the step where $L_n - S_{n-1} \bmod w$ is fixed to i and the failure probability goes to one as the amount of data used goes to infinity. With respect to the numbers in Table 4, we calculated that the average success rate of the attack for recovering the last round key S_n in RC5-16/2 goes to 9.375% as the amount of data goes to infinity. We also calculated the success rate with $2|p - 1/2|^{-2}$ texts as 13.9%. These results matched our experimental results in Section 2.2 very well.

3 New Attacks

We developed a number of new linear cryptanalytic attacks on RC5. They all use Approximation (1), but they are different from the attack of Kaliski and Yin in the way they use the approximation to recover the round key S_n. Our attacks are similar to "Algorithm 2" of Matsui [3] which is sometimes referred as the *1R-method.* We unroll the last round and substitute the actual value of $L_{n-1}[0]$ in Approximation (1), which is $(R_n - S_n)[\rho] \oplus L_n[0]$, where ρ denotes $L_n \bmod w$ (i.e. the rotation amount in the last half-round). So, the approximation becomes

$$R_0[0] \oplus (R_n - S_n)[\rho] \oplus L_n[0] = T. \tag{3}$$

An important difference of our attacks from the 1R-method of Matsui is that, when we substitute a wrong value s for S_n in Approximation (3), the bias of the approximation is *not* zero. Moreover, the bias can be expressed in terms of s, S_n, ρ and the probability of the approximation, as will be shown in Section 4.

Our attacks can be classified into two types: In the first one, we fix ρ (i.e. $L_n \bmod w$) to a certain value at each step and we aim to recover one key bit at a time. We will refer to the attacks of this type as the *1-bit attacks.* In the second type of attack, we aim to recover a group of consecutive key bits at the same time. We will refer to the attacks of this type as the *multi-bit attacks.* We will describe the attacks in more detail in Sections 5 and 6.

An important issue regarding the experimental comparison of the attacks presented in the following sections is that they are all run on relatively small versions of RC5 such as $r = 2, 4$. The reason for this choice of small parameters is just to make the experiments computationally feasible. Our attack techniques all use the same approximation (i.e. Approximation (3)), and they differ only in the way they use this approximation to recover the secret key. Therefore, increasing the number of rounds does not have much effect on the relative performance of

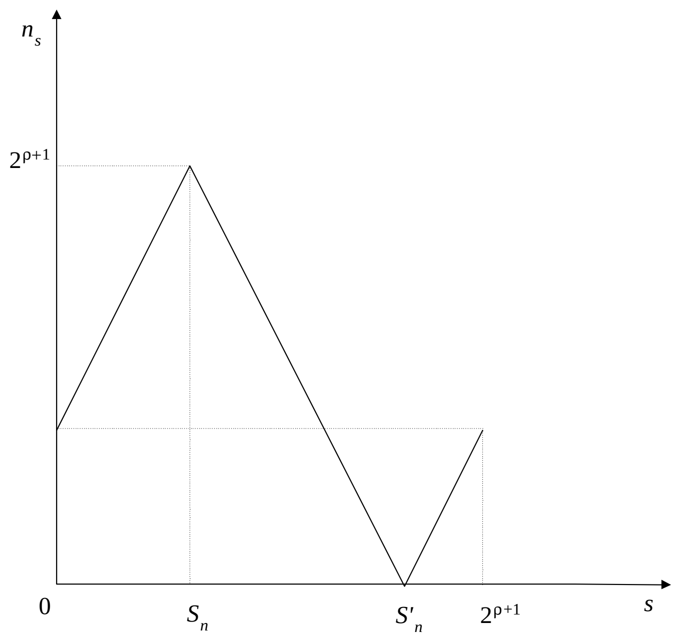

Fig. 1. The variable s denotes the possible key values that can be tried for $S_n[0\ldots\rho]$. n_s denotes the number of different values of $R_n[0\ldots\rho]$ such that $(R_n-s)[\rho]=(R_n-S_n)[\rho]$. S'_n is the key that differs from S_n only at the ρ^{th} bit.

these attack techniques and the experiments with relatively small values of r give a general comparison of the attacks.

4 Bias for a Wrong Key

As mentioned in Section 3, something special with Approximation (3) is that, when a wrong key value s is substituted for S_n, the bias of the approximation is not zero. An important observation to understand the behavior of the approximation is the following. When s is substituted for S_n in Approximation (3), the result is the same as the result for S_n if and only if $(R_n-s)[\rho]$ is the same as $(R_n-S_n)[\rho]$. These two bits agree when one of the following two conditions is satisfied:

Let S_{min} denote $\min\{S_n[0\ldots\rho-1], s[0\ldots\rho-1]\}$ and similarly, let S_{max} denote $\max\{S_n[0\ldots\rho-1], s[0\ldots\rho-1]\}$:

1. $s[\rho]=S_n[\rho]$, and $R_n[0\ldots\rho-1]<S_{min}$ or $R_n[0\ldots\rho-1]\geq S_{max}$.
2. $s[\rho]\neq S_n[\rho]$ and $S_{min}\leq R_n[0\ldots\rho-1]<S_{max}$.

Let n_s denote the number of different values of $R_n[0\ldots\rho]$ such that we have $(R_n-s)[\rho]=(R_n-S_n)[\rho]$. Figure 1 illustrates the value of n_s for $0\leq s<2^{\rho+1}$.

More specifically, n_s is $2^{\rho+1}$ for the correct key S_n; it decreases by two as s gets further from S_n in either direction; and it is zero at S'_n, which denotes the key that differs from S_n only at the ρ^{th} bit.

Assuming that the probability that the approximation holds and the probability that the result of the approximation is the same for both s and S_n are independent (where both probabilities are taken over the plaintext), we obtain a similar figure for the bias of the approximation with s substituted for S_n. Figure 2 shows the expected bias of Approximation (3) for different values of s. Let N denote the number of plaintext/ciphertext pairs satisfying $L_n \bmod w = \rho$ for some fixed ρ. Let U_s denote the number of those texts such that the left side of Approximation (3) is 1 when we substitute s for S_n, and let B_s denote the bias $U_s - N/2$. Figure 2 illustrates the expected bias $\mathrm{E}[B_s]$ for $0 \leq s < 2^{\rho+1}$, assuming $\mathrm{E}[B_{S_n}] > 0$.

The significance of Figure 2 is that it shows what the expected bias of Approximation (3) will be when $L_n \bmod w$ is fixed and a wrong value s is substituted for the round key S_n. This behavior of the bias has a crucial role in the attacks we develop in this paper, especially in the 1-bit attacks (see Section 5).

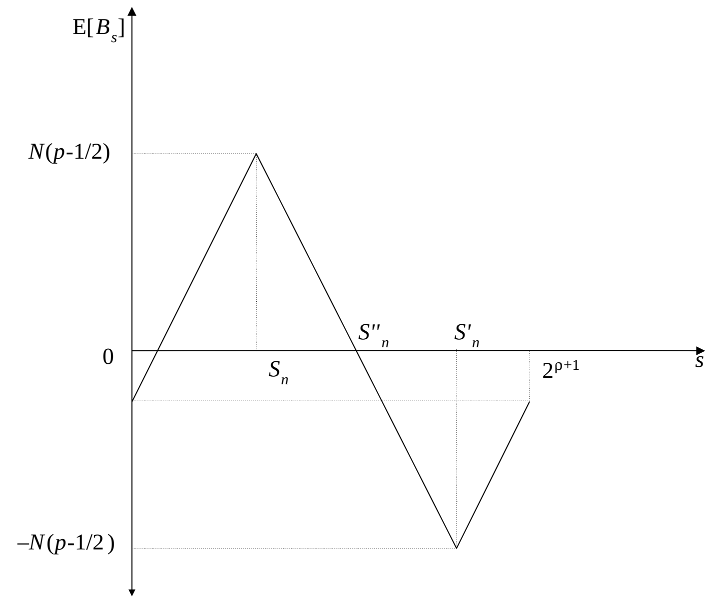

Fig. 2. Expected bias for different values of s for $L_n \bmod w = \rho$. S''_n is the key that differs from the correct key S_n only at the $\rho - 1^{st}$ bit. p denotes the probability of the approximation given $L_n \bmod w = \rho$.

5 1-Bit Attacks

In this section we discuss the attacks that recover the round key S_n in a bitwise fashion (i.e. recovering one bit at a time). The aim of these attacks is to recover the key bit $S_n[\rho - 1]$ by using the data with $L_n \bmod w = \rho$ and given that the key bits $S_n[0 \ldots \rho - 2]$ are already recovered. The idea of attacking the $\rho - 1^{st}$ bit instead of the ρ^{th} one is inspired by the fact that p may be either less or greater than 1/2 depending on the value of ρ, S_{n-1} and S_{n-2} (see Appendix A). Moreover, S'_n, the number that differs from S_n only at the ρ^{th} bit, has the exact inverse bias of the correct key S_n, (i.e. $B_{S'_n} = -B_{S_n}$) over the data with $L_n \bmod w = \rho$ (Figure 2). Therefore, we cannot distinguish between S_n and S'_n by using the data with $L_n \bmod w = \rho$ since we do not know if $p > 1/2$ or not; and we cannot know if $S_n[\rho]$ is 0 or 1. But this is not the case for $S_n[\rho - 1]$. S''_n, the number that differs from S_n only at the $\rho - 1^{st}$ bit, has a zero expected bias over the data with $L_n \bmod w = \rho$ regardless of p (Figure 2). Therefore, we can distinguish between S_n and S''_n, and hence find out $S_n[\rho - 1]$ by using the data with $L_n \bmod w = \rho$, even if $p < 1/2$.

All of our 1-bit attacks are based on a generic attack algorithm. Assume we have recovered the key bits $S_n[0 \ldots \rho - 2]$ and let s_0 and s_1 denote the two candidates $0||S_n[0 \ldots \rho - 2]$ and $1||S_n[0 \ldots \rho - 2]$ respectively, where $||$ denotes string concatenation. A_{s_i}, for $i = 0, 1$, is a statistical variable which is supposed to be large for the correct key and small for a wrong key. The generic attack algorithm is as follows:

<u>**Generic 1-Bit Attack**</u>
Step 1: Compute A_{s_i} for $i = 0, 1$.
Step 2: If $A_{s_0} \geq A_{s_1}$ guess $S_n[\rho - 1] = 0$; otherwise guess $S_n[\rho - 1] = 1$.

Our 1-bit attacks are defined by their definition of the variable A_{s_i}. Let $\mathcal{S}_{s_i}$ denote the set of points in the $2^{\rho-2}$ neighborhood of s_i; i.e. the set defined by $\mathcal{S}_{s_i} = \{s : |s - s_i| \leq 2^{\rho-2}\}$:

Attack 1: $A_{s_i} = |B_{s_i}|$.
Attack 2: $A_{s_i} = |\sum_{s \in \mathcal{S}_{s_i}} B_s|$.
Attack 3: $A_{s_i} = \sum_{s \in \mathcal{S}_{s_i}} |B_s|$.
Attack 4: $A_{s_i} = \max_{s \in \mathcal{S}_{s_i}} \{|B_s|\}$.

Intuitively, Attack 1 simply compares the bias of s_0 and s_1. The other three attacks on the other hand, also use the biases of the points in the $2^{\rho-2}$ neighborhood of s_0 and s_1 (the choice of $2^{\rho-2}$ is because $s_0 + 2^{\rho-2}$ is the mid-point of s_0 and s_1). As will be discussed shortly, our experiments have shown that Attack 1 has the best success rate among the four.

We calculate the success rate of Attack 1 as

$$\int_{-\infty}^{\infty} \int_{-|x+2\sqrt{N}|p-1/2|\,|}^{|x+2\sqrt{N}|p-1/2|\,|} \frac{1}{\sqrt{2\pi}} e^{-y^2/2} dy \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx, \tag{4}$$

and the success rate of Attack 2 as

$$\int_{-\infty}^{\infty} \int_{-|x+1.83\sqrt{N}|p-1/2|\,|}^{|x+1.83\sqrt{N}|p-1/2|\,|} \frac{1}{\sqrt{2\pi}} e^{-y^2/2} dy \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx. \tag{5}$$

It is not straightforward to obtain a closed-form theoretical result for the success rate of Attacks 3 and 4. Therefore, we compared the attacks experimentally on RC5-16/2 on a sample of 10,000 different cases and for different values of ρ. The experimental results indicated that Attack 1 is the best among the four. For Attacks 1 and 2, the experimental results matched the theoretical results given in (4) and (5) very well.

5.1 A Generalization

Attacks 1 and 2 are special cases of a more general attack, which we call *Attack G*:

Let $\mathcal{S}_{s_i,d}$ denote the d neighborhood of s_i; i.e. $\mathcal{S}_{s_i,d} = \{s : |s - s_i| \leq d\}$. Attack G uses the generic attack algorithm with $A_{s_i} = |\sum_{s \in \mathcal{S}_{s_i,d}} (B_s)\,|$. Notice that this is the same as Attack 1 for $d = 0$ and the same as Attack 2 for $d = 2^{\rho-2}$.

We calculate the success rate of Attack G approximately as

$$\int_{-\infty}^{\infty} \int_{-|x+2(1-\frac{d}{3\cdot 2^{\rho}-2d})\sqrt{N}|p-1/2|\,|}^{|x+2(1-\frac{d}{3\cdot 2^{\rho}-2d})\sqrt{N}|p-1/2|\,|} \frac{1}{\sqrt{2\pi}} e^{-y^2/2} dy \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx,$$

which is maximized at $d = 0$. This result implies that Attack 1 has the highest success rate among all versions of Attack G.

5.2 An Improvement

A limitation of Attack 1, and also other 1-bit attacks discussed so far, is a zero bias (i.e. $p = 1/2$) which occurs for certain values of ρ, S_{n-1} and S_{n-2} (see Appendix A). In such cases, these attacks are no better than random guessing. One way to overcome this problem is to use the data with $L_n \bmod w > \rho$ as well as those with $L_n \bmod w = \rho$ to recover $S_n[\rho - 1]$.

We present such a modification of Attack 1, which we call *Attack 1′*. It uses the data with $L_n \bmod w = \rho + 1$ as well as the data with $L_n \bmod w = \rho$ to guess $S_n[\rho - 1]$:

As in Attack 1, let s_0 and s_1 denote the two candidates $0||S_n[0 \ldots \rho - 2]$ and $1||S_n[0 \ldots \rho - 2]$ respectively, and similarly let s_i denote $i||S_n[0 \ldots \rho - 2]$ for $i = 00, 01, 10, 11$. The idea of Attack 1′ is to compare the bias for four possible key candidates s_{00}, s_{01}, s_{10}, s_{11}. If the bias is maximized for s_{00} or s_{10}, we guess $S_n[\rho - 1]$ as 0, otherwise we guess it as 1. The calculation of the bias for these four points is as follows. As in Attack 1, B_{s_0} and B_{s_1} denote the bias for s_0 and s_1 taken over the data with $L_n \bmod w = \rho$. Similarly, B_{s_i} denotes the bias for s_i for $i = 00, 01, 10, 11$, *but taken over the data with* $L_n \bmod w =$

$\rho + 1$. We guess $S_n[\rho - 1] = 0$ if $|B_{s_0}| + \max\{|B_{s_{00}}|, |B_{s_{10}}|\}$ is greater than $|B_{s_1}| + \max\{|B_{s_{01}}|, |B_{s_{11}}|\}$. Otherwise, we guess $S_n[\rho - 1] = 1$. (This is the generic 1-bit attack algorithm where A_{s_i} is defined as $|B_{s_i}| + \max\{|B_{s_{0i}}|, |B_{s_{1i}}|\}$.)

We experimentally compared Attack 1 and Attack 1′ on RC5-16/2. The experimental success rates are given in Table 1. N denotes the available number of texts for each particular value of L_n mod w. The results show that Attack 1′ is significantly better than Attack 1.

N	1,000	4,000	10,000	40,000	100,000
Attack 1	79.8%	91.9%	96.2%	98.4%	99.1%
Attack 1′	86.8%	96.4%	98.0%	99.5%	99.9%

Table 1. Success rate of Attack 1 and 1′ on RC5-16/2 for recovering one bit of the last round key S_n. The experimental results show that Attack 1′ is significantly better than Attack 1.

6 Multi-bit Attack

The idea of the multi-bit attack is quite straightforward: Instead of fixing L_n mod w at each step, calculate the bias over the data with many different values of L_n mod w. We know that when L_n mod w is fixed, the behavior of the bias of a wrong key is not random (see Section 4). By taking the bias over many different values of L_n mod w, we hope that the bias will behave more "normally" (i.e. zero expected bias for a wrong key, positive expected bias for the correct key).

6.1 The Attack

Although the formal description of the multi-bit attack may appear complicated. In fact, it is really intuitive. Suppose we have already recovered the key bits $S_n[0 \ldots k]$ and we are going to recover the next ℓ bits $S_n[k+1 \ldots k+\ell]$. The bias for each key candidate is computed over the data with $k+1 \leq L_n \bmod w \leq k+\ell$. The one with the highest bias is accepted. The formal description of the algorithm is as follows:

U_s denotes the number of the texts such that the left side of Approximation (3) is 1 when we substitute s for S_n. The expression $S_n[0 \ldots k]$ denotes the part of the round key that has been recovered so far. ℓ denotes the number of the key bits that is attacked at one iteration of the algorithm. Once these ℓ bits are recovered, the algorithm is repeated for the next ℓ bits of S_n.

Attack M (k, ℓ)

Step 1: For $0 \leq i < 2^\ell$, compute $U_{i \| S_n[0 \ldots k]}$ over the data with $k+1 \leq L_n$ mod $w \leq k+\ell$.

Step 2: Accept i that maximizes the bias $|U_{i\|S_n[0\ldots k]} - N/2|$, where N is the number of data with $k+1 \leq L_n \bmod w \leq k+\ell$.

The choice of the parameter ℓ is a matter of trade-off. The computational complexity of the attack increases as ℓ gets larger. More specifically, the number of active text bits at an iteration of Attack M is $k+1+\ell$, and the number of active key bits is ℓ. Hence, the computational complexity of an iteration of Attack M is $2^{2\ell+k+1}$ (see Matsui [3]). Therefore, the computational complexity of recovering a round key of w bits is $2^{2\ell} \cdot \frac{2^w-1}{2^\ell-1}$, for ℓ dividing w. On the other hand, the reliability of the guesses also increases as ℓ gets larger, especially those of the low order bits, as will be shown in Section 6.2. Therefore, the value of ℓ should be decided with respect to the constraints of the available computational power, time and the desired success rate.

But Attack M has some limitations. For example, suppose we are trying to recover the key bits $S_n[k+1\ldots k+\ell]$, and let S'_n denote the key that is the same as S_n in every bit except for the $k+\ell^{th}$ one. The bias for S_n and S'_n taken over the data with $k+1 \leq L_n \bmod w < k+\ell$ will be exactly the same, since they are exactly the same at bits $0, 1, \ldots, k+\ell-1$. Taken over the data with $L_n \bmod w = k+\ell$, the bias for S'_n will be the inverse of the bias for S_n (see Section 4). Therefore, when the bias for $L_n \bmod w = k+\ell$ is negative (i.e. $p < 1/2$), we incorrectly deduce that S'_n is the correct key with very high probability! Similar arguments apply to the lower order bits as well, but their effect is less significant. This fact implies that the guesses for the higher order bits will not be very reliable, as illustrated by the experimental results in Section 6.2.

6.2 Experimental Results

We tested Attack M on RC5-16/2 for $\ell = 6, 8, 10$ on a sample of 10,000 different cases. Our results are given in Table 2. The entries in the tables are given as a percentage of the 10,000 trials. The i^{th} column of the tables denotes the percentage of guesses that are correct at the bits lower than i, but wrong at the i^{th} bit. Another important point about the tables is that, the data amount N denotes the available number of texts for *each* different value of $L_n \bmod w$ (e.g. for $\ell = 10$, the total number of texts used is $10 \times N$). We chose this way of presentation to make the comparison between the tables easier.

The experimental results show that the success rate for the lower order bits improves as ℓ increases. But this improvement becomes less significant for higher data amounts. Another important point is that increasing the data amount does not help beyond a certain point and the failure rates at the low order bits are almost stabilized around 0.8–0.9%.

The high failure rates at higher order bits are due to the effect discussed at the end of Section 6.1, and the success rate for these bits cannot be improved beyond a certain point, even with unlimited amount of data. Therefore, we suggest discarding the top two bits guessed, and starting the next iteration of the attack to include these bits as well. The size of the discarded part may be different for

$\ell = 6$	failure at bit					
N	0	1	2	3	4	5
1,000	8.4%	5.5%	5.0%	4.4%	5.4%	8.1%
10,000	1.6%	1.9%	2.4%	1.9%	4.1%	7.1%
100,000	1.2%	1.4%	1.6%	1.5%	3.4%	7.2%
1,000,000	0.9%	1.5%	1.7%	1.6%	3.6%	7.2%

$\ell = 8$	failure at bit							
N	0	1	2	3	4	5	6	7
1,000	7.1%	4.7%	3.9%	3.7%	3.8%	4.9%	6.0%	7.7%
10,000	1.4%	1.3%	1.4%	1.4%	1.9%	2.2%	4.1%	6.8%
100,000	0.9%	0.9%	0.8%	1.3%	1.5%	1.5%	3.4%	7.5%
1,000,000	0.9%	0.9%	0.9%	1.2%	1.1%	1.3%	3.4%	7.3%

$\ell = 10$	failure at bit									
N	0	1	2	3	4	5	6	7	8	9
1,000	5.6%	4.7%	4.3%	3.6%	3.8%	4.0%	3.8%	4.8%	5.6%	7.6%
10,000	1.1%	1.2%	1.3%	1.2%	1.4%	1.4%	1.6%	2.1%	3.7%	7.2%
100,000	0.9%	0.9%	1.0%	0.9%	0.9%	1.1%	1.4%	1.3%	3.6%	6.9%
1,000,000	0.8%	0.7%	0.9%	0.9%	0.9%	1.2%	1.1%	1.4%	3.2%	7.2%

Table 2. Failure rates of Attack M on RC5-16/2 for $\ell = 6, 8, 10$. The i^{th} column represents the percentage of guesses that are correct at the bits lower than i, but wrong at the i^{th} bit. The results show that the attack gets better as ℓ increases; but this improvement is less significant when the amount of data used is higher.

different word sizes, and should be determined experimentally (or theoretically if possible). We denote the size of the discarded part by j and add the following step to the algorithm Attack M:

Step 3: Discard the top j bits of the key estimated.

7 Comparison of 1-Bit and Multi-Bit Attacks

We compared the two attack strategies experimentally on RC5-16/r for $r = 2, 4$. Table 3 lists the success rates of Attack 1′ and Attack M for guessing the first eight bits of S_n. Attack 1′ represents the most successful 1-bit attack. N denotes the number of plaintexts available for each different value of $L_n \bmod w$. As discussed in Section 3, even though the experiments are run for relatively small values of r, they give a general comparison of the attacks, mainly because the relative performance of the attacks will not be affected much by an increase in r since they all use the same approximation.

The results suggest that Attack M has a better success rate for smaller amounts of data but Attack 1′ becomes better as the amount of available data increases. An advantage of Attack 1′ over Attack M is that the success rate

r	N	Attack $1'$	Attack M
2	1,000	32.1%	65.5%
	10,000	88.2%	88.7%
	100,000	98.3%	91.7%
	1,000,000	99.5%	92.1%
4	1,000	0.6%	0.7%
	10,000	0.5%	0.8%
	100,000	2.3%	6.5%
	1,000,000	20.3%	22.0%

Table 3. Success rates of Attack $1'$ and Attack M on RC5-16/r for recovering the first eight bits of S_n. The results show that Attack M is better for smaller amounts of available data. The success rates fall sharply as r increases.

of Attack M does not improve much beyond 92% regardless of the increase in the data amount. But there is no such limit on the success rate of Attack $1'$. Besides, the 1-bit attacks have two other advantages over the multi-bit attacks. First, they are computationally less expensive. Second, a wrong guess in a 1-bit attack can be detected earlier and can be corrected more easily since the biases after a wrong bit guess will be significantly smaller than what is expected.

The dramatic decrease in the success rates as the number of rounds r increases suggests that our attacks are not practical enough to break RC5 for larger values of r and w. This thought is also supported by the fact that all of our attacks are based on Approximation (1) which has a quite low bias for larger values of r and w. At this point, it is not possible to calculate the exact success rates for a given amount of data. This fact is due to the lack of a concrete formula for the relation between the probability of Approximation (1) and $L_n \bmod w$. However, we conjecture that the data requirement for a significant success rate will be comparable to $|p - 1/2|^{-2}$, that is $4w^{2r-2}$ which is impractically high for reasonably high values of w and r (i.e. $w \geq 32$, $r \geq 6$).

8 Conclusions

We presented some new results about linear cryptanalysis of RC5. First, we showed that the attack of Kaliski and Yin [1] does not work as expected due to some unexpected consequences of fixing $L_n \bmod w$. We studied the statistical behavior of Approximation (1). Then, we presented some new techniques of using this approximation to recover the last round key S_n.

Our results on the attack of Kaliski and Yin has significances beyond the linear cryptanalysis of RC5: It is significant to emphasize that hidden assumptions may have extremely serious consequences. It is also significant to show that extreme care has to be taken when applying a method developed for a specific cipher to a cipher of different type.

The attacks we presented in this paper are examples of how linear cryptanalysis can carry on when the bias is different from zero for a wrong key substituted in the approximation. At this point, it is not possible to calculate the exact success rate of our attacks due to the lack of a concrete formula for the relation between the probability of Approximation (1) and L_n mod w. However, we conjecture that the data requirement for a significant success rate will be comparable to $|p - 1/2|^{-2}$, which is impractically high for our approximation. Therefore, we believe that RC5 still remains secure against linear cryptanalysis.

There are many open research problems that are to be solved about the linear cryptanalysis of RC5. An important one is to obtain a theoretical result for the relation between the probability of Approximation (1), and $L_n - S_{n-1}$ and S_{n-2}. In this way, it will be possible to obtain theoretical results for the success rate of the attacks that are based on Approximation (1), including the ones presented in this paper. Moreover, it should be possible to use such a relation in an attack which obtains further information about the round keys S_{n-1} and S_{n-2}.

Another significant improvement will be to develop better linear cryptanalytic attacks than the ones presented here. However, any attack based on Approximation (1) will be limited by the low bias of that approximation. Therefore, finding a better approximation is essential to improving the linear cryptanalysis of RC5 significantly. But any researcher trying to find a better linear approximation should be aware of a proposition of Kaliski and Yin [1] that states a limitation of linear approximations of RC5.

A way to circumvent this limitation may be to use non-linear Approximations [2]; not just at the end rounds, but at the intermediate rounds as well. The main reason for using linear approximations in DES-like ciphers is that it is easy to find approximations of S-boxes since they are relatively small. Moreover, if an approximation of an S-box is linear it can be distributed to and stated in terms of the input, output and key bits of that round. But this argument is not true for RC5 since it does not have small sub-blocks like S-boxes. Moreover, using linear approximations does not have the advantage of being easily distributed to input, output and key bits as it is in DES. Therefore we believe that, at least theoretically, finding a non-linear approximation of RC5 is not substantially more difficult than finding a linear approximation. But it should be noted that finding an approximation for RC5 is not an easy task in general since there are no small sub-blocks like S-boxes.

As a last minute note, we have recently found out that the probability p of Approximation (1) is *not* equal to $\frac{1}{2} + \frac{1}{2w^{r-1}}$ which was calculated by Kaliski and Yin. The reason of this unexpected result is that the two consecutive half-round approximations $R_i[0] = L_{i-1}[0] \oplus S_i[0]$ and $R_{i+2}[0] = L_{i+1}[0] \oplus S_{i+2}[0]$ are not independent, and therefore, the piling-up lemma cannot be used to calculate the probability of Approximation (1). We experimentally found out that the probability p is extremely key dependent and it can be a lot different from $\frac{1}{2} + \frac{1}{2w^{r-1}}$ depending on the key. Even when averaged over the keys, the probability of Approximation (1) is a lot different from $\frac{1}{2} + \frac{1}{2w^{r-1}}$. Now, this new finding leaves the whole issue of linear cryptanalysis of RC5 as an open question.

Acknowledgments

I gratefully acknowledge the invaluable discussions with Matt Robshaw and Lisa Yin. I would like to thank to Alan Sherman for his comments on the paper and I would like to thank to Leo Reyzin for his help with the implementation of the attacks. I am thankful to all my friends at the RSA Labs for their support and friendship during my visit there.

References

1. B. S. Kaliski and Y. L. Yin. On differential and linear cryptanalysis of the RC5 encryption algorithm. In D. Coppersmith, editor, *Advances in Cryptology — Crypto'95*, pages 171–184. Springer Verlag, New York, 1995.
2. L. Knudsen and M. Robshaw. Non-linear approximations in linear cryptanalysis. In U. Maurer, editor, *Advances in Cryptology — Eurocrypt'96*, pages 224–236. Springer-Verlag, New York, 1996.
3. M. Matsui. Linear cryptanalysis of DES cipher (I). *Journal of Cryptology*, to appear.
4. M. Matsui. Linear cryptanalysis method for DES cipher. In T. Helleseth, editor, *Advances in Cryptology — Eurocrypt'93*, pages 386–397. Springer-Verlag, Berlin, 1994.
5. R. Rivest. The RC5 encryption algorithm. In *Fast Software Encryption, Second International Workshop*, pages 86–96. Springer-Verlag, New York, 1995.

A Bias of the Approximation

The probability of the half-round approximation

$$R_{n-2}[0] = L_{n-3}[0] \oplus S_{n-2}[0] \tag{6}$$

depends on the value of $(L_n - S_{n-1}) \bmod w$ and $S_{n-2} \bmod w$. This fact implies that when L_n, S_{n-1} and S_{n-2} are fixed, the bias of the approximation may be different from its average bias $1/2w$. Table 4 lists the bias of Approximation (6) for different values of $(L_n - S_{n-1}) \bmod w$ and $S_{n-2} \bmod w$ for $w = 16$. These values are computed by exhaustively going through all possible values of (L_n, R_n). The parameter δ denotes the difference $(L_n - S_{n-1}) \bmod w$. The entries of the tables are the actual biases as a proportion of the average bias $1/2w$ (i.e. $(p - 1/2)/(1/2w)$).

	S_{n-2} mod w							
δ	0	1	2	3	4	5	6	7
0	2.25	0.75	1.75	2.13	1.63	1.50	0.75	1.88
1	0.50	-1.25	1.50	2.63	-0.38	-0.25	1.00	1.88
2	2.25	1.75	0.75	2.13	-0.38	0.50	1.75	-0.13
3	1.00	0.25	2.50	0.13	2.63	0.75	2.50	0.38
4	0.75	0.25	0.25	0.63	0.13	1.00	0.25	0.88
5	1.00	0.25	0.00	1.13	0.13	1.25	0.50	0.88
6	-0.25	1.25	1.25	-0.38	0.13	1.00	-0.75	1.88
7	0.50	2.75	2.00	-0.38	2.13	1.25	2.00	1.38
8	0.75	1.00	1.00	1.13	1.13	0.75	1.00	0.88
9	1.00	1.00	0.75	1.63	1.13	1.00	1.25	0.88
10	0.75	1.00	1.00	1.13	1.13	0.75	1.00	0.88
11	1.50	1.50	0.75	1.13	2.13	1.00	1.75	1.38
12	0.75	1.00	0.50	0.63	1.13	0.75	0.50	0.38
13	1.00	1.00	0.25	1.13	1.13	1.00	0.75	0.38
14	0.75	1.00	0.50	0.63	0.13	1.75	-0.50	1.38
15	1.50	2.50	1.25	0.63	2.13	2.00	2.25	0.88

	S_{n-2} mod w							
δ	8	9	10	11	12	13	14	15
0	1.88	0.88	1.38	2.25	1.75	1.38	1.13	1.75
1	0.38	-1.38	1.38	2.50	0.00	-0.63	1.13	2.00
2	1.88	1.88	0.38	2.25	-0.25	0.38	2.13	-0.25
3	2.38	2.63	-0.13	2.50	0.50	2.88	0.13	2.00
4	-0.13	0.88	1.38	0.25	0.75	0.38	1.13	0.25
5	0.38	0.63	1.38	0.50	1.00	0.38	1.13	0.50
6	0.88	-0.13	2.38	-0.75	0.75	0.38	0.13	1.25
7	0.38	2.63	1.88	0.50	1.50	1.88	1.13	1.50
8	0.63	0.88	0.88	1.00	1.00	0.88	1.13	1.00
9	1.13	0.63	0.88	1.25	1.25	0.88	1.13	1.25
10	0.63	0.88	0.88	1.00	1.00	0.88	1.13	1.00
11	1.13	1.63	0.38	1.25	1.75	1.38	1.13	1.25
12	0.63	0.88	0.38	0.50	1.00	0.88	0.63	0.50
13	1.13	0.63	0.38	0.75	1.25	0.88	0.63	0.75
14	1.63	-0.13	1.38	-0.50	1.00	0.88	0.63	0.50
15	1.13	2.63	0.88	0.75	1.75	2.38	1.63	0.75

Table 4. Bias of the $n-2^{nd}$ half-round approximation as a proportion of the expected bias $1/2w$ when L_n mod w is fixed, for $w = 16$. The variable δ denotes the difference $(L_n - S_{n-1})$ mod w. The numbers in the table show that how big an impact the hidden assumptions had on the bias of the $n-2^{nd}$ half-round approximation.