A Timing Attack on RC5

Howard M. Heys

Faculty of Engineering and Applied Science
Memorial University of Newfoundland
St. John's, Newfoundland, Canada A1B 3X5
email: howard@engr.mun.ca

Abstract: In this paper, we examine the application of a timing attack to the RC5 symmetric block cipher. The analysis is motivated by the possibility that a naive implementation of RC5 could result in the data-dependent rotations taking a time that is a function of the data. Such implementations are possible, for example, in a digital hardware or 8-bit microcontroller environment. Based on the assumption that accurate timing measurements are available for individual encryptions, the methodology of deriving key bits using timing information from a set of ciphertexts is outlined and it is shown that, for the nominal version of RC5, with only about 2000 ciphertexts and their associated timings, the cryptanalysis can be applied to determine 5 bits of the last half-round subkey with high probability. Further, using a set of about 64000 random ciphertexts, the attack can be applied to determine the entire subkey of the last half-round with high probability and, if 10^6 random ciphertexts and their timings are available, the attack has a significant likelihood of breaking the cipher by determining all subkeys of the cipher.

I. Introduction

The RC5 encryption algorithm [1] was introduced by Rivest as a symmetric block cipher designed to be efficiently implemented in software by making use of three basic operations: exclusive-OR, addition, and data-dependent rotations. An RC5 cipher is designated as RC5-$w/r/b$ where w represents the word size of the target processor in bits, r is the number of rounds of the cipher, and b is the number of bytes of key. The block size of the cipher is fixed at $2w$ bits. We shall focus the discussion in this paper on the nominal

version of the cipher, RC5-32/12/16, which has a 64-bit block size, an 80-bit key, and 12 rounds.

The application of the two powerful attacks of differential and linear cryptanalysis to RC5 is considered by Kaliski and Yin [2], who show that the 12-round nominal cipher appears to be secure against both attacks. In [3], Knudsen and Meier extend the analysis of the differential attacks of RC5 and show that, by searching for appropriate plaintexts to use, the complexity of the attack can be reduced by a factor of up to 512 for a typical key of the nominal RC5. As well, it is shown that keys exist which make RC5 even weaker against differential cryptanalysis. Recently, in [4], new differential cryptanalysis results imply that 16 rounds are required for the cipher with $w = 32$ to be secure. The results on linear cryptanalysis are refined by Selcuk in [5] and, in [6], it is shown that a small fraction of keys results in significant susceptibility to linear cryptanalysis. Despite these results, from a practical perspective RC5 seems to be secure against both differential and linear cryptanalysis.

In [7], Kocher introduces the general notion of a timing attack. The attack attempts to reveal the key by making use of information on the time it takes to encrypt. The applicability of the attack on asymmetric systems is demonstrated by examining timing variations for modular exponentiation operations. As noted by Kocher, implementations of RC5 on processors which do not execute the rotation in constant time are at risk from timing attacks.

In this paper, we examine the application of timing attacks to RC5 by considering an implementation for which the time it takes to encrypt (or decrypt) is non-constant and is a function of the data in each round. Although software implementations of RC5 would typically use processors for which the data-dependent rotation is performed in constant time, a naive digital hardware or 8-bit microcontroller implementation of RC5 could result in rotations with timing dependent on the size of the rotation. In such circumstances, timing attacks are applicable to RC5 and, as we shall demonstrate, have the potential to seriously compromise the security of the cipher.

II. Description of the Cipher

In our description of the cipher, all words are represented with the most significant bit at the left: the $(i+1)$-th least significant bit of a word X is represented by $X[i]$ and the block of bits from the $(i+1)$-th down to the $(j+1)$-th least significant bit, $i \geq j$, is represented by $X[i \ldots j]$.

The RC5 algorithm consists of r rounds involving the application of $2r+2$ subkeys. Alternatively, the cipher can be viewed as the mixing of 2 subkeys with the plaintext, followed by $2r$ half-rounds. Let L_0 and R_0 represent the left and right half of the plaintext input, respectively, each half consisting of w bits. We use the notation L_k and R_k to represent the left and right half, respectively, of the cipher data at the input to the k-th half-round. Also, S_{k+1} represents the $(k+1)$-th subkey consisting of w bits associated with the k-th half-round and generated by the key scheduling algorithm as detailed in [1]. Letting L_{2r+1} and R_{2r+1} represent the left and right half of the ciphertext, respectively, the RC5 encryption algorithm is given by:

$$\begin{aligned} &L_1 = L_0 + S_0 \\ &R_1 = R_0 + S_1 \\ &\text{for } k = 1 \text{ to } 2r \text{ do} \\ &\quad L_{k+1} = R_k \\ &\quad R_{k+1} = ((L_k \oplus R_k) \leftarrow R_k) + S_{k+1} \end{aligned} \tag{1}$$

where "+" represents addition modulo-2^w, "$\oplus$" represents bit-wise exclusive-OR, and "$X \leftarrow Y$" is the rotation of X to the left by the $\log_2 w$ least significant bits of Y. For example, if $w = 32$, X is rotated to the left the number of positions indicated by the value of $Y[4...0]$.

To decrypt, the operations of the algorithm must be appropriately reversed to generate the data for each half-round by essentially going backwards through the encryption algorithm. For example, data is rotated right and the subkeys are applied by subtracting modulo-2^w from the data.

III. Basic Principles of the Attack

The timing attack of RC5 is applicable to implementations where the data-dependent rotation is executed in non-constant time and where it is possible to measure accurately

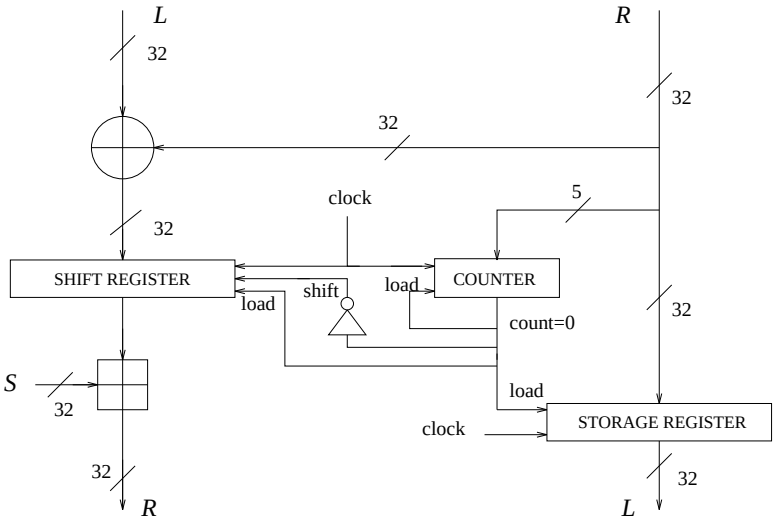

Figure 1: Naive Hardware Implementation of RC5 Half-Round

the timing associated with each encyption. For example, this is possible for an implementation of the cipher in synchronous, sequential digital logic hardware or for an implementation on an 8-bit microcontroller such as found on smartcards. Comments on the applicability of the attack to an 8-bit microcontroller are given in Section VII. In the remainder of the paper, we shall base our discussion on the implementation of the cipher in digital hardware.

Consider the block diagram of one potential hardware implementation of the nominal version of the algorithm as shown in Figure 1. The diagram represents a straightforward synchronous, sequential logic implementation of an RC5 half-round where in the k-th half-round the shift register contains the value of $L_k \oplus R_k$, the storage register contains the value of R_k, and the counter contains $R_k[4 \ldots 0]$. The rotation operation of the half-round is executed by shifting the data in the shift register by one bit in each clock cycle until the appropriate number of shifts is accomplished. The number of shifts is determined by decrementing the counter once each clock cycle until the counter is decremented to zero. When the counter is decremented to zero, the registers and counter are loaded with new values corresponding to half-round $k + 1$. The design represented in the diagram is not efficient - the rotations could be implemented using a barrel shifter to shift any number of bits in one clock cycle - but it does convey a simple, straightforward mapping of the algorithm into hardware. However, as we shall see, the time-varying nature of

the rotations is clearly unsound from the perspective of cipher security and should be avoided.

For the implementation of Figure 1, the total time to encrypt is directly proportional to the total number of clock cycles required to setup data for the first half-round, retrieve data from the last half-round, and execute 24 half-rounds. Hence, the number of clock cycles is of a format

$$N_{cycle} = a + N_{rot} \tag{2}$$

where a is a constant and N_{rot} is the total number of single-bit rotations (i.e., shifts of the shift register) required for the encryption. (Note that we have excluded the time required to generate subkeys, assuming that these are generated once and then saved for use with each encryption.) For the 12-round nominal cipher, N_{rot} is given by

$$N_{rot} = \sum_{k=1}^{24} \eta_k \tag{3}$$

where η_k represents the size of the rotation in bits required in half-round k and may be viewed as a random variable determined by the data at half-round k. For the nominal version of RC5, $0 \leq \eta_k \leq 31$ and, if it is assumed that all values are equally likely, the expected value of η_k is $\mu_\eta = 15.5$ and the variance of η_k is given by $\sigma_\eta^2 = 85.25$.

Consider now how information about timing can be used to extract key bits from the cipher. For simplicity, assume that each η_k is deterministic with a value of α. Assume that the cryptanalyst acquires a ciphertext (and the associated timing of the encryption) which has the 5 least significant bits of the left half to be all zero, i.e., $L_{25}[4 \ldots 0] = 0$. We can expect that the cryptanalyst knows the structure of the implementation and the period of the clock. Hence, the ciphertext timing information can be used to determine N_{cycle} and the effects of the constant a may be factored out. The cryptanalyst can then work with knowledge of the number of rotations, N_{rot}, for each acquired ciphertext. If it is known that the first 22 rounds takes 22α rotations and the last round has a rotation value of 0, then the size of the rotation for half-round 23 is $\eta_{23} = N_{rot} - 22\alpha$. However, it can seen that

$$\eta_{23} = R_{25}[4 \ldots 0] - S_{25}[4 \ldots 0] \tag{4}$$

where the subtraction is modulo-32 and $R_{25}[4\ldots0]$ is the 5 least significant bits of the ciphertext right half. Hence, $S_{25}[4\ldots0] = R_{25}[4\ldots0] - (N_{rot} - 22\alpha)$ and 5 bits of the last half-round subkey are determined with just one ciphertext and its timing information.

Of course, it is an artificial argument to consider the size of the rotation in each half-round of the cipher to be deterministic when it is actually a random variable and the actual sum of the rotations of the first 22 half-rounds is unknown to the cryptanalyst. In the next section, we consider a probabilistic model for deriving the expected value and variance of the timing for the first 22 half-rounds of the cipher.

IV. Encryption Timing Model

Since the total number of single-bit rotations for the first 22 half-rounds of the cipher is random, we now consider a probabilistic model which is useful in understanding the statistical nature of the attack. We use the model to get an estimate of the number of ciphertexts required to determine, with a high probability, the 5 bits of subkey represented as $S_{25}[4\ldots0]$. The model assumes that

- the number of single-bit rotations in round k, $\eta_k \in \{0, 1, 2, \ldots, 31\}$, is a uniformly distributed random variable and
- the rotations in different rounds are independent.

Under these assumptions, the sum of the number of rotations for the first 22 rounds of the cipher, which we shall represent as $v = \sum_{k=1}^{22} \eta_k$, is a random variable with a mean of $\mu_v = 22 \cdot \mu_\eta = 341$ and variance of $\sigma_v^2 = 22 \cdot \sigma_\eta^2 = 1875.5$. As well, based on the central limit theorem, v is approximately Gaussian distributed.

An effective way to determine the correct value of the partial subkey $S_{25}[4\ldots0]$ is to use a number of ciphertexts to test each possible value for the 5 bits and to determine which value is most consistent with the expected statistical distribution of the timing information. One method to do this would be to use a number of ciphertexts for which $L_{25}[4\ldots0] = 0$ to compute an estimate of the variance of v based on the value of each candidate partial subkey: it is expected that the variance estimate when the correct value

for the partial subkey is selected will be smaller than the estimate when an incorrect partial subkey is assumed. Let K represent the actual key bits $S_{25}[4\ldots0]$ and let $\tilde{K}$ represent the guess of the partial subkey K. The candidate key $\tilde{K}$ can be represented by

$$\tilde{K} = (K + \tau) \bmod 32 \tag{5}$$

where $-15 \leq \tau \leq 16$. The estimate of the variance of v for a particular candidate key $\tilde{K}$ is given by

$$\phi_\tau = E\left\{e_{\tau,x}^2\right\} \tag{6}$$

where $e_{\tau,x}$ represents the difference between the measured number of rotations for the entire 24 rounds and the expected number of rotations given the assumed candidate key for a ciphertext with $R_{25}[4\ldots0] = x$. The difference $e_{\tau,x}$ is composed as follows:

$$e_{\tau,x} = (v + r) - (\mu_v + \tilde{r}_{\tau,x}) \tag{7}$$

where r represents the actual value of the rotation in the 23rd half-round (i.e., $r = \eta_{23}$) and $\tilde{r}_{\tau,x}$ represents the guess of the rotation in the 23rd half-round (corresponding to the candidate key $\tilde{K}$) given $R_{25}[4\ldots0] = x$. Using ciphertexts for which $L_{25}[4\ldots0] = 0$, the size of the rotation in the 24th half-round is 0. Therefore, $v + r = N_{rot}$ and N_{rot} can be derived from the ciphertext timing information. The value of $\tilde{r}_{\tau,x}$ is determined from $\tilde{r}_{\tau,x} = (x - \tilde{K}) \bmod 32$. Hence, for a given ciphertext and candidate key, the value of $e_{\tau,x}$ can be calculated. We can also view equation (7) by letting $\Delta v = v - \mu_v$ and $\Delta r_{\tau,x} = r - \tilde{r}_{\tau,x}$ and we get

$$e_{\tau,x} = \Delta v + \Delta r_{\tau,x}. \tag{8}$$

Now assume that the cryptanalyst has available n ciphertexts for which $L_{25}[4\ldots0] = 0$ and, hence, for large n, the number of ciphertexts corresponding to a particular value x for $R_{25}[4\ldots0]$ is given by $n_x \approx n/32$. For each ciphertext and candidate key $\tilde{K}$, $e_{\tau,x}$ is computed and the mean of the square of $e_{\tau,x}$ is calculated. The result is equivalent to

$$\phi_\tau = \frac{1}{n}\sum_{x=0}^{31}\sum_{i=1}^{n_x}[\Delta v_{x,i} + \Delta r_{\tau,x}]^2 \tag{9}$$

where $\Delta v_{x,i}$ represents the i-th value of Δv for $R_{25}[4\ldots0] = x$. For the correct guess of the key (i.e., $\tau = 0$), $\Delta r_{\tau,x} = 0$ since $\tilde{r}_{\tau,x} = r$ and, hence, $\phi_0 = (1/n)\sum_{x=0}^{31}\sum_{i=1}^{n_x}(\Delta v_{x,i})^2$.

For incorrect candidate keys, for which $|\tau| \geq 1$, $\Delta r_{\tau,x} \neq 0$, and it can be shown that $E\{\phi_\tau\} > E\{\phi_0\}$. The cryptanalyst can therefore collect ciphertexts and timing information (implying rotation information) and determine the key K by picking the candidate key $\tilde{K}$ which minimizes ϕ_τ.

We now determine the probability that an incorrect key is selected over the correct key. For this to happen, we must have $\phi_\tau < \phi_0$ or, alternatively, $\phi_\tau - \phi_0 < 0$. Hence, the cryptanalyst must acquire enough ciphertext timings to ensure that $\phi_\tau - \phi_0 > 0$ for all $\tau \neq 0$. From (9), it can be seen that

$$\phi_\tau - \phi_0 = \frac{1}{n} \sum_{x=0}^{31} \sum_{i=1}^{n_x} [2\Delta r_{\tau,x} \Delta v_{x,i} + (\Delta r_{\tau,x})^2]. \tag{10}$$

We can consider $\phi_\tau - \phi_0$ to be a Gaussian distributed random variable with an expected value given by

$$E\{\phi_\tau - \phi_0\} \approx \frac{1}{32} \sum_{x=0}^{31} (\Delta r_{\tau,x})^2 \tag{11}$$

where we have used $n_x \approx n/32$ and $E\{\Delta v\} = 0$. As well, $\phi_\tau - \phi_0$ has a variance given by

$$Var\{\phi_\tau - \phi_0\} = \frac{1}{n^2} \sum_{x=0}^{31} \sum_{i=1}^{n_x} [(2\Delta r_{\tau,x})^2 \sigma_v^2] \approx \frac{\sigma_v^2}{8n} \sum_{x=0}^{31} (\Delta r_{\tau,x})^2. \tag{12}$$

It can be shown that

$$\sum_{x=0}^{31} (\Delta r_{\tau,x})^2 = |\tau|(32 - |\tau|)^2 + (32 - |\tau|)|\tau|^2 = 32|\tau|(32 - |\tau|) \tag{13}$$

and, consequently, it can be easily verified that

$$\max_\tau P(\phi_\tau - \phi_0 < 0) = P(\phi_\omega - \phi_0 < 0) \tag{14}$$

where $\omega = -1$ or $+1$. For $n = 2000$ ciphertexts for which $L_{25}[4 \ldots 0] = 0$, based on the Gaussian distribution, we get $P(\phi_1 - \phi_0 < 0) = 0.0021$ and the probability of being able to determine the correct 5 bits of subkey, $S_{25}[4 \ldots 0]$, is given by

$$P(S_{25}[4 \ldots 0] \text{ correct}) = 1 - P(\exists \tilde{K} | \tau \neq 0, \phi_\tau < \phi_0) \tag{15}$$

where

$$P(\exists \tilde{K} | \tau \neq 0, \phi_\tau < \phi_0) < 31 \cdot P(\phi_1 - \phi_0 < 0) = 0.0651. \tag{16}$$

Therefore, the probability of picking the correct 5 bits of subkey is greater than 93.5% with 2000 ciphertexts under the assumption that the rotations in all rounds are independent. Note that the ciphertexts must be chosen such that $L_{25}[4 \ldots 0] = 0$, which is true on average for 1 in 32 random ciphertexts. Hence, the correct 5 bits of subkey can be derived with high probability using about 64000 random ciphertexts and their timings.

As we shall see in Section VI, in fact, there are some dependencies in the rotations of different rounds which result in the probabilities of successfully deriving the key in practice being somewhat lower than expected from the model. Nevertheless, experimental evidence confirms that the approach works well when applied to the actual cipher.

V. Determining All Subkey Bits

In the previous sections, we illustrated how it is possible to determine 5 bits of the last half-round subkey S_{25} with high probability using a set of random ciphertexts and their timing information. Fortunately, it is straightforward to apply the techniques on the same ciphertexts to determine the remaining bits of subkey S_{25} and, with enough ciphertexts, it is possible to derive all the bits of the subkeys $S_k, 3 \leq k \leq 24$, as well.

Consider first the derivation of all subkey bits of the final half-round. To determine the subkey bits $S_{25}[9 \ldots 5]$, the cryptanalyst must use ciphertexts for which $L_{25}[4 \ldots 0] = 5$ so that the key bits of interest are involved in determining the rotation of half-round 23. In this case, the total number of rotations is given by

$$N_{rot} = \sum_{k=1}^{22} \eta_k + \eta_{23} + 5 \tag{17}$$

where $\eta_{23} = M[4 \ldots 0] \oplus 5$ for $M = [(R_{25}[9 \ldots 0] - S_{25}[9 \ldots 0]) \rightarrow 5]$ with "$X \rightarrow Y$" representing the rotation of X by Y positions to the right and the subtraction is taken modulo-2^{10}. Assuming the key bits $S_{25}[4 \ldots 0]$ are already known, it is not difficult to apply the techniques of the previous section to determine the value of the bits $S_{25}[9 \ldots 5]$ which minimize the computed ϕ_τ. In this case, however, the candidate key $\tilde{K}$ is a guess of $S_{25}[9 \ldots 5]$ and the guess for η_{23} corresponding to $\tilde{K}$, which we now represent as $\tilde{r}_\tau$, is determined by computing M_τ where M_τ is the value of M corresponding to the candidate key $\tilde{K}$. The value of $\tilde{r}_\tau$ is then given by $M_\tau[4 \ldots 0] \oplus 5$.

In general, the techniques can be applied to determine $S_{25}[i+4 \ldots i]$ using the previously determined key bits $S_{25}[i-1 \ldots 0]$ where ciphertexts are used which have $L_{25}[4 \ldots 0] = i$ and $\tilde{r}_\tau = M_\tau[4 \ldots 0] \oplus i$. The value of M_τ is given by

$$M_\tau = [(R_{25}[i+4 \ldots 0] - \tilde{S}_{25}[i+4 \ldots 0]) \rightarrow i] \tag{18}$$

where subtraction is modulo-2^{i+5}, $\tilde{S}_{25}[i-1 \ldots 0] = S_{25}[i-1 \ldots 0]$, and $\tilde{S}_{25}[i+4 \ldots i] = \tilde{K}$. To determine S_{25}, for each value of i, $i \in \{5, 10, 15, 20, 25\}$, ϕ_τ is calculated using the measured N_{rot} and $\tilde{r}_\tau$. The candidate key $\tilde{K}$ for which the minimum value occurs is selected as the correct partial subkey $S_{25}[i+4 \ldots i]$. This determines 30 bits of the subkey, $S_{25}[29 \ldots 0]$, and the last 2 bits of S_{25} can be derived by determining the bits $S_{25}[31 \ldots 27]$ using ciphertexts for which $L_{25}[4 \ldots 0] = 27$.

The set of random ciphertexts required to determine the first 5 bits can be used to derive the complete subkey S_{25}. Each partial subkey $S_{25}[i+4 \ldots i]$, $i \in \{5, 10, 15, 20, 25\}$, is attacked by selecting from the set ciphertexts with $L_{25}[4 \ldots 0] = i$. It can be shown that, under the assumptions of the model, 2000 ciphertexts with the appropriate value for $L_{25}[4 \ldots 0]$ are sufficient to determine any partial subkey with high probability and the set of 64000 random ciphertexts used to determine $S_{25}[4 \ldots 0]$ can be used to derive the subset of about 2000 ciphertexts needed to determine any partial subkey.

Once S_{25} is derived, the remaining subkeys associated with each half-round k, $2 \leq k \leq 23$, may be determined using the same set of ciphertexts. Once the subkey for a round k is determined, the ciphertext may be partially decrypted for one round so that the output of round $k-1$ is known. Correspondingly, the timing of the partial encryption of the first $k-1$ rounds may be determined by subtracting the time to execute the k-th round from the time to encrypt the first k rounds of the cipher. The new ciphertext and timing information may then be used to extract the subkey for round $k-1$ in exactly the same manner as for the subkey for round k.

The remaining subkeys, S_0, S_1, and S_2, are applied to the cipher by addition to the plaintext left half, plaintext right half, and the output of the rotation operation in the first half-round. All three of these subkeys cannot be determined using timing information but are trivially determined using only a modest number of known plaintexts and ciphertexts:

Number of Random Ciphertexts	*Probability of Success*		
	$S_{25}[4\ldots0]$	S_{25}	$S_3\ldots S_{25}$
10^4	0.611	0.083	0.000
10^5	0.893	0.794	0.009
10^6	0.901	0.827	0.024

Table 1: Experimental Results for 1000 Keys

S_1 is simply determined using one known plaintext and using the relationship $S_1 = L_2 - R_0$, S_2 can be determined with a modest number of known plaintexts using, for example, linear cryptanalysis [2], and S_0 can be easily derived once S_1 and S_2 are determined using $S_0 = [((R_2 - S_2) \rightarrow L_2) \oplus L_2] - L_0$.

VI. Experimental Results

In this section we present the experimental results which validate the effectiveness of the attack. The model of the previous sections assumes that the values of the rotations in different rounds are independent. This assumption, however, is not strictly correct. Consider, for example, the following scenario: $S_{24} = 0$ and $S_{25}[4\ldots0] = 0$. Suppose the cryptanalyst is attempting to determine $S_{25}[4\ldots0]$ and is therefore considering ciphertexts for which $L_{25}[4\ldots0] = 0$. If $R_{25}[4\ldots0] = x = 16$, then $\eta_{22} = R_{22}[4\ldots0] = L_{25}[20\ldots16] \oplus 16$ and, since $L_{25}[20\ldots16]$ is a uniformly distributed random variable, η_{22} behaves as anticipated by the model. However, if $R_{25}[4\ldots0] = x = 0$, then $\eta_{22} = R_{22}[4\ldots0] = 0$ and η_{22} is not a uniformly distributed random variable as suggested by the model.

These discrepancies from the model add inaccuracies to the process of statistically deriving the subkeys. However, experimental results demonstrate that the cryptanalytic technique is still very effective and the model provides a rough approximation of the effectiveness of the attack to determine 5 key bits of the last half-round subkey. For 2000 ciphertexts with $L_{25}[4\ldots0] = 0$ (equivalent to about 64000 random ciphertexts), experiments on the nominal RC5 for 1000 random keys resulted in an 86.2% chance of the partial subkey $S_{25}[4\ldots0]$ being correctly determined. Using 64000 random ciphertexts, the complete subkey S_{25} was correctly determined for 69.7% of the keys.

The effectiveness of the timing attack as determined in experiments is further illustrated in Table 1. It is clear from the table that few random ciphertexts are required to determine the bits of the last half-round subkey with a high probability. The correct derivation of all subkeys $S_3 \ldots S_{25}$ does not occur with nearly as high a probability: even modest deviations in probability from 1 when determining subkeys significantly reduces the probability that all 23 subkeys will be successfully determined. However, it is apparent that the attack can be very effective in determining subkeys for a large fraction of keys and should be seriously considered to ensure that an implementation of the cipher is not vulnerable.

VII. Applicability of the Attack to 8-bit Microcontrollers

To this point, we have focussed the discussion of the attack on a digital hardware implementation of RC5. However, for other environments where the rotations are not performed in constant time, the attack is also applicable. One such environment is an assembly language implementation on an 8-bit microcontroller such as the Motorola M68HC05, a popular processor used on smartcards.

For the M68HC05, any shifting of a byte must be performed by shifting one bit at a time. Hence, all operations of RC5 could easily be implemented to be constant in time, except for the data-dependent rotation. A straightforward implementation of the 32-bit rotation using 8-bit words would shift bytes one bit at a time using the carry bit to move the most significant bit from one byte to the least significant bit of the next byte. In such an implementation, it can be assumed that the k-th half-round operation will take a time (in terms of number of clock cycles) that can be represented as:

$$N_{hr} = c + d \cdot \eta_k \tag{19}$$

where c and d are constants and η_k is the integer value of $R_k[4 \ldots 0]$. Therefore, the total number of clock cycles required to encrypt is given by an expression

$$N_{cycle} = a + b \cdot N_{rot} \tag{20}$$

where a and b are constants and N_{rot} represents the total number of byte rotations (by one bit) for all half-rounds and is dependent on the 5 least significant bits of the right half

of the data in each half-round. Note the similarity to (2). The values of a and b can be determined if detailed information of the implementation is known. As a result, the only variable is the number of rotations which can be determined by an accurate measurement of N_{cycle}. Hence, using techniques similar to those outlined in the previous sections, the 5 least significant bits of the last half-round subkey can be determined. Subsequently, similarly to Section V, the last half-round subkey can be fully determined.

VIII. Conclusion

In this paper, we have demonstrated the potential effectiveness of timing attacks on RC5 implementations where the time to encrypt is proportional to the sum of the rotational values in each half-round. Such a situation could easily result from a naive, straightforward implementation of the cipher in digital hardware or an implementation on an 8-bit microcontroller which executes byte-wise rotations one bit at a time.

Admittedly, the cryptanalysis presented in this paper assumes accurate timing measurements of individual encryptions - an assumption which would likely be difficult to achieve in practice. However, the analysis clearly demonstrates the susceptibility of some implementations to cryptanalysis if such measurements could be made and suggests that it is vital for designers to be aware of the cryptographic issues when implementing RC5. Fortunately, the attack can be thwarted by ensuring that the rotations operate in constant time. For example, in the case of the digital hardware implementation, a barrel shifter could be used to perform any size rotation in one clock cycle.

References

[1] R.L. Rivest, "The RC5 Encryption Algorithm", *Proceedings of Fast Software Encryption 1994*, Springer-Verlag, pp. 86-96, 1995.

[2] B.S. Kaliski and Y.L. Yin, "On Differential and Linear Cryptanalysis of the RC5 Encryption Algorithm", *Advances in Cryptology - CRYPTO '95*, Springer-Verlag, pp. 171-184, 1995.

[3] L.R. Knudsen and W. Meier, "Improved Differential Attacks on RC5", *Advances in Cryptology - CRYPTO '96*, Springer-Verlag, pp. 216-228, 1996.

[4] A. Biryukov and E. Kushilevitz, "Improved Cryptanalysis of RC5", *Advances in Cryptology - EUROCRYPT '98*, Springer-Verlag, pp. 85-99, 1998.

[5] A.A. Selcuk, "New Results in Linear Cryptanalysis of RC5", *Proceedings of Fast Software Encryption 1998*, Springer-Verlag, pp. 1-16, 1998.

[6] H.M. Heys, "Linearly Weak Keys of RC5", *IEE Electronics Letters*, vol. 33, no. 10, pp. 836-837, 1997.

[7] P.C. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems", *Advances in Cryptology - CRYPTO '96*, Springer-Verlag, pp. 104-113, 1996.