On the Design and Security of RC2

Lars R. Knudsen[1], Vincent Rijmen[2], Ronald L. Rivest[3], and M.J.B. Robshaw[4]

[1] Dept. of Informatics, University of Bergen, Hi-techcenter, N-5020 Bergen, Norway
larsr@ii.uib.no
[2] K.U. Leuven, ESAT, Kardinaal Mercierlaan 94, B-3001 Heverlee, Belgium
vincent.rijmen@esat.kuleuven.ac.be
[3] M.I.T. Laboratory for Computer Science, 545 Technology Square, Cambridge, MA 02139, USA
rivest@theory.lcs.mit.edu
[4] RSA Laboratories, 100 Marine Parkway, Redwood City, CA 94065, USA
matt@rsa.com

Abstract. The block cipher RC2 was designed in 1989 by Ron Rivest for RSA Data Security Inc. In this paper we describe both the cipher and preliminary attempts to use both differential and linear cryptanalysis.

1 Introduction

RC2 is a block cipher[5] that was designed in 1989 by Ron Rivest for RSA Data Security, Inc. Initially held as a confidential and proprietary algorithm, RC2 was published as an Internet Draft during 1997 [12]. RC2 has many interesting and unique design features, particularly so when one considers the style of ciphers that dominated both the literature and the market at the time of its invention. The cipher was intended to be particularly efficient on 16-bit processors and with a 64-bit block size it was intended as a drop-in replacement for DES [11]. A significant feature of RC2 is the flexibility offered to the user in terms of the effective key-size. This has now become a common feature of many block cipher proposals and it is a property that has proven to be important in commercial applications. Over the years RC2 has been deployed widely and it features prominently in the S/MIME secure messaging standard [5]. Currently there are no published results on the cryptanalytic strength of RC2. As a first step this paper sets out some details on how the basic attacks of differential [1] and linear [8] cryptanalysis might apply.

2 The Design of RC2

There are two distinct parts to using RC2. First a *key expansion* procedure takes a user-supplied key of between one and 128 bytes in length together with a parameter that specifies the effective key-length of encryption. From this information an array $K[\cdot]$ of 64 16-bit round keys is

[5] RC2 is a registered trademark of RSA Data Security, Inc.

derived. Then a 64-bit plaintext block is encrypted using array $K[\cdot]$. Encryption consists of two styles of rounds. One is termed a `MIXING` round and the other a `MASHING` round.

Both the key expansion and encryption components rely on the use of a substitution table called `PITABLE`. This table specifies a random permutation on the integers $0, \ldots, 255$ and was derived from the expansion of $\pi = 3.14159\ldots$. The table itself will not concern us directly in this paper, but it is included for completeness in the Appendix. We will now describe the action of RC2 in more detail. We will use $x <\!<\!< k$ to denote the 16-bit word x rotated left by k bits, & will denote bitwise logical AND, $\oplus$ will denote bitwise exclusive-or and $\sim$ will denote bitwise complementation. All 16-bit word addition $+$ is performed modulo 2^{16}.

2.1 Key Expansion

During the key expansion procedure both byte operations and 16-bit word operations are used. The array $K[\cdot]$ that stores the 64 16-bit round keys will be referred to in two ways.

a) For word operations the positions of the buffer will be referred to as $K[0]$, ..., $K[63]$ where each $K[i]$ is a 16-bit word.
b) For byte operations the array of round keys will be referred to as $L[0]$, ..., $L[127]$ where each $L[i]$ is an eight-bit byte. It will always be the case that $K[i] = L[2i] + 256 \times L[2i+1]$ (That is, the lower order byte is given first).

Suppose that T bytes of key are supplied by the user with $1 \leq T \leq 128$. The key expansion procedure places the T-byte key into $L[0]$, ..., $L[T-1]$ of the key buffer. Regardless of the value of T however, the algorithm has a maximum effective key length in bits that is denoted $T1$. The effective key length in bytes $T8$ and a mask TM based on the effective key length in bits $T1$ are derived as $T8 = \lceil T1/8 \rceil$ and $TM = 255 \bmod 2^{8(1-T8)+T1}$. Key expansion consists of the following two loops and intermediate step:

1. `for` $i = T, T+1, ..., 127$ `do`
   $L[i] = \texttt{PITABLE}[L[i-1] + L[i-T]]$ (addition is modulo 256)
2. $L[128 - T8] = \texttt{PITABLE}[L[128 - T8] \,\&\, TM]$
3. `for` $i = 127 - T8, ..., 0$ `do`
   $L[i] = \texttt{PITABLE}[L[i+1] \oplus L[i + T8]]$

At the end of this key expansion the array $K[0]$, ..., $K[63]$ contains the 64 16-bit subkey words that will be used during encryption.

2.2 Encryption and Decryption

The encryption operation is defined in terms of primitive `MIX` and `MASH` operations. An array of four 16-bit words $R[0]$, ..., $R[3]$ are used to hold the initial plaintext, the intermediate results, and the final ciphertext. Indices to this array are always given modulo 4.

`MIX` $R[i]$

The primitive "`MIX` $R[i]$" operation is defined as follows, where $s[0] = 1$, $s[1] = 2$, $s[2] = 3$, and $s[3] = 5$. Here j is a "global" variable so that $K[j]$ is always the first key word in the expanded key which has not yet been used in a MIX operation.

$$R[i] = R[i] + K[j] + (R[i-1] \,\&\, R[i-2]) + ({\sim}R[i-1] \,\&\, R[i-3]);$$
$$j = j + 1;$$
$$R[i] = R[i] <\!<\!< s[i];$$

MIXING round
A MIXING round consists of MIX $R[0]$, MIX $R[1]$, MIX $R[2]$, MIX $R[3]$.

MASH $R[i]$
The primitive "MASH $R[i]$" operation is defined as follows:

$$R[i] = R[i] + K[\,R[i-1] \,\&\, \mathtt{003f}_x\,];$$

MASHING round
A MASHING round consists of MASH $R[0]$, MASH $R[1]$, MASH $R[2]$, MASH $R[3]$.

The entire encryption operation can now be described as follows. Here j is a global integer variable which is only affected by the mixing operations.

Encryption With RC2

1. Initialize words $R[0]$, ..., $R[3]$ to contain the 64-bit plaintext block.
2. Expand the key, so that words $K[0]$, ..., $K[63]$ become defined.
3. Initialize j to zero.
4. Perform five MIXING rounds.
5. Perform one MASHING round.
6. Perform six MIXING rounds.
7. Perform one MASHING round.
8. Perform five MIXING rounds.
9. The ciphertext is $R[0]$, ..., $R[3]$.

Decryption is the reverse of encryption. Since the details can easily be established they are not included here. Test vectors for encryption using RC2 are provided in the Appendix.

2.3 Features of RC2

RC2 is rather unusual in that the 64-bit plaintext block is split into four words each of 16 bits. In a style reminiscent of the hash function MD4 [13], much of the encryption process relies on one of these words being modified by a function of the other three, the four words then being swapped cyclically. This design approach was explored some seven years after the design of RC2, which now might be described as being an "unbalanced Feistel cipher" [4].

The key schedule for RC2 is also unusual. $T8$ is the number of bytes needed to contain the given $T1$ bits of key. When $T1$ is congruent to k, modulo 8, a mask TM containing ones in the low-order k bits is used to derive the correct effective key length. The first step of the key expansion expands the key to a full 128 bytes, using a non-linear byte-wide feedback shift-register approach. Step three is similar to the first, except that it starts at the high end and works towards the lower end. Steps two and three also work together to limit the effective key size to $T1$ bits. Step three corresponds to using a feedback register of only $T8$ bytes, and step two ensures that the initial state of that register has only $T1$ bits of entropy. Although the procedure limits the actually entropy of the key to $T1$ bits, it also ensures that the final key table depends upon each bit of the supplied key. If one supplies a 16-byte key, but set $T1 = 40$, then changing any bit of the supplied key should result in a different key table, although the number of possible key tables is limited to 2^{40}.

3 Differential Cryptanalysis of RC2

Differential cryptanalysis [1] can be a powerful style of attack. By choosing a pair of plaintexts with a particular difference, which can be adapted to the cipher in question, the cryptanalyst hopes that some identifiable, and unusual behavior, can be observed by processing the ciphertexts. One possible evolution of the difference between a pair of plaintexts during encryption can be described by a *characteristic*. In essence, a characteristic specifies the difference between two parallel encryptions at each stage of the encryption process and there is some associated probability that a pair being encrypted does indeed follow this description. A plaintext pair that follows the characteristic is typically called a *right pair*. A pair that does not is called a *wrong pair*.

Throughout our attack on RC2, we shall define the difference between two 16-bit words A and B to be $A \oplus B$. Furthermore in our analysis we shall be interested in how single-bit differences behave within RC2. The decision to restrict our attention to single-bit differences facilitates analysis but is also motivated by a typical assumption that characteristics involving multiple-bit differences over integer addition will generally hold with lower probability than single-bit characteristics [6]. We note that other more complex techniques [2, 7] might open new avenues for the analysis of RC2.

We will use e_t to denote the 16-bit word with a single one bit in position t from the right, all other bits being set to zero. We also view the leftmost bit of a 16-bit word to be the most significant bit. Thus we shall use e_{15} to denote a 16-bit word with the only non-zero bit being the most significant bit. We will denote the word of 16 zero bits as $\mathtt{0000_x}$ where the subscript $\mathtt{x}$ denotes hexadecimal notation and we will denote the *Hamming weight* (i.e. the number of ones in the binary expansion of some quantity x) as $\mathrm{Hwt}(x)$.

For the remainder of the paper, we shall consider `MIXING` and `MASHING` rounds in the following way. Instead of viewing the operation at each step as acting on a different word we shall consider the operations to be identical (i.e., at each `MIX` step $R[0] = R[0] + K[j] + (R[3] \,\&\, R[2]) + ({\sim}R[3] \,\&\, R[1])$) but that between steps the words are rotated cyclically (i.e., $\mathtt{TEMP} = R[0]$; $R[0] = R[1]$; $R[1] = R[2]$; $R[2] = R[3]$; $R[3] = \mathtt{TEMP}$).

3.1 Some Basic Characteristics for MIX

Given an input difference (e_t $\texttt{0000}_\texttt{x}$ $\texttt{0000}_\texttt{x}$ $\texttt{0000}_\texttt{x}$) to the first MIX step in a MIXING round, the output difference before rotation will be (e_t $\texttt{0000}_\texttt{x}$ $\texttt{0000}_\texttt{x}$ $\texttt{0000}_\texttt{x}$) with probability $p \geq 1/2$. Rotation then moves this single bit difference within the word, and the four words are swapped cyclically. We can summarize the four basic characteristics which hold with probability $p \geq 1/2$ (when averaged over all plaintexts and key words) for a MIX step. The value of the rotation $s[i]$ depends on the step i in which the characteristic is applied. Note that addition within the subscript of e_t is to be performed modulo 16.

$$(e_t\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}) \rightarrow (\texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ e_{t+s[i]}) \quad (1)$$

$$(\texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ e_t) \rightarrow (\texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ e_t\ \texttt{0000}_\texttt{x}) \quad (2)$$

$$(\texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ e_t\ \texttt{0000}_\texttt{x}) \rightarrow (\texttt{0000}_\texttt{x}\ e_t\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}) \quad (3)$$

$$(\texttt{0000}_\texttt{x}\ e_t\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}) \rightarrow (e_t\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}) \quad (4)$$

Apart from (1) with $t = 15$ which holds with probability $p = 1$, these characteristics hold with probability $p = 1/2$ on average. There are times where the characteristics do not hold. The following are the cases where the characteristic hold with certainty:

- In (2), if $(R[2]\ \&\ e_t) = (R[1]\ \&\ e_t)$ then $p = 1$.
- In (3), if $(R[3]\ \&\ e_t) = \texttt{0000}_\texttt{x}$ then $p = 1$.
- In (4), if $(R[3]\ \&\ e_t) = e_t$ then $p = 1$.

In the first MIXING round, the attacker chooses the plaintext and this allows the cryptanalyst to capture some of these special cases in an attack.

3.2 Some Basic Characteristics for MASH

There are two MASHING rounds in RC2 and the basic MASH step is $R[0] = R[0] + K[\,R[3]\ \&\ \texttt{003f}_\texttt{x}\,]$. Given an input difference ($\texttt{0000}_\texttt{x}$ $\texttt{0000}_\texttt{x}$ $\texttt{0000}_\texttt{x}$ e_t) to a MASHING round with $(e_t\ \&\ \texttt{003f}_\texttt{x}) = \texttt{0000}_\texttt{x}$ the same key word $K[\cdot]$ will be added to both sets of partially encrypted data. The four basic useful characteristics for MASH are as follows:

$$(e_t\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}) \rightarrow (\texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ e_t) \quad (5)$$

$$(\texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ e_t) \rightarrow (\texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ e_t\ \texttt{0000}_\texttt{x}) \quad (6)$$

$$(\texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ e_t\ \texttt{0000}_\texttt{x}) \rightarrow (\texttt{0000}_\texttt{x}\ e_t\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}) \quad (7)$$

$$(\texttt{0000}_\texttt{x}\ e_t\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}) \rightarrow (e_t\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}\ \texttt{0000}_\texttt{x}) \quad (8)$$

Characteristic (5) holds with probability $p = 1/2$ unless $t = 15$ when it holds with probability $p = 1$, characteristics (7) and (8) hold with probability $p = 1$, and characteristic (6) holds with probability $p = 1$ if $(e_t\ \&\ \texttt{0x3f}) = \texttt{0000}_\texttt{x}$. Joining these four characteristics together to pass across a MASHING round with probability $p = 1$ is straightforward.

3.3 Towards a Differential Attack on RC2

In this section we combine characteristics for both MIXING and MASHING rounds while moving towards a full analysis of RC2. We will assume that the subkey words $K[0], \ldots, K[63]$ are independent and we aim to recover the expanded key table $K[\cdot]$ in our attack.

The characteristics of interest are built around single-bit differences and as noted in Section 3.1 there are advantages to having this single non-zero bit in the most significant bit of a word. Depending on which word $R[\cdot]$ is the subject of the characteristic we use, different rotation amounts feature during MIXING. This leads to conditions on t, the position of the single-bit difference in the plaintext, that provide some advantages in an attack. Another consideration is the presence of the MASHING rounds and one aim might be to nullify their action. If a one-bit characteristic specifies an input difference to a MASHING round of e_t in any one of the words, then provided $t = 15$ the characteristic will pass through the MASHING round unhindered with probability $p = 1$. If $5 < t < 15$ then there is a characteristic that holds with probability $p = 1/2$. There are six MIXING rounds between the two MASHING rounds and so with the difference (e_t $0000_{\mathtt{x}}$ $0000_{\mathtt{x}}$ $0000_{\mathtt{x}}$) as input to the first MASHING round we can establish the values of t that are useful to us.

A more accurate reflection of the success of a final attack is given by considering *differentials* [10] instead of characteristics (which provide only a lower bound to the probability of the differential). In Section 3.5 we will consider the issue of differentials in more detail but from this point on we will anticipate later analysis by referring to the use of differentials during our description of the attack. The observations provided so far allow us to present in Table 1 the differentials that are useful to us.

3.4 Recovering Key Information

In a differential cryptanalytic attack the attacker typically chooses a differential for $(n-1)$ rounds of an n-round block cipher. The attacker then tries to deduce key information from the last round of the cipher [1]. Here, the most effective attack on RC2 appears to require that bits of the subkey $K[0]$ used in the first MIXING round are recovered first.

Consider a differential with input difference ($0000_{\mathtt{x}}$ $0000_{\mathtt{x}}$ $0000_{\mathtt{x}}$ e_t). The starting values of $R[1]$ and $R[2]$ are chosen so that $(R[1] \,\&\, e_t) = (R[2] \,\&\, e_t)$. After the first MIX step the difference will be ($0000_{\mathtt{x}}$ $0000_{\mathtt{x}}$ e_t $0000_{\mathtt{x}}$). The output difference from the second MIX step will depend on the value of bit t in register $R[3]$. If this bit is zero then word $R[1]$ with difference $0000_{\mathtt{x}}$ will be chosen. Otherwise word $R[2]$ with difference e_t will be selected and a difference will be introduced into another word. Note that the value of this bit depends on the plaintext (which we know) and on bits of the first 16-bit subkey word $K[0]$.

We can trace the output of the second MIX step to the end of the penultimate MIXING round by using the differentials in Table 1. If the pair is a right pair then we can recover one bit of information from $K[0]$ as follows. A necessary condition for a pair to be a good pair is that

$$e_t \;\&\; ((R[0] + (R[3] \,\&\, R[2]) \tag{9}$$

plaintext difference	*difference at start of last* `MIXING` *round*	*prob.*	*values of t*
$(e_t\ \mathtt{0000_x}\ \mathtt{0000_x}\ \mathtt{0000_x})$	$(e_{t+15}\ \mathtt{0000_x}\ \mathtt{0000_x}\ \mathtt{0000_x})$	2^{-58}	4
$(e_t\ \mathtt{0000_x}\ \mathtt{0000_x}\ \mathtt{0000_x})$	$(e_{t+15}\ \mathtt{0000_x}\ \mathtt{0000_x}\ \mathtt{0000_x})$	2^{-59}	1, 2, 3
$(\mathtt{0000_x}\ e_t\ \mathtt{0000_x}\ \mathtt{0000_x})$	$(\mathtt{0000_x}\ e_{t+14}\ \mathtt{0000_x}\ \mathtt{0000_x})$	2^{-58}	5
$(\mathtt{0000_x}\ e_t\ \mathtt{0000_x}\ \mathtt{0000_x})$	$(\mathtt{0000_x}\ e_{t+14}\ \mathtt{0000_x}\ \mathtt{0000_x})$	2^{-59}	1, 3
$(\mathtt{0000_x}\ e_t\ \mathtt{0000_x}\ \mathtt{0000_x})$	$(\mathtt{0000_x}\ e_{t+14}\ \mathtt{0000_x}\ \mathtt{0000_x})$	2^{-60}	0, 2, 4
$(\mathtt{0000_x}\ \mathtt{0000_x}\ e_t\ \mathtt{0000_x})$	$(\mathtt{0000_x}\ \mathtt{0000_x}\ e_{t+13}\ \mathtt{0000_x})$	2^{-58}	14
$(\mathtt{0000_x}\ \mathtt{0000_x}\ e_t\ \mathtt{0000_x})$	$(\mathtt{0000_x}\ \mathtt{0000_x}\ e_{t+13}\ \mathtt{0000_x})$	2^{-59}	7, ..., 13
$(\mathtt{0000_x}\ \mathtt{0000_x}\ \mathtt{0000_x}\ e_t)$	$(\mathtt{0000_x}\ \mathtt{0000_x}\ \mathtt{0000_x}\ e_{t+11})$	2^{-58}	6
$(\mathtt{0000_x}\ \mathtt{0000_x}\ \mathtt{0000_x}\ e_t)$	$(\mathtt{0000_x}\ \mathtt{0000_x}\ \mathtt{0000_x}\ e_{t+11})$	2^{-59}	15, 0, ..., 5

Table 1. 26 differentials that are potentially useful in an attack on RC2. The associated probabilities are lower bounds provided by the analysis of a characteristic contained within the specified differential.

$$+ (\tilde{}R[3]\,\&\,R[1]) + K[0]) \lll 1) = 0.$$

Let $x = R[0] + (R[3]\,\&\,R[2]) + (\tilde{}R[3]\,\&\,R[1])$ which we control via the choice of plaintext. Then we have the following condition for a right pair:

$$(x + K[0])\,\&\,e_{t-1} = 0. \tag{10}$$

Denote by k the value derived by setting the top $((16 - t) \bmod 16)$ bits of $K[0]$ to zero. Let $y = x\,\&\,e_{t-1}$ and let z be the quantity derived by setting the top $((16-t+1) \bmod 16)$ bits of x to zero. Then we have that

$$(x + K[0])\,\&\,e_{t-1} = 0 \Leftrightarrow y = (z + k)\,\&\,e_{t-1}. \tag{11}$$

To mount an attack to recover bit $(t-1)$ of k for some given t we encrypt plaintext pairs with $z = 0$ until we obtain a right pair. Once we have a right pair we observe the value of y. From this we deduce the value of bit $(t-1)$ in k and hence in $K[0]$. We can then repeat this approach choosing pairs with different values to z so that information on the subkey $K[0]$ is recovered bit by bit.

By using different differentials with different values of t (see Table 1) we are able to introduce some error-checking into the attack[6]. In this way the bits of $K[0]$ that we recover can be verified. All recovered bits of $K[0]$ have to be correct before the next bit of $K[0]$ can be correctly derived. Note that *structures* [1] can be useful in reducing the plaintext requirements for a differential attack when more than one differential is useful. With n useful differentials we can ask for a *structure* of 2^n

[6] Not all values of t are valid for use due to the two `MASHING` rounds.

plaintexts with specifically chosen differences. From these we derive 2^{n-1} plaintext pairs for each of the n characteristics.

There remains the issue of detecting when a data pair is a good pair. We note that the difference at the start of the final `MIXING` round has Hamming weight one for a good pair. We might therefore measure the Hamming weight of the ciphertext and if the weight is less than some threshold the pair can be considered a right pair. Depending on the threshold we might accept some wrong pairs as being right pairs, something that would provide a wrong answer to the bit we wish to recover with probability 1/2. To improve the robustness of the attack one might aim to collect more right pairs. Then the value of the bit suggested most often can be assumed to be the correct value to the key bit we are trying to recover. As a demonstration we provide the success rate for different amounts of plaintext in experiments on eight-round RC2. (There are eight `MIXING` rounds with a `MASHING` inserted after round five as occurs in RC2.) A decision on whether a good pair had occurred was made according to whether the Hamming weight of the difference in the ciphertext was less than some threshold. Then, once a value for the key bit had been counted more than the other (this difference being denoted by *excess*) that value for the key bit was set. Each entry in the table was computed after 20 experiments.

	Hamming weight					
excess	11		12		13	
2	90%	2^{29}	20%	$2^{28.5}$	0%	$2^{27.5}$
4	100%	2^{30}	95%	2^{30}	20%	$2^{29.5}$
8	100%	2^{31}	100%	2^{31}	65%	2^{31}

3.5 The Effectiveness of Differential Cryptanalysis

As we previously mentioned it is differentials and their probabilities that reflect the effectiveness of a differential attack. Whereas a characteristic describes one specific evolution of differences through encryption, from a given starting difference there might well have been other "paths" through the cipher to the same target difference than the one described by one particular characteristic. With RC2 this leads to a particularly interesting interaction between the `MIXING` and `MASHING` rounds.

First we will consider in abstract terms the probability that a one-bit difference in some word a produces a one-bit difference in the word d when we define $d = a+b+c$ for unknown constants b and c. One approach might be to consider this as two separate additions and to consider the intermediate word $e = a+b$ first. Since a one-bit difference in a produces a one-bit difference in e with probability 1/2 and a one-bit difference in e provides a one-bit difference in $d = e + c$ with probability 1/2 we would say that the characteristic over the two additions has probability 1/4. However it would then be misleading to use this characteristic to provide an approximation to the probability of the differential from a to d. Instead, the probability of the propagation of a one-bit difference from

a to d is $1/2$ since $b+c$ is a fixed value. Consequently the probability of the differential from a to d must also be $1/2$.

Recall that the probability of the differential is given by the sum of the probabilities of all the characteristics that satisfy the differential. By looking at two successive additions in isolation we inadvertently restrict our attention to single-bit differences in the intermediate value e. Let α, $0 \leq \alpha \leq n-1$, denote the position of the one bit difference in a. A one-bit difference in a will give a difference in e with Hamming weight h with probability 2^{-h}, $1 \leq h < n-\alpha$, and with probability $2^{-n+\alpha+1}$ for $h = n-\alpha$. Since this h-bit difference was caused by a one-bit difference in the previous step[7] an h-bit difference in e will be transformed to a one-bit difference in d by the addition of c with probability $1/2$. Thus we get

$$p = 2^{-1}(2^{-n+\alpha} + \sum_{h=1}^{n-\alpha} 2^{-h}), \quad \text{if } \alpha < n-1 \tag{12}$$

$$p = 1 \text{ if } \alpha = n-1. \tag{13}$$

One place where this has an effect is when a `MIXING` round follows a `MASHING` round. Each word $R[0]$, ..., $R[3]$ is modified by a `MASH` step in turn. At the first subsequent `MIX` step $R[0]$ is modified by means of addition. By looking at the two additions in isolation one under-estimates the probability of the differential.

In the analysis of RC2 we need to take account of this effect since it applies to some extent to the `MIXING` rounds as well as during the transition between `MIXING` and `MASHING` rounds. Within the `MIXING` rounds an intermediate quantity is used as input to a multiplexor function. This reduces the probability that this particular characteristic is followed by a factor of 2^{-h} for each multiplexor when the Hamming weight of the difference is h. If we denote the number of multiplexing functions between two successive additions by k then (12) can be rewritten as follows:

$$p = \sum_{h=1}^{n-\alpha} 2^{-h} \cdot 2^{-hk} \cdot 2^{-1} + 2^{-(n-\alpha)} \cdot 2^{-(n-\alpha)k} \cdot 2^{-1} \tag{14}$$

$$= 2^{-1} \left(\sum_{h=1}^{n-\alpha} 2^{-(k+1)h} + 2^{-(k+1)(n-\alpha)} \right) \tag{15}$$

$$= 2^{-1} \left(\frac{1 - 2^{-(k+1)(n-\alpha)}}{1 - 2^{-(k+1)}} \frac{1}{2^{(k+1)}} \right) + 2^{-(k+1)(n-\alpha)} \tag{16}$$

$$\approx 2^{-1} \left(\frac{1}{2^{(k+1)} - 1} \right). \tag{17}$$

The last approximation is reasonable for smaller α ($\alpha < n-3$) but would need some correction for larger values of α. For $k = 0,1,2,3$, (17) gives $p = 1/2, 1/6, 1/14, 1/30$, which should be compared with the respective probabilities of the characteristics we previously derived:

[7] In general it is not true that an h-bit difference goes to a one-bit difference with such a high probability.

$1/4, 1/8, 1/16, 1/32$. In the case of two consecutive `MIXING` rounds we have that $k = 3$ and so the probability of a one-bit to one-bit differential across two `MIXING` rounds is $1/30 \times 2^{-3} = 1/240$.

The effect we are using here can be extended to a series of additions whereby the intermediate values of interest have differences with a variety of Hamming weights even though the starting and ending difference have weight one. Consider three consecutive mixing rounds. Let a be a one-bit difference in the leftmost words of two inputs and let α be the position of that bit, where $0 \leq \alpha \leq n-1$. Let d be the difference in the leftmost words after three mixing rounds and suppose that h_1 and h_2 denote the Hamming weights of the leftmost words after one, respectively two, mixing rounds. Then the probability that d is a one-bit difference can be estimated as follows, where $k = 3$ and where for simplicity we have eliminated the term for $h = n - \alpha$.

$$p \simeq \sum_{h_1=1}^{n-\alpha} \sum_{h_2=1}^{n-\alpha} 2^{-h_1} \cdot 2^{-h_1 k} \cdot 2^{-h_2} \cdot 2^{-h_2 k} \cdot 2^{-1} \tag{18}$$

$$= 2^{-1} \left(\sum_{h_1=1}^{n-\alpha} 2^{-(k+1)h_1} \cdot \sum_{h_2=1}^{n-\alpha} 2^{-(k+1)h_2} \right) \tag{19}$$

$$\approx 2^{-1} \left(\frac{1}{2^{(k+1)} - 1} \right) \left(\frac{1}{2^{(k+1)} - 1} \right). \tag{20}$$

Again the final approximation requires that α is small. For $k = 3$ p is $2^{-1}(1/15)^2$. We can now estimate the probability of the differential over three mixing rounds by $2^{-1}(1/15)^2 \times 1/8 \simeq 1/3600$. This extends easily to more rounds and in general the probability of a differential over r mixing rounds is $(1/15)^{r-1} \times 1/16$. Note that the `MASHING` rounds can be passed with probability one.

For a more accurate assessment a slight correction should be applied for rounds where the difference is close to the most significant bit, but experimental evidence given below suggests that the expressions derived are reasonable to use. The number of rounds in the table refers to the number of `MIXING` rounds used. After five `MIXING` rounds an additional `MASHING` round is inserted as occurs when encrypting with RC2. The final column is derived as an average over at least five sets of experiments for each row.

# rounds	*# pairs/test*	*# right pairs expected*	*# right pairs obtained*
3	2^{19}	146	146
4	2^{22}	78	79
6	2^{29}	44	47
7	2^{31}	12	13

Note that the probability of the differential obtained in this section does not take into account text pairs which have internal differences in more than one word before they resynchronize. This was observed occasionally during experiments but cases where differences in more than one word resynchronize are rare and we ignore their impact on our estimates.

3.6 Differential Cryptanalysis of RC2

We arrive at the following estimates for the data required to recover information about the subkey $K[0]$. Once this subkey word has been recovered the attack is repeated on what would now become a reduced version of RC2. When we take into account the key-recovery techniques of Section 3.4 we estimate that a differential attack on RC2 with r `MIXING` rounds (including the `MASHING` rounds) requires at most 2^{4r} chosen plaintexts. An attack on RC2 with 16 `MIXING` rounds requires use of a differential with probability at least $2^{-58.7}$ ("at least" since we have not yet accounted for such phenomena as a one-bit difference in the most significant bit at a `MIXING` round). In this regard RC2 with 16 `MIXING` rounds compares favorably to DES (2^{47} pairs [1]) and 12-round RC5 (2^{44} pairs [3]). It is fair to observe, however, that RC2 is not a fast cipher and an optimized version of DES and 12-round RC5 are both likely to be faster than RC2.

4 Linear Cryptanalysis of RC2

Linear cryptanalysis has provided the best theoretical attack on DES in terms of data requirements [9]. However, its usefulness on other ciphers is often limited. The aim of such an attack is to relate bits of the plaintext and ciphertext to bits of the key via a linear equation which holds with some probability p. Such an approximation can generally be used to provide an estimate for one bit of the key and more advanced techniques are available to extract more key information [9]. If an approximation holds with probability p then the important quantity for the cryptanalyst is the absolute value of the bias of the approximation $b = |p - 1/2|$. Typically the data required to use such an approximation is given by $c \times b^{-2}$ known plaintexts for some small constant c [9].
The `MIX` step in RC2 is $R[0] = R[0]+K[j]+(R[3] \,\&\, R[2])+(\tilde{}R[3] \,\&\, R[1])$. Across integer addition the best linear approximation involves the least significant bit of each quantity, and will hold with probability one. The multiplexor function $x = (R[3] \,\&\, R[2]) + (\tilde{}R[3] \,\&\, R[1])$ has linear approximations of varying usefulness. The absolute value of the highest non-trivial bias is 1/4 when averaged over all plaintexts. In a slight abuse of notation we will consider a 16-bit word x as a vector in Z_2^{16} and we will use the 16-bit quantity α to indicate the bits of x that are to be used in a linear approximation. This is most conveniently described by means of the scalar product of two vectors. Thus the $\{0,1\}$-vector α will be used to denote the specific bits of x to be used in an approximation and $\alpha \cdot x$ is the value of these bits combined using exclusive-or. Useful linear approximations across the multiplexor are of the form

$$\alpha \cdot x = \alpha \cdot R[1] \qquad \alpha \cdot x = \alpha \cdot R[1] \oplus \alpha \cdot R[3]$$
$$\alpha \cdot x = \alpha \cdot R[2] \qquad \alpha \cdot x = \alpha \cdot R[2] \oplus \alpha \cdot R[3]$$

where $\mathrm{Hwt}(\alpha) = 1$. More generally approximations to the multiplexor function with non-zero bias have the form

$$\delta \cdot x = \alpha \cdot R[1] \oplus \beta \cdot R[2] \oplus \gamma \cdot R[3] \qquad (21)$$

where δ is the bitwise inclusive-or of α and β and γ is either 0 or it consists of ones in positions where either α or β have ones. The greater the value of $\mathrm{Hwt}(\gamma)$ the lower the absolute value of the bias of the approximation. The following approximation to the first `MIX` step (which includes the cyclic swap of the $R[\cdot]$ words) might be useful

$$e_1 \cdot (R[3]^{\mathrm{new}}) = e_0 \cdot (R[0]^{\mathrm{old}}) \oplus e_0 \cdot (K[j]) \oplus e_0 \cdot R[2]^{\mathrm{old}}.$$

This has a bias of absolute value 1/4. The following steps require no approximation and there appears to be no better non-trivial linear approximations for a complete `MIXING` round. We might illustrate this approximation in the following way:

step	$R[0]$	$R[1]$	$R[2]$	$R[3]$	*round 1*
	e_0	–	e_0	–	*start*
1	–	–	–	e_1	$\lvert b\rvert = 1/4$
2	–	–	e_1	–	$\lvert b\rvert = 1/2$
3	–	e_1	–	–	$\lvert b\rvert = 1/2$
4	e_1	–	–	–	$\lvert b\rvert = 1/2$

In continuing this approximation into the next `MIXING` round we would be forced to approximate the bit $e_1\&R[0]$. One integer addition involves the subkey word $K[4]$ and depending on this value the bias of the approximation will vary[8]. The second integer addition involves the output from the multiplexor function. By the conditions given above this approximation must involve $e_1\&R[1]$ or $e_1\&R[2]$ and we can construct the following approximations for the second and third `MIXING` rounds. Here we assume that the bias of the approximation across the multiplexor function is at most 1/4. Similarly, we assume that the bias of the approximation across the integer addition is at most 1/4. This occurs in approximating steps 1 and 3 and the value of $|b|$ is given for those steps individually.

step	$R[0]$	$R[1]$	$R[2]$	$R[3]$	*round 2*
	e_1	–	–	–	*start*
1	–	e_1	–	e_2	$\lvert b\rvert \leq 1/8$
2	e_1	–	e_2	–	$\lvert b\rvert = 1/2$
3	–	$e_1 \oplus e_2$	–	e_4	$\lvert b\rvert \leq 1/8$
4	$e_1 \oplus e_2$	–	e_4	–	$\lvert b\rvert = 1/2$

4.1 The Effectiveness of Linear Cryptanalysis

The typical way to measure the effectiveness of linear cryptanalysis is to appeal to the so-called *piling-up lemma* [8]. By doing this, we are lead to estimate a bias of $\leq 2^{-2} \times 2^{-3} \times 2^{-3} \times 2^{2} = 2^{-6}$ for our approximation to the first two `MIXING` rounds of RC2. In the case of RC2, however, routine use of the piling-up lemma can lead to misleading results.
As an example, suppose that the two subkeys used in steps one and three of round two are zero. In isolation the approximation to step one (A_1, say) holds with probability 5/8. In step three we find that the

[8] Note that the whole issue of key-dependence in linear cryptanalysis is a complex one that is rarely addressed in detail.

second approximation (A_2, say) involves bits that previously determined whether A_1 held. Analysis shows that the probability that A_2 holds given that A_1 held is 13/20 and not 5/8 when A_2 is considered in isolation. Furthermore, the probability that A_2 doesn't hold when A_1 doesn't hold is 5/12 instead of 3/8. So when the two approximations are combined the probability that the combined approximation to round two holds is $(5/8 \times 13/20) + (3/8 \times 5/12) = 9/16$ which leads to a bias of 1/16. This is greater than the 1/32 predicted by use of the piling-up lemma.

Much of the complicated interaction between the two approximations is due to the role of addition in the cipher. As an example, if we suppose that approximation A_1 holds, then it can be shown that the probability that the least significant bit of $R[2]$ is equal to zero is 11/20. Since this bit plays a pivotal role in determining whether A_2 holds it is no surprise that the piling-up lemma gives misleading results.

For the user of RC2 there is circumstantial evidence that linear cryptanalysis is unlikely to pose a threat to RC2. Such attacks appear to be ineffective for ciphers that mix integer addition and bitwise operations unless the approximation can be limited to the least significant bits across an addition [6]. Such a restriction appears unlikely as an extension of the current approximation into a third `MIXING` round illustrates:

step	$R[0]$	$R[1]$	$R[2]$	$R[3]$	*round 3*
	$e_1 \oplus e_2$	$-$	e_4	$-$	*start*
1	$-$	$e_1 \oplus e_2 \oplus e_4$	$-$	$e_2 \oplus e_3$	$\lvert b\rvert \leq 1/16$
2	$e_1 \oplus e_2 \oplus e_4$	$-$	$e_2 \oplus e_3$	$-$	$\lvert b\rvert = 1/2$
3	$-$	$e_1 \oplus e_3 \oplus e_4$	$-$	$e_4 \oplus e_5 \oplus e_7$	$\lvert b\rvert \leq 1/128$
4	$e_1 \oplus e_3 \oplus e_4$	$-$	$e_4 \oplus e_5 \oplus e_7$	$-$	$\lvert b\rvert = 1/2$

Nevertheless, there are complex interactions between the individual steps of RC2 and these often provide unintuitive results. In particular we have discovered cases where adding a non-trivial approximation to an existing approximation actually boosts the absolute value of the bias. (Such an example can be found in step 3 above when the subkeys in all rounds are set to zero.) Under such circumstances the true effectiveness of linear cryptanalysis in attacking RC2 has to remain an open problem.

5 Conclusions

In this paper we have described the block cipher RC2. While the cipher is perhaps slower than other alternatives available today, it does appear to offer effective resistance to differential cryptanalysis. Our attempts to apply linear cryptanalysis to RC2 have provided some intriguing insights, but are as yet insufficient to determine the actual resistance of RC2 to linear cryptanalysis; this remains an open problem. It is important that RC2 continues to come under close scrutiny from the cryptanalytic community.

References

1. E. Biham and A. Shamir. *Differential Cryptanalysis of the Data Encryption Standard.* Springer-Verlag, New York, 1993.
2. J. Borst, L.R. Knudsen, and V. Rijmen. Two attacks on reduced IDEA. In W. Fumy, editor, *Advances in Cryptology — Eurocrypt '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 1–13, 1997. Springer Verlag.
3. A. Biryukov and E. Kushilevitz. Improved cryptanalysis of RC5. Preprint.
4. M. Blaze and B. Schneier. The MacGuffin block cipher algorithm. In B. Preneel, editor, *Fast Software Encryption*, volume 1008 of *Lecture Notes in Computer Science*, pages 97–110, 1995. Springer Verlag.
5. S. Dusse, P. Hoffman, B. Ramsdell, L. Lundblade, and L. Repka. S/MIME Message Specification. September 23, 1997. Available from http://www.imc.org/draft-dusse-smime-msg.
6. 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*, volume 963 of *Lecture Notes in Computer Science*, pages 171–184, 1995. Springer Verlag.
7. L.R. Knudsen and W. Meier. Improved differential attacks on RC5. In N. Koblitz, editor, *Advances in Cryptology — Crypto '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 216–228, 1996. Springer Verlag.
8. M. Matsui. Linear cryptanalysis method for DES cipher. In T. Helleseth, editor, *Advances in Cryptology — Eurocrypt '93*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397, 1994. Springer-Verlag.
9. M. Matsui. The first experimental cryptanalysis of the Data Encryption Standard. In Y. Desmedt, editor, *Advances in Cryptology — Crypto '94*, volume 839 of *Lecture Notes in Computer Science*, pages 1–11, 1994. Springer-Verlag.
10. X. Lai, J. Massey and S. Murphy. Markov ciphers and differential cryptanalysis. In D. Davies, editor, *Advances in Cryptology — Eurocrypt '91*, volume 547 of *Lecture Notes in Computer Science*, pages 17–38, 1991. Springer Verlag.
11. National Institute of Standards and Technology (NIST). *FIPS Publication 46-2: Data Encryption Standard*. December 30, 1993.
12. R.L. Rivest. A Description of the RC2^{TM} Encryption Algorithm. File `draft-rivest-rc2desc-00.txt` available from `ftp://ftp.ietf.org/internet-drafts/`.
13. R.L. Rivest. The MD4 message digest algorithm. In A.J. Menezes and S.A. Vanstone, editors, *Advances in Cryptology — Crypto '90*, volume 537 of *Lecture Notes in Computer Science*, pages 303–311, 1991. Springer-Verlag.
14. R.L. Rivest. The RC5 encryption algorithm. In B. Preneel, editor, *Fast Software Encryption*, volume 1008 of *Lecture Notes in Computer Science*, pages 86–96, 1995. Springer Verlag.

Appendix

The substitution table `PITABLE` specified in hexadecimal notation for input byte `ab`.

a*	*b 0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00:	d9	78	f9	c4	19	dd	b5	ed	28	e9	fd	79	4a	a0	d8	9d
10:	c6	7e	37	83	2b	76	53	8e	62	4c	64	88	44	8b	fb	a2
20:	17	9a	59	f5	87	b3	4f	13	61	45	6d	8d	09	81	7d	32
30:	bd	8f	40	eb	86	b7	7b	0b	f0	95	21	22	5c	6b	4e	82
40:	54	d6	65	93	ce	60	b2	1c	73	56	c0	14	a7	8c	f1	dc
50:	12	75	ca	1f	3b	be	e4	d1	42	3d	d4	30	a3	3c	b6	26
60:	6f	bf	0e	da	46	69	07	57	27	f2	1d	9b	bc	94	43	03
70:	f8	11	c7	f6	90	ef	3e	e7	06	c3	d5	2f	c8	66	1e	d7
80:	08	e8	ea	de	80	52	ee	f7	84	aa	72	ac	35	4d	6a	2a
90:	96	1a	d2	71	5a	15	49	74	4b	9f	d0	5e	04	18	a4	ec
a0:	c2	e0	41	6e	0f	51	cb	cc	24	91	af	50	a1	f4	70	39
b0:	99	7c	3a	85	23	b8	b4	7a	fc	02	36	5b	25	55	97	31
c0:	2d	5d	fa	98	e3	8a	92	ae	05	df	29	10	67	6c	ba	c9
d0:	d3	00	e6	cf	e1	9e	a8	2c	63	16	01	3f	58	e2	89	a9
e0:	0d	38	34	1b	ab	33	ff	b0	bb	48	0c	5f	b9	b1	cd	2e
f0:	c5	f3	db	47	e5	a5	9c	77	0a	a6	20	68	fe	7f	c1	ad

Test vectors for the RC2 encryption algorithm.

key length (bytes)	*effective key length (bits)*	*key*	*plaintext*	*ciphertext*
8	63	00000000 00000000	00000000 00000000	ebb773f9 93278eff
8	64	ffffffff ffffffff	ffffffff ffffffff	278b27e4 2e2f0d49
8	64	30000000 00000000	10000000 00000001	30649edf 9be7d2c2
1	64	88	00000000 00000000	61a8a244 adacccf0
7	64	88bca90e 90875a	00000000 00000000	6ccf4308 974c267f
16	64	88bca90e 90875a7f 0f79c384 627bafb2	00000000 00000000	1a807d27 2bbe5db1
16	128	88bca90e 90875a7f 0f79c384 627bafb2	00000000 00000000	2269552a b0f85ca6
33	129	88bca90e 90875a7f 0f79c384 627bafb2 16f80a6f 85920584 c42fceb0 be255daf 1e	00000000 00000000	5b78d3a4 3dfff1f1

This article was processed using the LaTeX macro package with LLNCS style