Cryptanalysis of SPEED

Chris Hall[1], John Kelsey[1], Vincent Rijmen[2], Bruce Schneier[1], and David Wagner[3]

[1] Counterpane Systems
101 E. Minnehaha Pkwy
Minneapolis, MN 55419
(612) 823-1098
{hall,kelsey,schneier}@counterpane.com
[2] K.U.Leuven, Dept. ESAT, SISTA/COSIC Lab
K. Mercierlaan 94, B-3001 Heverlee
Belgium
vincent.rijmen@esat.kuleuven.ac.be
[3] U.C. at Berkeley
Soda Hall
Berkeley, CA 94720-1776
daw@cs.berkeley.edu

Abstract. The cipher family SPEED (and an associated hashing mode) was recently proposed in *Financial Cryptography '97*. This paper cryptanalyzes that proposal, in two parts: First, we discuss several troubling potential weaknesses in the cipher. Next, we show how to efficiently break the SPEED hashing mode using differential related-key techniques, and propose a differential attack on 48-round SPEED. These results raise some significant questions about the security of the SPEED design.

1 Introduction

In *Financial Cryptography '97*, Zheng proposed a new family of block ciphers, called SPEED [12]. One specifies a particular SPEED cipher by choosing parameters such as the block size and number of rounds; the variations are otherwise alike in their key schedule and round structure. Under the hood, SPEED is built out of an unbalanced Feistel network. Zheng also proposed a hash function based on running a SPEED block cipher in a slightly modified Davies-Meyer mode.

One of the main contributions of the SPEED design is its prominent use of carefully chosen Boolean functions which can be shown to have very good non-linearity, as well as other desirable theoretical properties. One might therefore hope that SPEED rests on a solid theoretical foundation in cryptographic Boolean function theory. Nonetheless, this paper describes serious weaknesses in the cipher; many lead to practical attacks on SPEED.

This paper is organized as follows. Section 2 briefly summarizes the SPEED design. In Section 3, we discuss some preliminary analysis of the SPEED design, including comments on differential characteristics in SPEED, and on the non-surjectivity of the SPEED round function. Then we shift emphasis: Section 4 discusses differential characteristics for SPEED with 48 rounds. There appears to be an obvious 1-bit characteristic with probability 2^{-50} after 40 rounds; however, this characteristic does not actually work. We discuss this and other failed characteristics in Section 4. In Section 5 we describe how a differential attack can be mounted despite these problems. Section 6 gives differential related-key attacks on block cipher and shows how to apply them to efficiently find collisions in the SPEED hash function. Section 7 assesses the practical implications of these attacks, proposing a rule of thumb to characterize when we can consider a parametrized cipher family "broken." Finally, we conclude this paper in Section 8.

2 Background

SPEED is a parameterized unbalanced Feistel cipher with a variable block width w, variable key length l, and a variable number of rounds R (R must be a multiple of 4 and w a multiple of

8). The block is split up into eight equally-sized pieces: $B_7, \dots, B_0$. The round function is then characterised by

$$t(B_7, \dots, B_1, B_0) = (B_6, \dots, B_1, B_0, r(B_7) \boxplus K_i \boxplus v(F_i(B_6, \dots, B_1, B_0)))$$

where $\boxplus$ denotes addition modulo $2^{w/8}$, F_i is a round-dependent function with a $w/8$-bit output, v is a data-dependent rotation, K_i is a round-dependent subkey, and r is a bijective function from $\{0,1\}^{w/8}$ to $\{0,1\}^{w/8}$ (r is always right rotate by $w/16 - 1$ by bits). See Figure 1 for one round of SPEED.

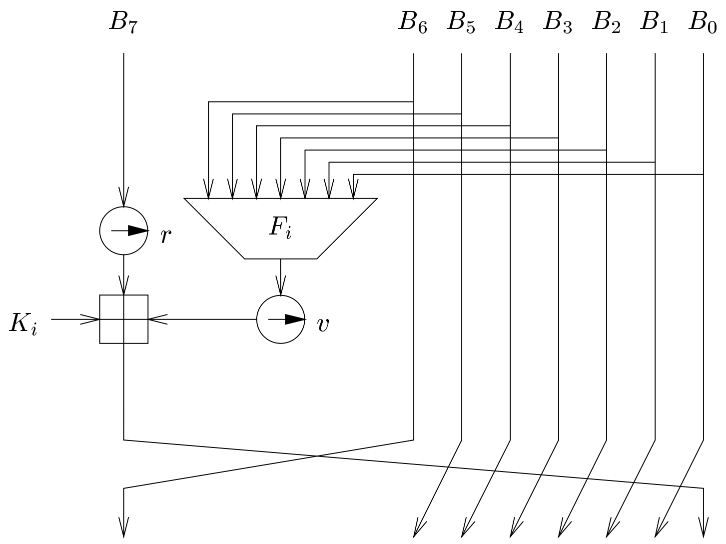

Fig. 1. One round of SPEED.

We sometimes refer to the seven input pieces, $B_6, \dots, B_0$, as the source block and the modified piece, B_7, the target block. The number of rounds is a parameter; the paper suggests using at least 48 rounds for adequate security for the block cipher, and at least 32 rounds for the hash function. We assume that the underlying CPU operates on $w/8$-bit words.

The round dependent function F_i takes seven $w/8$-bit inputs and produces one $w/8$-bit output. The function v takes a $w/8$-bit input and produces a $w/8$-bit output by rotating the input a number of bits dependent on the input. The function that results from combining F_i and v will be denoted h_i.

$$h_i(x_6, \dots, x_0) = v(F_i(x_6, \dots, x_0))$$

The exact details of the rotation in v are not important to our analysis and may be found in [12] In this paper, we will write $a^{\gg b}$ to stand for the result of rotating a right by b bits.

There are four different F functions F_i for $i \in \{1,2,3,4\}$, each is used for a different set of rounds. Each F_i is built out of a single (1-bit) Boolean function f_i on 7 (1-bit) variables, which is extended to a $w/8$-bit function by bitwise parallel evaluation. In other words, bit position i of the output depends only on seven input bits, each at position i in one of the input words. For example, F_1 is $F_1(x_6, \dots, x_0) = x_6x_3 \oplus x_5x_1 \oplus x_4x_2 \oplus x_1x_0 \oplus x_0$ where x_ix_j denotes bitwise AND. SPEED uses F_1 for the first $R/4$ rounds, then F_2 for the next $R/4$ rounds, and so on.

In summary, each round of SPEED uses one evaluation of F, two rotations, and an addition modulo $2^{w/8}$ to update the block.

3 Preliminary Analysis

In this section, we begin analysis of SPEED with respect to potential cryptanalytic attacks.

3.1 Non-Surjectivity of the Round Function

The round function involves generating a word to add (modulo the wordsize) to the target block. However, due to a flaw in the round function's design, we can significantly limit the possible output values. All output values are possible from F. However, the data dependent rotation in v limits the possible values actually added to the target block. In other words, the combined function $v \circ F$ is non-surjective. Rijmen et al [7] identified several attacks on ciphers with non-surjective round functions, so this property of SPEED is worrisome.

Applying the attack of [7] is not as straightfoward as one might hope. There Rijmen et al depend on the fact that the range of the h-function is known and hence one can perform hypothesis testing based upon the combined output of several h-functions. However, for SPEED the output of each h-function is combined with a subkey and hence the range is unknown. Fortunately we know the shape that the distribution should take and it would appear that one can modify the analysis to perform hypothesis testing using the shape of the entire distribution rather than individual values of the distribution.

We have performed a preliminary analysis of a modified version of the cipher in which the data independent rotate is removed from each round. This allows us to write out an equation which is the sum of 6 subkeys (assuming a 48-round cipher), the outputs of 6 h-functions, part of the input block, and part of the output block. It appears that one simply performs $2^{w/8}$ hypothesis tests, one for each possible value of the sum of the 6 subkeys, selecting the value which produces the closest distribution to the output of 6 h-functions.

Analysis gets substantially more difficult when the data independent rotate is taken into account, since the carry bits that "cross the rotation boundary" spoil naive attempts to isolate the sum of the h-function outputs from the sum of the subkeys. Nonetheless, for larger word sizes the spread of the carry bits is limited. We conjecture that it may be possible to extend the technique to handle the data independent rotations in these cases, though the analysis will almost certainly be much messier.

3.2 Implications of Correlated Outputs

The outputs of successive round functions are correlated. For instance,

$$F_1(x_6, \ldots, x_0) = F_1(x_7, \ldots, x_1)$$

with probability $1/2 + 1/32$ over all choices of $x_0, \ldots, x_7$. This shows that there are non-trivial correlations between successive outputs of F_1; this could potentially lead to correlations between successive outputs of h. Similar correlations occur for F_3 and F_4, though this property does not seem to hold for F_2.

We have not been able to extend this troublesome weakness to a full attack on SPEED. Nonetheless, this gives another indication of how the SPEED design may get a great deal less strength than expected from the very strong Boolean functions chosen for it. If successive h-function outputs are correlated strongly enough, this could make linear or differential-linear attacks possible on the cipher. We leave this as an open question for future research.

3.3 Differential Characteristics

The key insight necessary for mounting a differential attack on this round function is that F is a very good nonlinear function, but it applies it to each bit position of each source block word independently. In other words, F exhibits very poor diffusion across bit positions. Therefore, we see immediately that flipping any one bit in the input of F can only affect one output bit of F.

In particular, if the underlying Boolean function behaves "randomly," flipping an input bit of F should leave its output unchanged with probability 1/2.

This would appear, at first glance, to yield a very simple eight-round iterative differential characteristic with probability approximately 2^{-8}. However, there is a problem. The straightforward attack doesn't work; we explain why below.

Complications There are two complications to our differential attack. First, the Boolean functions aren't really random, and so don't have quite the probability distribution we would have expected. Table 1 lists the probability that the output of F remains unchanged after flipping one input bit in the input at position i.

	0	1	2	3	4	5	6
F_1	.5	.5	.5	.5	.5	.5	.5
F_2	.5	.5	.5	.25	.5	.5	.75
F_3	.5	.5	.5	.5	.5	.5	.5
F_4	.5	.5	.5	.5	.5	.5	.5

Table 1. Probability that the output of F_i remains unchanged after flipping one input bit.

The second complication is much, much more problematic. SPEED is, in the terminology of [9], a source-heavy UFN. Furthermore, there is no key addition before the input of the F-function. This means that the inputs to the Feistel F-function in successive rounds can't be assumed to be independent, as they generally can be in a balanced Feistel network or in a target-heavy UFN.

If the inputs to successive rounds' F-functions aren't independent, then it's possible that the success probabilities of each round's differential characteristic are also not independent. In fact, taking this effect into account, we find that the probability for six of the eight possible eight-round (one-bit) iterative characteristics is precisely 0—the inter-round dependence makes it impossible for the characteristic to penetrate more than eight rounds. When extended to an eleven-round characteristic (across F_2), the characteristic always has a probability of 0. However, later in this paper we will show how to fix the attack by tweaking the differential characteristic. See Section 4.

The problem of inter-round dependence was mentioned as a theoretical possibility in [9]; here we give a practical example where it arises. The dependence also arises again in Section 6, where precisely this difficulty complicates a related-key attack on the cipher.

4 More on Differential Characteristics

Differential characteristics are also possible with more than one bit at the same bit position set. The flipped bits rotate through different bit positions (due to the constant rotate in the round function), but end up in the same bit position for most rounds. We will discuss the use of such characteristics later in this paper.

As we have already noted, the obvious 1-bit differential attack against the cipher does not work. The problem is that we will have a hard time ramming our differential through four rounds where F_2 is the F-function used for each round. Suppose that we have the 3-round characteristic in Table 2, starting with round i.

The problem is that this characteristic is impossible and we will not have the desired differences after round $i+1$. The reason this characteristic always fails is that successive outputs of F_2 are correlated, as we saw in Section 3.2. This means that the characteristic's round probabilities are not independent, so we cannot simply multiply them to obtain the full probability. It is fairly easy to see that any 1-bit characteristic across 11 rounds of F_2 will necessarily have this 3-round characteristic, and hence the differential fails to attack the cipher with 44 or more rounds.

r	$\Delta m_{i,7}$	$\Delta m_{i,6}$	$\Delta m_{i,5}$	$\Delta m_{i,4}$	$\Delta m_{i,3}$	$\Delta m_{i,2}$	$\Delta m_{i,1}$	$\Delta m_{i,0}$	P
-	0	0	0	0	0	0	0	A	-
i	0	0	0	0	0	0	A	0	1/2
i+1	0	0	0	0	0	A	0	0	1/2
i+2	0	0	0	0	A	0	0	0	1/2

Table 2. Failed 1-bit Differential Characteristic. $\Delta m_{i,j}$ is the value of the difference in data block j at the input of round i.

Unfortunately when trying to find a differential characteristic that would work we found that even slightly more complicated differentials also failed. Consider the 2-bit (8 round) differential with probability 2^{-10} given in Table 3. We found that repeatedly concatenating this differential causes it to fail after at most 12 rounds (assuming a 48-round cipher). In the next section we discuss the analysis we used to determine that both of the above differentials would fail.

r	$\Delta m_{i,7}$	$\Delta m_{i,6}$	$\Delta m_{i,5}$	$\Delta m_{i,4}$	$\Delta m_{i,3}$	$\Delta m_{i,2}$	$\Delta m_{i,1}$	$\Delta m_{i,0}$	P
-	0	0	0	0	0	0	A	A	-
i	0	0	0	0	0	A	A	0	1/2
i+1	0	0	0	0	A	A	0	0	1/2
i+2	0	0	0	A	A	0	0	0	1/2
i+3	0	0	A	A	0	0	0	0	1/2
i+4	0	A	A	0	0	0	0	0	1/2
i+5	A	A	0	0	0	0	0	0	1/4
i+6	A	0	0	0	0	0	0	B	1/4
i+7	0	0	0	0	0	0	B	B	1/2

Table 3. Failed 2-bit Differential Characteristic.

4.1 A Failure Test for Differential Characteristics

If we make the assumption that all subkeys are independent and random, then there is a rather simple analysis technique we can apply to determine when a characteristic might fail. The technique will never fail a legitimate characteristic, but it may give a false positive. Therefore, passing the test does not imply that a characteristic will work.

The first observation is that with high probability, in characteristics such as those found in Table 2 and Table 3, bit j of $\Delta m_{i,7}$ will only be affected by bit j of the remaining $m_{i,k}$ and bit j of the subkey. Hence one can construct a state transition diagram in which each node contains the round number, bit j of the appropriate subkey, and bit j of each $m_{i,k}$ (for both halves of a pair satisfying the characteristic). We connect all nodes using directed arcs if one node is a legitimate successor of the other. This requires that the round numbers differ by 1, the $m_{i,k}$ satisfy the appropriate relations (as defined by the round function), and the output difference of the appropriate F-function applied to the $m_{i,k}$ is 0.

Once we have constructed such a graph we can view it as several different layers — one for each round of the characteristic. Clearly all edges originating from layer i in this construction will end in layer $i+1$. Therefore we can construct an adjacency matrix A_i for the transition from layer i to $i+1$ and an overall adjacency matrix $A = \prod_i A_i$. The test we propose is to check whether A is the zero matrix. If it is, there will be no transition from a starting state to an ending state which satisfies our characteristic. Therefore we can eliminate those characteristics for which A is the zero matrix.

There is a small complication in that the last few rows of both of the characteristics we proposed above show non-zero bit differences in other bit positions. However, we observed that the remaining bits adjacent to bit j' (where $B = 2^{j'}$) are effectively random. Therefore to simplify our analysis we viewed the resulting 40-round characteristic as several superimposed n-round characteristics. We made the simplying assumption that each characteristic was independent of the others and hence were only concerned that each of the characteristics was independently possible. This assumption seems well justified, especially given that it will not eliminate legitimate characteristics (and we are only presently concerned with eliminating bogus characteristics).

Using these techniques, we examined each the characteristics given in Table 2 and Table 3. Our analysis found that each of these characteristics would not work so we were able to eliminate them. In fact, we analyzed the next most obvious 2-bit characteristic given in Table 4 but found that it also fails.

r	$\Delta m_{i,7}$	$\Delta m_{i,6}$	$\Delta m_{i,5}$	$\Delta m_{i,4}$	$\Delta m_{i,3}$	$\Delta m_{i,2}$	$\Delta m_{i,1}$	$\Delta m_{i,0}$	P
-	0	0	0	0	0	A	0	A	-
i	0	0	0	0	A	0	A	0	1/2
i+1	0	0	0	A	0	A	0	0	1/2
i+2	0	0	A	0	A	0	0	0	1/2
i+3	0	A	0	A	0	0	0	0	1/2
i+4	A	0	A	0	0	0	0	0	1/4
i+5	0	A	0	0	0	0	0	B	1/4
i+6	A	0	0	0	0	0	B	0	1/4
i+7	0	0	0	0	0	B	0	B	1/2

Table 4. Another Failed 2-bit Differential Characteristic.

Fortunately, the differential given in Table 5 did not fail. In fact, one can construct independent round keys, a plaintext, and a ciphertext for which the differential holds. However, there appears to be one small difficulty even with this differential in that it will not work for *all* keys. Further research may reveal a differential (or family of differentials) which will work against all keys.

r	$\Delta m_{i,7}$	$\Delta m_{i,6}$	$\Delta m_{i,5}$	$\Delta m_{i,4}$	$\Delta m_{i,3}$	$\Delta m_{i,2}$	$\Delta m_{i,1}$	$\Delta m_{i,0}$	P
-	0	0	0	0	0	A	A	A	-
i	0	0	0	0	A	A	A	0	1/2
i+1	0	0	0	A	A	A	0	0	1/2
i+2	0	0	A	A	A	0	0	0	1/2
i+3	0	A	A	A	0	0	0	0	1/2
i+4	A	A	A	0	0	0	0	0	1/4
i+5	A	A	0	0	0	0	0	B	1/8
i+6	A	0	0	0	0	0	B	B	1/4
i+7	0	0	0	0	0	B	B	B	1/2

Table 5. A Partially Successful 3-bit Differential Characteristic.

5 Mounting an Effective Differential Attack

The important point that we should remember from Section 4 is that in general, input differences with a small Hamming weight with large probability cause output differences with a small

Hamming weight. Even if a pair does not follow one of the decribed characteristics, the Hamming weight of the output difference will usually be about the same as the Hamming weight of the input difference. This behaviour is quite similar to that of RC2 [4] and RC5 [8]. Therefore we will use a variant of the differential attack that has been developed to attack RC5 [2] and is also used on RC2 [4].

5.1 Differentials and Characteristics

It has been observed [5] that the effectiveness of a differential attack depends on the probability of a *differential*, rather than on the probability of a characteristic. Indeed, when applying a differential attack to a block cipher, we are only concerned with the values of the differences in the first and last few rounds. The intermediate differences are not important. The analysis of RC2 has shown that in ciphers with limited diffusion, the difference in probability between characteristics and differentials may be significant.

We verified experimentally that there exist several 12-round differentials that go from a one-bit input difference to a one-bit output difference with significant probability, even for the cases where it is difficult to find a differential characteristic with nonzero probability (cf. Section 4). These 12-round differentials can be combined to produce longer differentials. For example, the 48-round differential with input difference $(0, 0, 0, 0, 0, 40, 0, 0)$ (in base-16) and output difference $(80, 0, 0, 0, 0, 0, 0, 0)$ (in base-16) has probability 2^{-60} (this holds exactly for 64-bit blocks, but the probability stays approximately the same for larger block lengths).

In fact, in our attack we will loosen the restrictions on the output difference, and consider for the last round only the Hamming weight of the difference, rather than specific values.

5.2 Key Recovery

The key recovery procedure works as follows. We choose pairs of plaintexts with a non-zero difference in one bit of B_0 only. We select the texts such that the output difference of h in the first round is zero. This is made particularly easy by the absence of a key addition at the inputs of F. Whether the output of F_1 in the second round is also zero, as required by the characteristic, depends on the plaintexts and the unknown value of the first round key only.

When we find a pair with a small output difference, we assume that it follows the characteristic in the second round. This gives us information about the value of the first round key. By judicious choice of the plaintexts, we can determine the bits of the first round key a few at a time. More requirements could be imposed on the plaintext pairs, in order to make sure that all pairs pass the first two rounds with probability one. This would, however, complicate the key recovery phase.

5.3 Filtering Issues

A basic, non-optimal approach is to use a characteristic that determines the differences until the last round of the cipher. This ensures that all wrong pairs are filtered, but the probability of the characteristic is quite low. The fact that differences spread very slowly through the rounds of SPEED can be used to relax the conditions on the pairs. Instead of requiring that B_7 of the output difference equals 80 (in base-16) and that the remaining B_i have difference zero we will just accept any pair where the Hamming weight of the output difference is below some threshold H. This improves the probability of the characteristic, because pairs are allowed to "fan out" in the last rounds. The disadvantage is that it becomes possible that wrong pairs are accepted. In order to get correct results, a value of H has to be selected such that the expected number of wrong pairs after filtering is below the expected number of good pairs. A block length of 128 or 256 allows for a larger H. Therefore, versions of SPEED with a block length of 128 or 256 bits require a higher number of rounds than the 64-bit version to be secure against the differential attack. The signal-to-noise ratio of the attack can be improved by using more sophisticated filtering techniques that are described in [2].

5.4 Experimental Results

We implemented differential attacks on versions of SPEED with a reduced number of rounds. The results are given in Table 6. For versions with more than 28 rounds, the plaintext requirements become impractical. From the obtained results, we estimate that a differential attack on SPEED with R rounds requires at least $2^{2(R-11)}$ plaintext pairs. Because of the effects described in Section 4, the plaintext requirements will probably increase even more if $R \geq 44$. This means that SPEED with 48 rounds and a block length of 64 bit is secure against our differential attack. For versions with a block length of 128 or 256 bit, more than 64 rounds are needed to obtain adequate resistance.

# rounds	success rate	# pairs
16	100%	2^{10}
20	100%	2^{18}
24	100%	2^{25}
28	80%	2^{31}

Table 6. Experimental results for reduced versions of SPEED. The numbers in the first three rows are obtained from 100 experiments each, the numbers in the last row are obtained from 10 experiments.

6 Related Key Attack

Related key attacks were first described in [1]. They are a class of attacks in which one examines the results of encrypting the same ciphertext under related keys. We perform such an analysis to produce collisions in the encryption function: two keys which encrypt the same plaintext to the same ciphertext.

In [12] the author suggests using SPEED in a variant of Davies-Meyer hashing in order to transform SPEED into a hash function. Specifically, a message is padded to a multiple of 256 bits by appending unique padding and a length (the exact details of the padding are beyond the scope of this paper). The resulting message M, is split into 256-bit chunks $M_0, M_1, \ldots, M_{n-1}$. The hash is D_n where $D_0 = 0$ and $D_i = D_{i-1} + E_{M_{i-1}}(D_{i-1})$. $E_K(X)$ denotes the encryption of X with key K. (Addition is defined slightly differently, but the exact definition does not affect our attack so we omit it.)

The cipher may be 64, 128, or 256 bits wide and hence so may the corresponding hash (although 64 bits would easily fall to a conventional birthday attack). Furthermore, the author of [12] suggests using 32 to 48 rounds for efficiency. We have successfully produced hash collisions for the 128-bit hashes with 32 rounds and also for 48 rounds. Using the reference implementation obtained from the author, we found the following collision for the 128-bit hash with 32 rounds (in base-16):

```
M  = 21EA FE8E 1637 19F7 22D2 8CCB 3724 3437
     B00F 7607 3C91 3710 2B69 C9C9 58FB 0823
     AEC2 CD05 FD80 14E6 B11E 43C0 5767 76F7
     FF07 17EC FCBA 224E 9627 A16A 8D6E 83A9

M' = 21EA FE8E 1637 19F7 22D2 8CCB 3724 3437
     B00F 7607 3C91 3710 2B69 C9C9 58FB 0823
     AEC2 CD05 FDC0 14E6 B11E 4380 5767 76F7
     FF07 17EC 7CBA 224E 9627 216A 8D6E 83A9
```

This leads to the following values when hashing (in base-16):

D_0 = 0000 0000 0000 0000 0000 0000 0000 0000
D_1 = 90DA 7F34 46FA A373 B048 11F7 F8D9 BB3D
$D_2 - D_1$ = 9781 9517 B5CC A046 D0F1 3719 ED9B A0B6

D_0 = 0000 0000 0000 0000 0000 0000 0000 0000
D_1 = 90DA 7F34 46FA A373 B048 11F7 F8D9 BB3D
$D_2 - D_1$ = 9781 9517 B5CC A046 D0F1 3719 ED9B A0B6

We also found the following collision for the 128-bit hash with 48 rounds (in base-16):

M = 3725 6571 48D5 CF52 DAE1 4065 7115 11A0
E3C5 9428 7BFD 18CB EF79 82BB 1D7F 2F55
36F2 CD58 9058 FE57 D696 EA4C BD75 F7C9
1989 A048 39FB 9B76 9011 CAC0 65F6 EBC7

M' = 3725 6571 48D5 CF52 DAE1 4065 7115 11A0
E3C5 9428 7BFD 18CB EF79 82BB 1D7F 2F55
38F2 CB58 9058 FC57 D896 EA4C BD75 F7C9
1985 A04C 39FB 9B7A 900D CAC0 65F6 EBC7

This leads to the following values when hashing (in base-16):

D_0 = 0000 0000 0000 0000 0000 0000 0000 0000
D_1 = DA2B A119 A4F8 AA70 59ED 6FE4 188B 7969
$D_2 - D_1$ = CAB1 DA86 B6D3 1442 E05C A005 7B26 C432

To produce collisions, we combine two different attacks:

1. A differential attack against the key schedule which produces two key schedules with a desired difference.
2. A related key attack against the cipher using the two related key schedules.

We feel that the attack is more illuminating when we address these two attacks in the opposite order. Therefore we will describe the related key attack first in order to give a motivation for the attack against the key schedule.

6.1 Related-Key Attack Against the Cipher

The fundamental observation that makes this attack possible is that a 1-bit input difference to any of the four F-functions will produce a zero output difference with probability 1/2 (actually this doesn't quite hold for F_2, but this doesn't seem to strongly affect experimental results). In our attack, we attempt to introduce 1-bit differences into the encryption state through the key schedule. We do this in such a way so that after several rounds, the introduced differences negate each other and the resulting encryption state is the same for both keys. In Table 7 of Appendix 1, we have shown all 32 rounds of our related-key attack. In summary, we encrypt a message with two different keys where the subkeys for rounds 1, 4, 9, 12, 17, and 25 have specific differences so that the encryption state under the two keys will be identical in rounds 13–16 and rounds 26–32. Initially the two encryption states are the same, and after round 25, the two encryption states will be identical with total probability 2^{-19}. Since the remaining keywords of the key schedule are identical, the probability that the same plaintext will encrypt to the same ciphertext under the two different key schedules is 2^{-19}.

Note, there are four variations on this attack in which we add 1 to 4 rounds prior to round 1 in which the two key schedules are identical. Hence the subkeys for rounds $t+1$, $t+4$, $t+9$, $t+12$, $t+17$, and $t+25$ with $t \in \{0,1,2,3,4\}$ are given the specified differences. If we define variant t to be the above differential with t additional initial rounds, then the desired key differences are:

$$\Delta K[i] = \begin{cases} 2^j & \text{if } i \in \{t+1, t+17\} \\ -2^j & \text{if } i \in \{t+4\} \\ -(2^j)^{\gg 7} & \text{if } i \in \{t+9, t+25\} \\ (2^j)^{\gg 7} & \text{if } i \in \{t+12\} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The collisions we found at the beginning of this section were for variant 2 with $j = 64$.

Subtle Difficulties with the Attack

It is not hard to see that our attack makes the simplifying assumption that the inputs to each round are independent and uniformly distributed. Hence any pair of inputs that will lead to the desired difference is possible. However, in many cases this is an incorrect assumption and it can strongly affect our attack.

To make the discussion easier, we will examine a smaller version of the cipher with 1-bit words. This is a fair thing to do since adjacent bits in a word affect each other within the F-function only through the data dependent rotate. As stated in Section 2, the F-function is composed of a non-linear function F_i ($i \in \{1,2,3,4\}$) composed with a data-dependent rotate v. If the output difference for F_i is zero, then given two different inputs the output difference of the data-dependent rotate will also be zero. This means that a 1-bit difference in any word will not change affect any adjacent bits if the output difference of F_i is zero.

Once we have made the reduction, we can regard the sequence of rounds as the successive states of a non-linear shift register. Specifically, we have a sequence of states X_i where X_0 is the input to our encryption function, $X_{i+1} = (X_i || (k_{i+1} \oplus F_k(X_i)))_{0\ldots6}$, where F_k can be F_1, F_2, F_3 or F_4, depending on the particular round number. The output of the cipher is then X_r where r is the number of rounds.

In our 32-round related-key attack, $k_i \neq k'_i$ for a small number of i. If one examines the sequence of states produced by the same initial state X_0 and two related keys k_i and k'_i, then for variant j we want that $X_{j+16} \oplus X'_{j+16} = 0$ and $X_{j+17} \oplus X'_{j+17} = 2^0 = 1$. For $i = 0, \ldots, 6$, $k_{i+j+17} = k'_{i+j+17}$ so we can examine a simplified shift register whose feedback function is $Y_{i+1} = (Y_i || (k_{i+1} \oplus G_k(Y_i)))$, where $G_k = F_3$ if $0 < i \leq 6 - j$, and $G_k = F_4$ if $6 - j < i \leq 6$, $Y_0 = X_{j+17}$, $Y'_0 = X'_{j+17}$, and k is an arbitrary key. That is, we can consider these 7 rounds (round $j+17$ to $j+23$) in isolation since the subkeys are the same for both values of our key.

We want to examine the sequence of states Y_i and Y'_i where $Y_0 \oplus Y'_0 = 1$. In order for our related-key attack to work, we must have that $Y_i \oplus Y'_i = 2^i$ for $0 \leq i \leq 6$. Unfortunately it turns out that for certain j (e.g. $j = 0, 1$), there are no input states Y_0 and Y'_0 and key k for which the resulting sequences have this property. We determined this by performing a brute-force search with a computer. There are only 64 different k (k_6 really has no influence) and 64 different Y_0 to consider. Hence we need only examine $64 \cdot 64 = 4096$ different cases.

Our analysis showed that variants 0 and 1 will not produce the desired collisions, but that variants 2, 3, and 4 will. In performing a computer search we found that we were not able to find collisions within the expected time for variants 0 and 1, providing evidence for the correctness of our analysis. The collision we provided above was for variant 2, showing that there does exist a satisfying sequence of states.

Extending the Attack to 48-rounds

Unfortunately it is not as straightforward as one might hope to extend our attack to 48 rounds. By duplicating the attack of rounds 17–32 for rounds 33–48, in Table 7, we obtain a plausible looking related-key attack for the 48-round version of the cipher. In summary, we have an additional sixteen subkeys, two of which have specific differences (the subkeys for rounds 33 and 42). However, in attempting to perform a computer search for collisions, we found that we were unable to produce collisions after the first 32 rounds, let alone the entire 48 rounds. This is what initially led us to the analysis performed in the previous section.

It turns out that for the 48-round version of the cipher, the five different variants of the attack result in a shift register with feedback function:

$$H_k(X,i) = \begin{cases} F_2(X) \oplus k_i & \text{if } 0 < i \le \min(7-j, 6) \\ F_3(X) \oplus k_i & \text{if } 7-j < i \le 6 \end{cases}$$

where j is the variant we wish to examine. For the resulting shift register, there are no initial states Y_0 and Y_0' and key k such that $Y_i \oplus Y_i' = 2^i$ for $0 \le i < 7$. Consequently our related-key attack will fail to produce a zero output difference after round 32.

A slightly modified version of our attack does work. We present the first 32 rounds of the attack in Table 8 (in Appendix 1). The last 16 rounds are simply a copy of rounds 17–33. The associated differential attack on the key schedule is presented in Table 10 (also in Appendix 1). In essence we overlap two differential attacks so that differences are introduced for 9 rounds instead of 8 rounds as before. The total probability that a key will satisfy the differential is 2^{-18}. The total probability that two related keys will produce a collision is 2^{-32}.

6.2 Differential-Attack on the Key Schedule

In order to carry out our related key attack, we must find two different keys which will produce key schedules with the differences specified in (1). We performed a straightforward differential attack on the key schedule to produce the desired pair. The specifications for using SPEED as a hash function require a 256-bit key. Since we are using a 128-bit blockwidth, this implies that the first 16 subkeys will be our key and the remaining subkeys will be derived from the key.

The key scheduling algorithm is straightforward and we include a modified copy from [12] here:

Step 1. Let $kb[0], kb[1], \ldots, kb[31]$ be an array of double-bytes where $kb[0], \ldots, kb[15]$ are the original 16 double-byte values of the key.

Step 2. This step constructs $kb[16], \ldots, kb[31]$ from the user key data $kb[0], \ldots, kb[15]$. It employs three double-byte constants $Q_{l,0}$, $Q_{l,1}$, and $Q_{l,2}$.

1. Let $S_0 = Q_{l,0}$, $S_1 = Q_{l,1}$, and $S_2 = Q_{l,2}$.
2. For i from 16 to 31 do the following:
   (a) $T = G(S_2, S_1, S_0)$.
   (b) Rotate T to the right by 11 bits.
   (c) $T = T + S_2 + kb[j] \pmod{2^{16}}$, where $j = i \pmod{16}$.
   (d) $kb[i] = T$.
   (e) $S_2 = S_1$, $S_1 = S_0$, $S_0 = T$.

where

$$G(X_0, X_1, X_2) = (X_0 \oplus X_1) \wedge (X_0 \oplus X_2) \wedge (X_1 \oplus X_2).$$

The two crucial observations about the key scheduling algorithm are:

1. Adjacent bits in the input have minimal influence on each other. Hence we can work as if it took 1-bit inputs and produced a 1-bit output.
2. When viewed as a function on 1-bit inputs, flipping one or two inputs will flip the output with probability exactly 1/2.

7 Practical Considerations: Performance and Security

SPEED is defined as having both a variable block length and a variable number of rounds. Since almost any Feistel network is secure after enough rounds, what does it mean to say that SPEED is "broken"? For a cryptographic engineer, this means that the still-secure variants of SPEED are significantly slower than other secure alternatives.

Table 2 compares the throughput of SPEED with the throughput of other block ciphers. Benchmarks for the other algorithms were taken from [10, 6][1]. We only include the SPEED block-length and round-number pairings that we believe are secure.

Algorithm	Block width	# rounds	Clocks/byte of output
Blowfish	64	16	19.8
Square	128	8	20.3
RC5-32/16	64	16	24.8
CAST-128	64	16	29.5
DES	64	16	43
SAFER (S)K-128	64	8	52
IDEA	64	8	74
Triple-DES	64	48	116
SPEED	64	64	160
SPEED	64	80	200
SPEED	64	96	240
SPEED	128	64	80
SPEED	128	80	100
SPEED	128	96	120
SPEED	256	64	40
SPEED	256	80	50
SPEED	256	96	60

Fig. 2. Comparison of Different Block Ciphers.

8 Conclusions

In this paper, we have discussed the SPEED proposed block cipher in terms of cryptanalytic attack. We have pointed out a few potential weaknesses, demonstrated an attack on the Davies-Meyer hashing mode of SPEED with 32 and 48 rounds, and explored other attacks on the 48-round SPEED block cipher.

It is interesting to note that SPEED, though built using very strong component functions, doesn't appear to be terribly secure. The SPEED design apparently relied upon the high quality of the binary functions used, the fact that different functions were used at different points in the cipher, and the data-dependent rotations to provide resistance to cryptanalysis. Unfortunately, the most effective attacks aren't made much less powerful by any of these defenses.

It is also interesting to note the new difficulties that occur in attacking this kind of cipher. The source-heavy UFN construction of SPEED forced us to reconsider our assumptions about carrying out differential attacks, and added somewhat to the difficulty of thinking about these attacks. Surprisingly many of the obvious differential attacks did not work against the cipher, although it's not obvious that a different choice of F-functions would present the same problems.

[1] In [12], the author compares differet variants of SPEED with IDEA. Since [10, 6] both benchmark IDEA twice as fast as [12], we performed our own efficiency analysis on SPEED. We estimate that a fully optimized version of SPEED on a Pentium will take 20 clock cycles per round, for word sizes of 64 bits, 128 bits, and 256 bits.

References

1. E. Biham, "New Types of Cryptanalytic Attacks Using Related Keys," *Journal of Cryptology*, v. 7, n. 4 (1994), pp. 229–246.
2. A. Biryukov and E Kushilevitz, "Improved cryptanalysis of RC5," *Advances in Cryptology, Proc. Eurocrypt'97, LNCS*, to appear.
3. J. Kelsey, B. Schneier, and D. Wagner, "Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES," *Advances in Cryptology—CRYPTO '96*, Springer-Verlag, 1996, pp. 237–251.
4. L.R. Knudsen, V. Rijmen, R.L. Rivest and M.J.B. Robshaw, "On the design and security of RC2," *Fast Software Encryption, LNCS 1372*, S. Vaudenay, Ed., Springer-Verlag, 1998, pp. 206–221.
5. X. Lai, J.L. Massey, and S. Murphy, "Markov ciphers and differential cryptanalysis," *Advances in Cryptology, Proc. Eurocrypt'91, LNCS 547,* D.W. Davies, Ed., Springer-Verlag, 1992, pp. 17–38.
6. B. Preneel, V. Rijmen, and A. Bosselaers, "Recent developments in the design of conventional cryptographic algorithms," *Computer Security and Industrial Cryptography - State of the Art and Evolution, LNCS*, Springer-Verlag, to appear.
7. V. Rijmen, B. Preneel, E. De Win, "On weaknesses of non-surjective round functions," *Designs, Codes, and Cryptography*, Vol. 12, No. 3, November 1997, pp. 253-266.
8. R.L. Rivest, "The RC5 encryption algorithm," *Fast Software Encryption, LNCS 1008*, B. Preneel, Ed., Springer-Verlag, 1995, pp. 86–96.
9. B. Schneier, J. Kelsey, "Unbalanced Feistel Networks and Block Cipher Design," *Fast Software Encryption–Third International Workshop*, Springer-Verlag, 1996.
10. B. Schneier, D. Whiting, "Fast Software Encryption: Designing Encryption Algorithms for Optimal Software Speed on the Intel Pentium Processor," *Fast Software Encryption–Fourth International Workshop*, Springer-Verlag, 1997.
11. R. Winternitz and M. Hellman, "Chosen-key Attacks on a Block Cipher," *Cryptologia*, v. 11, n. 1, Jan 1987, pp. 16–20.
12. Y. Zheng, "The SPEED Cipher," in Proceedings of *Financial Cryptography '97*, Springer-Verlag.

Appendix 1

This section contains the details for the two related key and differential attacks we performed against SPEED.

Note, the first column of Table 7 (and also Table 8) shows the additive differences between the respective words of the two key schedules. For example, $K'[0]-K[0]=64$. Note that in the 128-bit version of the cipher, each word is 16 bits wide. Hence $m_{i,7}$ (the target block) will be rotated right $16/2-1=7$ bits and $64^{\gg 1}=2^{15}$. With probability 2^{-12}, two different key schedules with the specified differences will have the exact same encryption state after round 12.

R	$\Delta K[i]$	$\Delta m_{i,7}$	$\Delta m_{i,6}$	$\Delta m_{i,5}$	$\Delta m_{i,4}$	$\Delta m_{i,3}$	$\Delta m_{i,2}$	$\Delta m_{i,1}$	$\Delta m_{i,0}$	P
-	-	0	0	0	0	0	0	0	0	1^A
1	64	0	0	0	0	0	0	0	64	$1/4^B$
2	0	0	0	0	0	0	0	64	0	1/2
3	0	0	0	0	0	0	64	0	0	1/2
4	-64	0	0	0	0	64	0	0	-64	$1/4^C$
5	0	0	0	0	64	0	0	-64	0	1/2
6	0	0	0	64	0	0	-64	0	0	1/2
7	0	0	64	0	0	-64	0	0	0	1/2
8	0	64	0	0	-64	0	0	0	0	1/2
9	2^{15}	0	0	-64	0	0	0	0	0	$1/2^D$
10	0	0	-64	0	0	0	0	0	0	1/2
11	0	-64	0	0	0	0	0	0	0	1
12	2^{15}	0	0	0	0	0	0	0	0	1
13	0	0	0	0	0	0	0	0	0	1
14	0	0	0	0	0	0	0	0	0	1
15	0	0	0	0	0	0	0	0	0	1
16	0	0	0	0	0	0	0	0	0	1
17	64	0	0	0	0	0	0	0	64	1/4
18	0	0	0	0	0	0	0	64	0	1/2
19	0	0	0	0	0	0	64	0	0	1/2
20	0	0	0	0	0	64	0	0	0	1/2
21	0	0	0	0	64	0	0	0	0	1/2
22	0	0	0	64	0	0	0	0	0	1/2
23	0	0	64	0	0	0	0	0	0	1/2
24	0	64	0	0	0	0	0	0	0	1
25	2^{15}	0	0	0	0	0	0	0	0	1
26	0	0	0	0	0	0	0	0	0	1
27	0	0	0	0	0	0	0	0	0	1
28	0	0	0	0	0	0	0	0	0	1
29	0	0	0	0	0	0	0	0	0	1
30	0	0	0	0	0	0	0	0	0	1
31	0	0	0	0	0	0	0	0	0	1
32	0	0	0	0	0	0	0	0	0	1

A The initial input to the cipher is the same for both keys.

B The probability that an additive difference of 64 produces an XOR difference of 64 is 1/2. 1/4 comes from multiplying by the probability that a 1-bit input difference will produce a zero output difference for the F-function.

C The probability that an additive difference of -64 produces an XOR difference of 64 is 1/2. The XOR difference of 64 and -64 occur in the same bit-position so the probability that the output difference of the F-function is zero is 1/2.

D $(64^{\ggg 7}) + 2^{15} \equiv 0 \pmod{2}^{15}$. Hence $\Delta m_{9,0} = 0$ with probability 1 (assuming $\Delta f = 0$, which occurs with probability 1/2).

Table 7. Related Key Attack Against 32-round Cipher.

R	$\Delta K[i]$	$\Delta m_{i,7}$	$\Delta m_{i,6}$	$\Delta m_{i,5}$	$\Delta m_{i,4}$	$\Delta m_{i,3}$	$\Delta m_{i,2}$	$\Delta m_{i,1}$	$\Delta m_{i,0}$	P
-	-	0	0	0	0	0	0	0	0	1
0	64	0	0	0	0	0	0	0	64	1/2
1	64	0	0	0	0	0	0	64	64	1/4
2	0	0	0	0	0	0	64	64	0	1/2
3	-64	0	0	0	0	64	64	0	-64	1/4
4	-64	0	0	0	64	64	0	-64	-64	1/4
5	0	0	0	64	64	0	-64	-64	0	1/2
6	0	0	64	64	0	-64	-64	0	0	1/2
7	0	64	64	0	-64	-64	0	0	0	1/2
8	2^{15}	64	0	-64	-64	0	0	0	0	1/2
9	2^{15}	0	-64	-64	0	0	0	0	0	1/2
10	0	-64	-64	0	0	0	0	0	0	1/2
11	2^{15}	-64	0	0	0	0	0	0	0	1
12	2^{15}	0	0	0	0	0	0	0	0	1
13	0	0	0	0	0	0	0	0	0	1
14	0	0	0	0	0	0	0	0	0	1
15	0	0	0	0	0	0	0	0	0	1
16	64	0	0	0	0	0	0	0	64	1/2
17	64	0	0	0	0	0	0	64	64	1/4
18	0	0	0	0	0	0	64	64	0	1/2
19	0	0	0	0	0	64	64	0	0	1/2
20	0	0	0	0	64	64	0	0	0	1/2
21	0	0	0	64	64	0	0	0	0	1/2
22	0	0	64	64	0	0	0	0	0	1/2
23	0	64	64	0	0	0	0	0	0	1/2
24	2^{15}	64	0	0	0	0	0	0	0	1
25	2^{15}	0	0	0	0	0	0	0	0	1
26	0	0	0	0	0	0	0	0	0	1
27	0	0	0	0	0	0	0	0	0	1
28	0	0	0	0	0	0	0	0	0	1
29	0	0	0	0	0	0	0	0	0	1
30	0	0	0	0	0	0	0	0	0	1
31	0	0	0	0	0	0	0	0	0	1

Table 8. First 32 Rounds of 48-round Related-Key Attack.

j	$\Delta K[j]$	ΔT_a^a	ΔT_b	ΔT_c	$\Delta S2$	$\Delta S1$	$\Delta S0$	ΔG	$\Delta K[j+16]$	P
-	-	-	-	-	0	0	0	0	-	1
1	2^l	0	0	2^l	0	0	2^l	0	2^l	1/2
2	0	0	0	0	0	2^l	0	0	0	1/2
3	0	0	0	0	2^l	0	0	0	0	1/2
4	-2^l	0	0	0	0	0	0	0	0	1/2
5	0	0	0	0	0	0	0	0	0	1
6	0	0	0	0	0	0	0	0	0	1
7	0	0	0	0	0	0	0	0	0	1
8	0	0	0	0	0	0	0	0	0	1
9	$A = -(2^l)^{\gg 7}$	0	0	A	0	0	A	0	A	1/2
10	0	0	0	0	0	A	0	0	0	1/2
11	0	0	0	0	A	0	0	0	0	1/2
12	$(2^l)^{\gg 7}$	0	0	0	0	0	0	0	0	1
13	0	0	0	0	0	0	0	0	0	1
14	0	0	0	0	0	0	0	0	0	1
15	0	0	0	0	0	0	0	0	0	1
16	0	0	0	0	0	0	0	0	0	1

a. Denotes the value of T after step 2a. Similarly, T_b and T_c denote the value of T after steps 2b and 2c respectively.

Table 9. Differential Attack Against 32-round Key Schedule.

j	$\Delta K[j]$	ΔT_a^a	ΔT_b	ΔT_c	$\Delta S2$	$\Delta S1$	$\Delta S0$	ΔG	$\Delta K[j+16]$	P
-	-	-	-	-	0	0	0	0	-	1
0	64	0	0	64	0	0	64	0	64	1/2
1	64	0	0	64	0	64	64	0	64	1/2
2	0	0	0	0	64	64	0	0	0	1/2
3	-64	0	0	0	64	0	0	0	0	1/2
4	-64	0	0	0	0	0	0	0	0	1/2
5	0	0	0	0	0	0	0	0	0	1
6	0	0	0	0	0	0	0	0	0	1
7	0	0	0	0	0	0	0	0	0	1
8	2^{15}	0	0	2^{15}	0	0	2^{15}	0	2^{15}	1/2
9	2^{15}	0	0	2^{15}	0	2^{15}	2^{15}	0	2^{15}	1/2
10	0	0	0	0	2^{15}	2^{15}	0	0	0	1/2
11	2^{15}	0	0	0	2^{15}	0	0	0	0	1/2
12	2^{15}	0	0	0	0	0	0	0	0	1
13	0	0	0	0	0	0	0	0	0	1
14	0	0	0	0	0	0	0	0	0	1
15	0	0	0	0	0	0	0	0	0	1

Table 10. First 16 Rounds of Differential Attack Against 48-round Key Schedule.