On the Security of Kaweichel

Dieter Schmidt*

September 30, 2006

Abstract

In this article the block cipher Kaweichel is analysed with regard to linear cryptanalysis and differential cryptanalysis. As a result of this investigation new versions of Kaweichel are proposed with a reduced number of rounds.

1 Introduction

In this report the block cipher Kaweichel is analysed. It was first proposed in [2, 7] with little attention paid to its security. Although the s-boxes of Kaweichel are derived from the userkey and are supposed to be secret, we assume here that they are known to the cryptanalyst. It is shown that Kaweichel with 16 rounds is immune from linear cryptanalysis and suggested that it is also immune from differential cryptanalysis.

The rest of the paper is organized as follows: First we introduce the reader to the notion. After that we describe Kaweichel. A section on linear cryptanalysis follows and after that we look at differential cryptanalysis. The author concludes in section 6 and open problems are mentioned in section 7.

2 Preliminaries

Let denote: $\boxplus$ additon modulo 2^{64}, $\oplus$ addition modulo 2 (XOR) of two 64-bit words, rotr e the rotation of a 64-bit word by e positions to the right. This rotation can be written as multiplication $2^{64-e} mod 2^{64}-1$ if the value of the 64-bit word is less then $2^{64}-1$. S-box denotes a 8-bit to 64-bit substitution-box, which can be expressed as indexed addressing of 8-bit values into a table of 64-bit values.

*Denkmalstrasse 16, D-57567 Daaden, Germany, dieterschmidt@usa.com

3 Description of Kaweichel

Kaweichel is a generalized Feistel cipher. The first Feistel cipher published was the Data Encryption Standard (DES) of the U.S. government, which was based on work by IBM. It is published in [1].

When using a Feistel cipher the plaintext is first divided into two equal halves. The size of the plaintext block is typically 64 bit or 128 bit, i.e. it is a power of 2. The left half of the plaintext block is used as the input for a so called round function (F-function), which modifies the input under the control of the round key. The output of the round function is added modulo 2 (XORed) to the right half of the plaintext block. After that, the two halves are exchanged and the procedure is repeated, until the defined number of iterations (rounds) is reached. After the last iteration, the two halves of the ciphertext block are not exchanged. Thus the whole construction is self-inverse except for the order of the round keys. This means that for encryption and decryption the same hard- and software can be used, only the order of the round keys has to be inverted for decryption.

For the construction of the round function one choses usually parallel substitutions (s-boxes). The output bits of these s-boxes are permuted in order to achive diffusion. For the derivation of the round keys from the userkey one has to choose a key schedule.

The basic idea behind this construction is that a weak, iterated encryption function will result in a cryptographically strong cipher. But there minimum requirements for the round function (F-function). It should, for example, offer sufficient resistance againt differential [1] and linear cryptanalysis [4].

The construction of Kaweichel (see figures 1 and 2) differs in several points from that of a classical Feistel ciphers.

- Before the left data block is used as input for round function, a key P_i is added modulo 2^{64} to that data block.
- Rather then using a round key for the round function, the s-boxes are key dependant. This method got first widely known with the block cipher Blowfish [8]. The advantage is, that differential [1] and linear cryptanalysis [4] are not applicable, since they require the knowledge of the s-boxes.
- After the output of the round function is added modulo 2 (XORed) to the right data block, the bits of the right block are rotated to the right by a fixed amount.
- The right and left halves are exchanged after the last iteration.
- After the last iteration a pair of keys P_{32}, P_{33} is added modulo 2^{64} to both halves (final transformation).

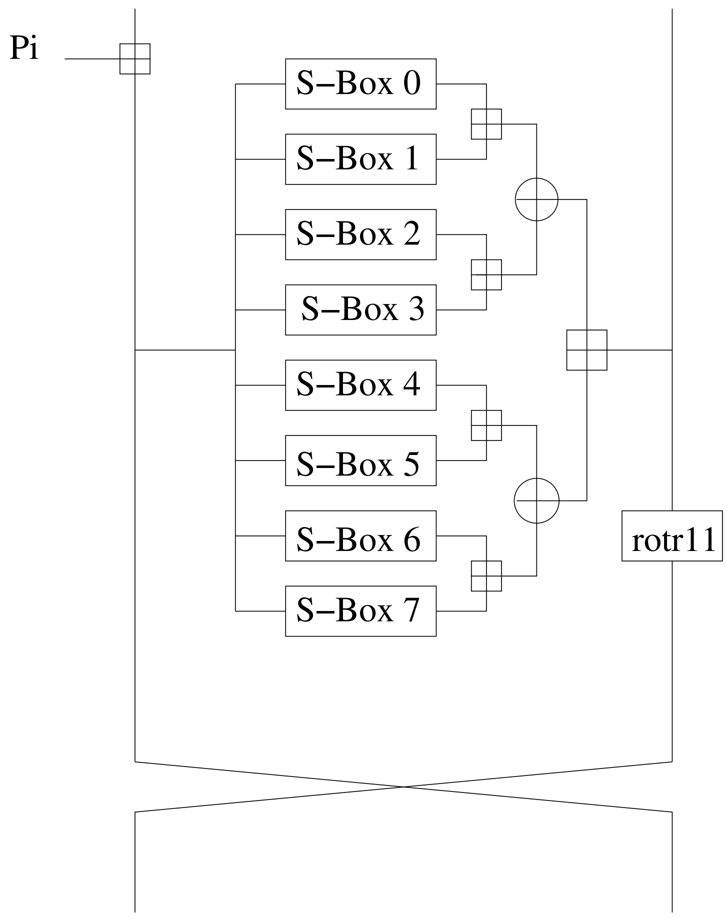

Figure 1: One iteration of the block cipher Kaweichel

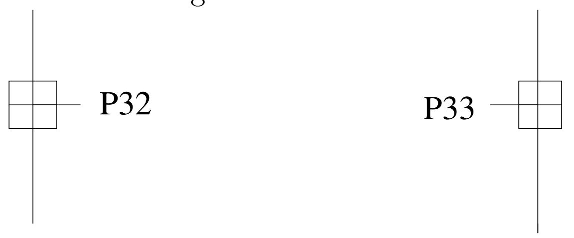

Figure 2: The final transformation

Each of the points 1 and 3 to 5 causes the cipher not to be self-inverse, i.e. for encryption and decryption separate hard- and software needs to be implemented.

Kaweichel works with a block size of 128 bit, thus each half is 64 bit long. The 64 bit input to the round function is first divided into eight equal units of eight bit (one byte). Each of these units is used as an index into a table of $256 = 2^8$ values (s-box) in such a manner, that the least significant byte is used as input for s-box 0, the next byte for s-box 1, and so on, and the most significant byte is used as input for s-box 7. The 64 bit results of two adjacent s-boxes are added modulo 2^{64}, e.g. the outputs of s-box 0 and s-box 1. This leaves four values. Two adjacent of these four values are added modulo 2(XORed). This leaves two 64 bit values. They are added modulo 2^{64} to form the output of the round function. If one denotes the output of the s-boxes by S_0 to S_7, the output of the round function becomes:
$roundoutput = ((S_0 \boxplus S_1) \oplus (S_2 \boxplus S_3)) \boxplus ((S_4 \boxplus S_5) \oplus (S_6 \boxplus S_7))$
The combination of the output of the s-boxes was changed compared to Blowfish [8] to allow for a better parallelization in both hard- and software.

For the number of rounds the author recommends for the time being $N = 32$. The maximum key length is thus 30 (i.e. $N - 2$) words of 64 bits or 1920 bits. Shorter keys are appended with zeros to reach 1920 bit, but the length of key should not fall below 256 bits.

For the rotation the value 11 is used.

For the derivation of the round keys, the keys for the final transformation and the s-boxes, the following holds: First the round keys and the keys for the final transformation are assigned random or pseudorandom values. After that the s-boxes are assigned random or pseudorandom values, beginning with s-box 0 and Index 0 (for details, see the function init_cipher in the reference implementation). In the reference implementation (see [6]) the binary digits of π (less the initial 3) are used for that purpose. After that, the 1920 bit long userkey is added modulo 2 (XORed) to the first 30 roundkeys $P_i, i = 0 \ldots 29$. This limitation of the userkey ensures that in the following encryptions all output bits depend on all the bits of the userkey. After that the plaintext block is assigned the all-zero string and encrypted once. The round keys P_0 and P_1 are then assigned the right half and the left half of the ciphertext block. The cipher is then employed in Output Feedback Mode (OFB) and the values generated are assigned the next round keys P_2 and P_3. This is repeated, until all the round keys, the keys of the final transformation and the s-boxes have been assigned new values (see function expand_key of the reference implementation).

4 Linear Cryptanalysis

Linear Cryptanalysis was first introduced by Matsui in [4]. If P denotes the plaintext, C the ciphertext and K the key, it looks for linear approximations of the form

$$\mathrm{P}_{ij} \oplus P_{ik} \oplus \ldots \oplus P_{im} \oplus C_{in} \oplus C_{io} \oplus \ldots \oplus C_{ip} = K_{iq} \oplus K_{ir} \oplus \ldots \oplus K_{is} \quad (1)$$

for certain bits of the plaintext, the ciphertext and the key. Let p be the probability, that this expression holds. Then $\epsilon = |p - \frac{1}{2}|$ denotes the bias of the linear approximation. If several approximations are combined, then for the resulting bias ϵ the Piling-Up Lemma holds:

$$\epsilon = 2^{n-1} \prod_{i=1}^{n} \epsilon_i \quad (2)$$

. The plaintext requirement for this attack ist given by $N = \epsilon^{-2}$.

Recent research published in [5] is very useful to analyse Kaweichel, since its keys are added modulo 2^{64}. This new research proves that for addition modulo 2^n the best linear approximation for a bit $x_i, 0 \leq i \leq n-1$ of the result is $x_i = p_i \oplus k_i$ if p and k are added with bias $\epsilon_i = 2^{-(i+1)}$. If we neglect the nonlinearity added by the round function for sake of simplicity, the logarithm of the biases is given by $-\log_2 \epsilon_i = (i+1)$. Thus by adding the positions increased by one of the plaintext bits or intermediate bits as they propagate through the cipher one gets the right results. For the resulting logarithm of the bias the Piling-Up Lemma is applied and looked for the minimum. A simple computer program will do that job. The result is that for 16-round Kaweichel, which means nine additions for each half:

$$\mathrm{PL}_{13} \oplus CL_{53} = K_{1,13} \oplus K_{3,2} \oplus K_{5,55} \oplus K_{7,44} \oplus K_{9,33} \oplus K_{11,22} \oplus K_{13,11} \oplus K_{15,0} \oplus K_{17,53} \quad (3)$$

Here PL_{13} denotes the bit at position 13 of the left plaintext half, CL_{53} denotes the bit at position 53 of the left ciphertext half and $K_{i,j}$ denotes bit j of round key i. Note that the key for the final transformation of the left half is K_{17}. This approximation holds with bias $\epsilon = 2^{-167}$, which is the largest found. The resulting complexity of this attack is $N = \epsilon^{-2} = 2^{334}$. Since there are only 2^{128} known plaintexts available, we conclude that 16-round Kaweichel is immune from linear cryptanalysis.

5 Differential Cryptanalysis

We assume here, that the reader is familiar we the notions of differential cryptanalysis. If not, we refer her or him to the literature, especially [1]. For this section extensive use was made of the algorithms given in [3]. We omit here the swapping of the left and right halves after each round. Instead we assume that odd rounds are applied from left to right and even rounds from right to left.

Consider a differential of the form:

$$0000 \quad 0000 \quad 0000 \quad 0000_x, 0000 \quad 0000 \quad 0000 \quad 0400_x \tag{4}$$

It passes round one with probability 1. After applying the rotation the right half looks like that:

$$0000 \quad 0000 \quad 0000 \quad 0000_x, 8000 \quad 0000 \quad 0000 \quad 0000_x \tag{5}$$

This differential passes the addition of the second round key with probability 1. The difference is now fed into s-box 7. Since the values of the s-boxes are created randomly, we can assume that a certain output XOR of that s-box has probability 2^{-7}. We can further assume that the output XOR spreads to all eight s-boxes in the next round. The probability of an characteristic for 16 rounds based on these values is:

$$P=1*2^{-7}*\prod_{i=3}^{16}(2^{-7})^8 = 1*2^{-7}*\prod_{k=3}^{16}2^{-56} = 2^{-771} \tag{6}$$

We assume here that the differential probability of the additions in the round functions and of the keys is 1. That looks impressive, but we can do better then that. Go back to round 2 and assume that a XOR value different from zero occurs only in bits 10 to 3 of s-box 7. This implies that for two outputs of that s-box the other 56 bits are equal. To derive the probability for such an event, we look at the probability that a set of 256 randomly selected 56-bit numbers contains no collision. That probability is:

$$P=\prod_{i=0}^{255}(1-\frac{i}{2^{56}}) = \prod_{i=0}^{255}\frac{2^{56}-i}{2^{56}} = \frac{(2^{56})!}{(2^{56})^{256}*(2^{56}-256)!} \approx 1-2^{-41} \tag{7}$$

The probability that two or more values are equal is thus 2^{-41}. The maximum probability that this 8 bit XOR-value of s-box 7 remains confined to bits 10 to 3 in one following addition of that round is $p = 2^{-1}$ We have for this one round characteristiv a probability of $p_i = 2^{-1}*2^{-1}*2^{-7} = 2^{-9}$. The resulting differenc is:

$$\text{XX00} \quad 0000 \quad 0000 \quad 0000_x, 8000 \quad 0000 \quad 0000 \quad 0000_x \tag{8}$$

where X denotes a half byte with a XOR difference. In round 3 s-box 7 is again the only active s-box and the resulting difference is:

$$\text{XX00} \quad 0000 \quad 0000 \quad 0000_x, XX10 \quad 0000 \quad 0000 \quad 0000_x \tag{9}$$

In round 4 we get 2 active s-boxes. As a result we assume that the output difference of that round affects all bits with a probability of $(2^{-7})^2 = 2^{-14}$. The difference now is:

$$\text{XXXX} \quad \text{XXXX} \quad \text{XXXX} \quad \text{XXXX}_x, XX10 \quad 0000 \quad 0000 \quad 0000_x \tag{10}$$

In round 5 we have 8 active s-boxes. The resulting probability is $(2^{-7})^8 = 2^{-56}$. It is assumed that this probability holds for all following rounds. Note that the maximum differential probability for addition is assumed to be 1. We then get for a 16 round characteristic:

$$\text{P=1*}2^{-9} * 2^{-9} * 2^{-14} * \prod_{i=5}^{16} 2^{-56} = 2^{-704} \tag{11}$$

A characteristic in which s-box 0 is the only active in rounds 2 and 3 can be derived likewise. It has probability $P = 2^{-704}$ The plaintext difference for this is:

$$0000 \quad 0000 \quad 0000 \quad 0000_x, 0000 \quad 0000 \quad 0000 \quad 0800_x \tag{12}$$

The probabilities for the derived characteristics do not allow a successful differential attack on Kaweichel.

6 Conclusion

The analysis in the previous sections shows, that 16 round Kaweichel is immune from linear cryptanalysis and suggest that it is immune from differential cryptanalysis. It is therefore decided to propose versions of Kaweichel with 16 rounds and 24 rounds, named Kaweichel-16 and Kaweichel-24. The first version of Kaweichel published in [2, 7] is now Kaweichel-32. The keylength for Kaweichel-16 is 256 bits to 896 bits, for Kaweichel-24 256 bits to 1408 bit and for Kaweichel-32 256 bits to 1920 bits. In the reference implementation [6] the constant NUMROUNDS has to be adapted for the new versions. The new version allow for a tradeoff between speed an security.

7 Open Problems

This analysis of Kaweichel is somewhat incomplete. While the complexity derived for the linear attack is truly a lower bound, the author thinks further research on Kaweichel is needed. Firstly it is not known to the author, if characteristics of higher probability then those derived are possible. If one bears in mind, that the round function is not surjective, it is clear that an input XOR to the round function different from 0 can cause an output XOR to be 0. This may give rise to characteristics with higher probabilty.

Another pivotal point are the differential properties of addition modulo 2^{64}. It was often assumed that the maximum differential probability is 1. However, this is only the case for a small number of differentials. In [3] it is shown, that the probability of possible differentials is:

$$\mathrm{P}[\mathrm{X} \neq 0] = \frac{1}{2}\left(\frac{7}{8}\right)^{n-1} \tag{13}$$

For n=64 this is $P[X \neq 0] \approx 2^{-13}$ The question is wether this low probability together with the low probability of the whole characteristics makes Kaweichel susceptible to attacks with impossible differentials.

8 Acknowledgements

The author thanks Claus Grupen of Siegen University for continued encouragement and support.

References

[1] Biham, Eli and Adi Shamir: *Differential Cryptanalysis of the Data Encryption Standard,* Springer Verlag, Berlin, Heidelberg, New York, 1993

[2] Grupen, Claus and Dieter Schmidt: *Beschreibung einer Blockchiffre -Kaweichel- (in German)*, available from: `http://www.infoserversecurity.org/itsec_infoserver_v0.5/sections/science/docs/1095771791/kaweichel.pdf`

[3] Lipmaa, Helger and Shiho Moriai: *Efficient Algorithms for Computing Differential Properties of Addition*, available from: http://eprint.iacr.org/2001/001.pdf

[4] Matsui, Mitsuru: Linear Cryptanalysis Method for DES Cipher, in Helleseth, Tor (Ed.): *Advances in Cryptology - EUROCRYPT '93,* Springer Verlag, Berlin, Heidelberg, New York, 1993

[5] Mukhopadhyay, Debdeep and Dipanwita RoyChowdhury: *Key mixing in Block Ciphers through Addition modulo* 2^n, available from: http://eprint.iacr.org/2005/383.pdf

[6] Schmidt, Dieter: *Reference Implementation of the Block Cipher Kaweichel in C*, available from: http://www.infoserversecurity.org/itsec_infoserver_v0.5/sections/science/docs/1095771918/kaweichel.zip

[7] Schmidt, Dieter: *Kaweichel, an Extension of Blowfish for 64-Bit Architecturs*, available from: http://eprint.iacr.org/2005/144.pdf

[8] Schneier, Bruce: Description of a New Variable-Length-Key, 64-Bit Block Cipher (Blowfish), in Anderson, Ross (Ed.): *Fast Software Encryption - Cambridge Security Workshop, Proceedings,*, Springer Verlag, Berlin, Heidelberg, New York, 1994