Analysis of SEED

March 24, 2003

Executive Summary

This report presents the results of a limited time evaluation of the block cipher SEED.

SEED is a so-called Feistel cipher where the ciphertext is computed as a function of the plaintext and the secret key in a number of rounds. The number of rounds chosen for SEED is 16. The S-boxes used in SEED are reminiscent of that used in AES, the Advanced Encryption Standard, whereas the linear transformation used to combine outputs of the S-boxes is new. SEED also mixes the use of different group operations, a technique which seems to have become popular (although the AES does not do this).

The report contains analysis of SEED against the state-of-the-art attacks on block ciphers. Our results are to a wide degree in line with the analysis done by the authors.

In summary, our analysis does not show any important flaws nor weaknesses in SEED. The best known attack at this point of time is an exhaustive search for the key.

We note that the analysis was performed without access to complete computer code implementing the block cipher.

In the Appendix we give a compressed overview of the state-of-the-art in block cipher cryptanalysis.

Contents

1 Introduction

SEED is a 128-bit block cipher with a 128-bit key. The structure is the traditional Feistel network, an iterated cipher where the ciphertext is computed as a function of the plaintext and the key in a number of rounds. SEED has 16 rounds. In each round the inputs are divided into two halves of each 64 bits. One half is input to a function F, whose output is exclusive-ored to the other half of the input whereafter the halves are swapped. The function F divides its input into two halves of each 32 bits. Each half is exclusive-ored with a 32-bit subkey, after which the right half is exclusive-ored to the left half. Then the texts enter a 3-stage mini-cipher using a function G and modular addition modulo 2^{32}.

The model for cryptanalysis in this report is the standard one for block ciphers, where it is assumed that the attacker has access to a black-box which encrypts any chosen input or decrypts any chosen output using SEED with a secret, randomly chosen key.

2 Structural features and characteristics

The two S-boxes S_1 and S_2 used in the function G in SEED are defined as follows: $S_1(x) = A_1 \cdot x^{247} \oplus b_1$ and $S_2(x) = A_2 \cdot x^{251} \oplus b_2$, where A_1 and A_2 are binary 8×8 matrices, x^d denotes exponentiation in the finite field $GF(2^8)$, "$\cdot$" denotes the product of the matrix and the input x where the latter is viewed as an 8-bit vector, and where b_1 and b_2 are constants.

It is well-known that $(x \oplus y)^{2^r} = x^{2^r} \oplus y^{2^r}$ in $GF(2^s)$ for any integer values r, s. (In fact, a more general result applies but this is not relevant for this document). The only nonlinear components in the S-boxes are the permutations x^{247} and x^{251} in $GF(2^8)$. It follows that both these functions are derived from the function $x^{-1} = x^{254}$ in $GF(2^8)$ and a linear function, since

$$x^{247} = \frac{1}{x^8} = (x^8)^{-1},$$

$$x^{251} = \frac{1}{x^4} = (x^4)^{-1},$$

and both x^8 and x^4 are linear functions. All in all, an alternative description of the S-boxes in SEED is the following: $S_1(x) = A_1 \cdot (B_1 \cdot x)^{-1} \oplus b_1$ and $S_2(x) = A_2 \cdot (B_2 \cdot x)^{-1} \oplus b_2$, where B_1 and B_2 are binary 8×8 matrices, representing x^8 respectively x^4.

The function G is defined as follows. Let the 4-byte input be denoted (a, b, c, d), then the outputs (a', b', c', d') are defined

$$a' = (S_1(a)\&m_0) \oplus (S_2(b)\&m_1) \oplus (S_1(c)\&m_2) \oplus (S_2(d)\&m_3), \quad (1)$$

$$b' = (S_1(a)\&m_1) \oplus (S_2(b)\&m_2) \oplus (S_1(c)\&m_3) \oplus (S_2(d)\&m_0), \quad (2)$$

$$c' = (S_1(a)\&m_2) \oplus (S_2(b)\&m_3) \oplus (S_1(c)\&m_0) \oplus (S_2(d)\&m_1), \quad (3)$$

$$d' = (S_1(a)\&m_3) \oplus (S_2(b)\&m_0) \oplus (S_1(c)\&m_1) \oplus (S_2(d)\&m_2), \quad (4)$$

where $(m_0, m_1, m_2, m_3) = (fc_x, f3_x, cf_x, 3f_x)$ (subscript x denotes hexadecimal notation) and "&" is the bitwise AND operation.

By a closer look at these computations it follows that the function G is a bijective mapping. As an example, given a', b', c', d' one can retrieve a as follows. The sums $a' \oplus b' \oplus c'$, $a' \oplus b' \oplus d'$, $a' \oplus c' \oplus d'$, and $b' \oplus c' \oplus d'$ contain disjoint sets of two bits of $S_1(a)$, and as a result all eight bits of $S_1(a)$ can be computed from a', b', c', d'. Since S_1 is a permutation, a can be found. In similar manners the values of b, c, and d can be found.

The function F is itself a 3-stage iterated cipher, and F is bijective only if G is bijective.

Fixed points of the S-boxes: It is argued in [2] that one reason for modifying the outputs of the power polynomials x^{247} and x^{251} by affine mappings was to remove the fixed points 0 and 1. However, although these affine mappings remove these two fixed points, it has also the effect that it introduces three other fixed points, namely 23, 230, for S-box S_1 and 28 for S-box S_2.

3 Alternative, equivalent specification

In this section we give an alternative specification of SEED. This is not in itself a weakness, but for some attacks it might be advantageous to use the alternative algorithm.

Define $S(x, y) = (x \oplus y, y)$, $G_L(x, y) = (G(x), G(x) + y)$ and $G_R(x, y) = (G(y) + x, G(y))$, where '+' is modular addition modulo 2^{32}. Let f_{K_1,K_2} be the function of the exclusive-or of the two subkeys in the round function F.

Then the round function F can be described as follows

$$G_L \circ G_R \circ G_L \circ S \circ f_{K_1,K_2}(x, y).$$

But also the following alternative description exists. Apply S to both 64-bit halves of the plaintext, then perform 16 rounds of encryption in a Feistel network using the function

$$S \circ G_L \circ G_R \circ G_L \circ f_{K_1 \oplus K_2,K_2}(x, y).$$

Finally, apply S to both 64-bit halves of the outputs of the last round to obtain the ciphertext. In other words, it is possible to use the function S at the end of the function F instead of in the beginning with a small modification to the key-schedule and with a simple initial and simple final transformation of the plaintext and ciphertext.

4 Key-schedule

The key-schedule of SEED takes a 128-bit user-selected key and returns 16 pairs of round keys each of two times 32 bits, in total 1024 bits.

Let the round keys in round i be denoted $k_{i,0}$ and $k_{i,1}$. They are generated as follows. The user selected key is divided into four pieces, a, b, c, d, each of 32 bits.

- for i:=1 to 16 do
  - $k_{i,0} = G(a + c - kc_i)$
  - $k_{i,1} = G(b - d + kc_i)$
  - if i odd do $b||a = (b||a)^{>>8}$
    else do $d||c = (d||c)^{<<8}$,

where kc_i for $i = 1, \ldots, 16$ are round constants derived in a pseudo random fashion. We refer to [1] for the notation used.

At a first glance it appears that there are no related keys for SEED because of the use of the highly nonlinear function G in the key-schedule. However, there are keys whose subkeys are related. Notice that the generation of $k_{i,0}$ depends only on the rotated versions of a and c together with the round constant. Thus if it holds for two user-selected keys K and K^* that $a+c$ is always a constant, then the subkeys $k_{i,0}$ for K and the subkeys $k^*_{i,0}$ for K^* are equal, i.e., $k_{i,0} = k^*_{i,0}$ for $i = 1, \ldots, 16$. If $a + c$ is to be constant for all values of i, then it must hold that $(b||a) = (b||a)^{<<8} = (d||c) = (d||c)^{<<8}$, which means that if $a = a_0, a_1, a_2, a_3$, where a_i are byte valued, and similar for b, c, and d, then it must hold that for some constant e that

- $a_i + c_j = e$ for all $i, j = 0, 1, 2, 3$,
- $a_i + d_j = e$ for all $i, j = 0, 1, 2, 3$,
- $b_i + c_j = e$ for all $i, j = 0, 1, 2, 3$,
- $b_i + d_j = e$ for all $i, j = 0, 1, 2, 3$,

Summing up, let the user-selected key be divided into the 32-bit words a, b, c, d. Consider the keys obtained from having $a_0 = a_1 = a_2 = a_3 = x$ for some value x, $b = a$, $c_0 = c_1 = c_2 = c_3 = y$, for some value y, and $d = c$. Then it holds that the keys for which $x + y$ is a constant will produce the same values for $k_{i,0}$ for $i = 1, \ldots, 16$. Thus, there are 2^{16} keys which can be divided into 256 classes of each 256 keys, such that in one class all 256 keys produce identical values of the subkeys $k_{i,0}$ for $i = 1, \ldots, 16$. Table 1 lists three keys and the subkeys they generate.

It follows by very similar observations that there are 2^{16} keys which can be divided into 256 classes of each 256 keys, such that in one class all 256 keys produce identical values of the subkeys $k_{i,1}$ for $i = 1, \ldots, 16$. Table 2 lists three (other) keys and the subkeys they generate.

Despite of these findings the key-schedule of SEED does not seem to allow for related-key attacks. First of all, the "related keys" reported above are few, second, the relations between them does not seem to be strong enough to allow for these kinds of attacks.

The above phenomena are not a threat for SEED when used for encryption. If a key is chosen uniformly at random, the probability to pick one of the above keys is very small. Moreover it is not clear for which applications an attacker

Key =	9b9b9b9b	9b9b9b9b	11111111	11111111
Round key no.				
1	4124db1d	3451bd29		
2	9a0f9a3a	4b127456		
3	79efee8e	273d39c9		
4	57215006	b12689b3		
5	03c24bbc	5f7092c7		
6	c0a53c4c	2b831b79		
7	cf3ebb62	d29fac9a		
8	2a14ef6c	a2c6cfe2		
9	7b85aa09	07894284		
10	f527f311	9100f2f9		
11	4ee60e85	14546a91		
12	26d5c935	864101db		
13	803e5e92	34e0e2c0		
14	c91d482b	2b10ede5		
15	0788fd30	2d60d71e		
16	f92d78ce	2bd7ef41		
Key =	3a3a3a3a	3a3a3a3a	72727272	72727272
Round key no.				
1	4124db1d	e0ef1874		
2	9a0f9a3a	711b066c		
3	79efee8e	5c178ff9		
4	57215006	0b809197		
5	03c24bbc	26afe9b0		
6	c0a53c4c	3c1b8a18		
7	cf3ebb62	573ddbb6		
8	2a14ef6c	c0be0d10		
9	7b85aa09	75080ba7		
10	f527f311	56ab375e		
11	4ee60e85	39e99972		
12	26d5c935	1591baad		
13	803e5e92	0ffc828b		
14	c91d482b	2d9680fc		
15	0788fd30	8e5a5bd0		
16	f92d78ce	5e235141		
Key =	2a2a2a2a	2a2a2a2a	82828282	82828282
Round key no.				
1	4124db1d	07460ff4		
2	9a0f9a3a	b82298f4		
3	79efee8e	7ee3b13e		
4	57215006	46c3d6b0		
5	03c24bbc	4af65578		
6	c0a53c4c	1bb446d4		
7	cf3ebb62	0b5a1d9e		
8	2a14ef6c	bfaa5324		
9	7b85aa09	4c16e012		
10	f527f311	1d68f56f		
11	4ee60e85	7def7131		
12	26d5c935	52eff20b		
13	803e5e92	3c3c924e		
14	c91d482b	e02f858f		
15	0788fd30	74fd6be4		
16	f92d78ce	a9ccd586		

Table 1: Examples of keys which produce equal first subkeys in every round.

Key =	9b9b9b9b	9b9b9b9b	efefefef	efefefef
Round key no.				
1	68c9edf1	7d28cbaf		
2	7e8e4d27	f9c76fad		
3	aa37e9ee	f59dd258		
4	5da694ad	7605924a		
5	c61b186a	b3c83014		
6	45dc4ae5	bfb0fcbe		
7	05ce5df3	fd6a1882		
8	9ab323b3	6ef967c7		
9	a08e3ccc	d883dcd7		
10	8c92b184	13ddd10c		
11	77553f19	af7cecc4		
12	24e69b24	e007b43e		
13	bca52806	5f7651a0		
14	dd2474e9	1e09a2f2		
15	0eeecd5b	9c28a623		
16	3685e91e	bcad5740		

Key =	3a3a3a3a	3a3a3a3a	8e8e8e8e	8e8e8e8e
Round key no.				
1	0d92c044	7d28cbaf		
2	de60205a	f9c76fad		
3	d9258549	f59dd258		
4	9d84df1f	7605924a		
5	ea0a79a8	b3c83014		
6	638fd5fa	bfb0fcbe		
7	470a077c	fd6a1882		
8	c252c5d8	6ef967c7		
9	a6b5f762	d883dcd7		
10	a55b43b7	13ddd10c		
11	ca3a056e	af7cecc4		
12	a678af9c	e007b43e		
13	4aa21758	5f7651a0		
14	a0ab171a	1e09a2f2		
15	1d432710	9c28a623		
16	ad80bb01	bcad5740		

Key =	2a2a2a2a	2a2a2a2a	7e7e7e7e	7e7e7e7e
Round key no.				
1	f23c7655	7d28cbaf		
2	0b5f9dbd	f9c76fad		
3	656eb6da	f59dd258		
4	886b8015	7605924a		
5	caac1ba9	b3c83014		
6	54d62348	bfb0fcbe		
7	a9bdeb44	fd6a1882		
8	8bb07ddf	6ef967c7		
9	661831f3	d883dcd7		
10	05090fea	13ddd10c		
11	60f094cc	af7cecc4		
12	3393a0f5	e007b43e		
13	770ab190	5f7651a0		
14	10702afd	1e09a2f2		
15	0ef8e298	9c28a623		
16	7c8e917d	bcad5740		

Table 2: Examples of keys which produce equal second subkeys in every round.

would be able to exploit the use of such keys. On the other hand, such similarities between the subkeys of different keys do not appear in other modern block ciphers and could probably have been prevented.

5 Differential and linear cryptanalysis

5.1 Differential cryptanalysis

In differential cryptanalysis one talks about an active S-box if the inputs to the S-box are different. The branch number [9] of a function consisting of several S-boxes is defined to be the least number of active S-boxes in any inputs and in the corresponding outputs for the particular function. The branch number of the function G in SEED is four. This is an easy exercise to show. If the inputs to G have only one active S-box then the outputs have four active S-boxes, in total five active S-boxes. If the inputs to G have two active S-boxes then the outputs have at least two active S-boxes, in total at least four active S-boxes in the inputs and outputs, which is an effect of the used masks, m_0, m_1, m_2, m_3. If the inputs to G have three active S-boxes then the outputs have at least one active S-boxes, in total at least four active S-boxes, which follows from the fact that G is a bijective mapping. So, clearly the branch number is four.

Next we consider the function F which has three applications of the bijective function G. It follows by a closer look that the number of active G-functions in F is at least two. In other words, for any pair of different inputs to the function F there will be different inputs to at least two of the three G functions.

Let us first look at a simplified version of SEED, where the modular additions are replaced by exclusive-ors, as was done in [1]. Let us denote by F' the resulting modified function of F. From the branch number of the function G it follows that the number of active S-boxes in total for the function F' is always at least four.

Since the function F' is bijective for a r-round characteristic for SEED there will be different inputs to at least $\lfloor r/3 \rfloor \cdot 2$ F'-functions. SEED has 16 rounds and previous experience show that in a successful differential attack one needs to be able to specify a characteristic over at least 13 rounds (but possible up to 15 in case of SEED). This means that in at least eight of the rounds the inputs to F are different. As a consequence, it is estimated that in a successful differential attack the characteristics involved will give rise to 32 active S-boxes. The best characteristic over one S-box in SEED has a probability of 2^{-6}, which means that a characteristic with 32 active S-boxes will have a probability of 2^{-192}. This is certainly low enough to conclude that a differential attack on the modified version of SEED based on characteristics, as outlined here, is very unlikely to be possible.

However, from this analysis of SEED using the modified function F' one cannot conclude directly a similar result for unmodified SEED. The reason for this is the mixed use of the group operations, modular addition and exclusive-or. In the above differential analysis of modified SEED we used the exclusive-or of two

inputs as the definition of a difference of two inputs. However, when two halves in F are combined with the modular addition operation, the number of active S-boxes can change and in fact decrease. This has the effect that the number of active S-boxes in total for the function F can be less than the above stated 4. We illustrate this with an example which has been found in experiments. Let the (exclusive-or) differences in the two inputs to the function F be $000011d4_x$ and $000011d4_x$, that is, there are equal differences in the two halves. Clearly, the differences are unchanged after the exclusive-or of the subkeys. The next operation is to exclusive-or the right half of the input to the left half. As a consequence the inputs to the first application of G are equal as are the outputs, and the number of active S-boxes are zero. In the following operation the left half is added (mod 2^{32}) to the right half. With some high probability this means that there are two active S-boxes in the inputs to the second application of G. As explained above (and in [1]) there is a positive probability that in the outputs of G there are again two active S-boxes. The next operation is to add the right half to the left half. In the example, the left halves (in the two texts considered) are equal at this point, and with a positive probability the modular addition can have the effect that there is only one active S-box in the inputs to the third application of G. In the example, the differences in the two halves at this point were $0000ff00_x$ and 00010100_x. It follows that in total for this differential there were only three active S-boxes in the G functions. Therefore one cannot directly take the analysis of SEED using the modified function F' and transfer them to SEED with the unmodified function F.

A complete analysis of F and SEED with respect to differential cryptanalysis is very complex and involved and it is unlikely that one can ever conclude having found the best differentials for SEED. However, it seems highly unlikely that one can find differentials with only 3 active S-boxes per round for an r-round differential for large values of r. An estimate that at least three active S-boxes are needed in two of every three rounds of a differential still gives a low, best differential probability. As an example, in a 13-round differential the inputs to at least eight F functions will be different, which using the new bound will give rise to at least 24 active S-boxes. The best characteristic over one S-box in SEED has a probability of 2^{-6}, which means that a differential over 13 rounds will have a probability of at most 2^{-144}, which is low enough to conclude that a differential attack is unlikely to be found on SEED with the current state-of-the-art.

5.2 Linear cryptanalysis

As for differential cryptanalysis one speaks about active S-boxes in the linear characteristics considered. In this case, an S-box is called active if one or more bits in the linear approximations are used as input or output of the particular S-box. For the function F a linear approximation must involve bits in the inputs and outputs of at least two of the three G functions, as also stated in [1].

It is possible to build linear approximations across one G-function with only one active S-box. For example, above it is shown that the sum of three output

bytes of G give two bits of one of the bytes output from an S-box. However, it appears to be very difficult to build good linear approximations for the function F and for several rounds of SEED using approximations with only one active S-box per active G function. The best approach seems to be that of [1] which involves two active S-boxes per G function involved in the approximation. It is possible at this point to give an upper bound to the best linear approximations for r rounds of SEED using arguments similar to the ones in the differential analysis above.

Let us first consider a version of SEED where the modular additions are replaced by exclusive-or operations. A linear approximation across r rounds will involve the inputs to at least $\lfloor r/3 \rfloor \cdot 2$ F'-functions. Previous experience show, as in differential cryptanalysis, that for a 16-round Feistel cipher a successful linear attacker needs to be able to specify a characteristic over at least 13 rounds. This means that in at least eight of the rounds the inputs to F are involved. In each of these rounds it is safe to estimate that there will be at least four active S-boxes. Since the linear probability, see e.g., [1] for a definition, for one S-box is at most 2^{-6}, one gets an estimate for 13 rounds of 2^{-192}. These estimates were made on a simplified version of SEED without incorporating the effect of the modular additions. The mixed use of exclusive-or operations and the modular additions is likely to complicate a linear attack even further. Where this mixed use of operations could potentially help an attack in a differential attack it seems it will only worsen things for an attacker in a linear attack. Modular additions introduce carry bits which will lower the probabilities of the involved characteristics. We believe that it is safe to conclude that it is very unlikely that SEED is vulnerable to a linear attack.

6 Other cryptanalysis

There are trivial attacks on all block ciphers. An exhaustive search for the secret key of SEED can be done using only a few known plaintexts and their corresponding ciphertexts in time about 2^{128}. Also, the "matching ciphertext attack" applies in the Electronic Code Book and Cipher Block Chaining modes and requires about 2^{64} ciphertext blocks to succeed with good probability and which enables the attacker to deduce information about the plaintext blocks.

Higher order differentials. This attack applies to ciphers which uses nonlinear components of a low algebraic degree. The S-boxes of SEED have an algebraic degree of seven, which is the maximum degree for a bijective S-box on 8 bits. There are three layers of S-boxes in every F-function of one round and a total of 16 rounds. Therefore since an attack based on dth order differentials requires a collection of 2^d texts, it is very likely that the algebraic degree of the ciphertexts as a function of plaintexts is high enough to prevent a higher order differential attack from being practical, if possible at all.

The integral attacks apply to only a few rounds of SEED. It is conjectured that reduced versions of SEED with up to six rounds is vulnerable to these attacks but not versions with more than six rounds.

The interpolation attacks apply to ciphers which use simple mathematical functions only. The S-boxes in SEED are constructed from the inverse function in a finite field, which has a simple description. However, the fact that both the inputs and outputs are mixed with affine mappings makes the description much more complex. This together with the mixed use of exclusive-ors and modular additions make the interpolation attacks very unlikely to be applicable.

The slide attacks apply best to ciphers with very simple key-schedules. However, the key-schedule of SEED uses both the S-boxes and different round constants, which are good means to prevent these attacks from being effective.

The non-surjective attacks are not applicable as the round function is bijective and thus not vulnerable to these attacks.

7 Previous results on SEED

As far as we are informed the only public analysis of SEED is that of the authors themselves [1]. The analysis in this paper is much in line with this analysis by the authors. However, we reported on a few things which the authors do not seem to have noticed.

First of all, the S-boxes are both derived from the inverse function in a Galois field. This makes the S-boxes similar to those of the AES. For the AES several reports have indicated that the use of linear transformations of the inverse in a Galois field as the only nonlinear component in a block cipher might be dangerous. However, one important difference between SEED and AES in this respect is that the former uses a mix of group operations, the exclusive-or and additions modulo 2^{32}. Therefore, the concerns for the AES that it might be possible to express the ciphertexts as a simple mathematical expression of the plaintext and the key do not apply in the same degree for SEED.

For the key-schedule it was shown in [1] that related-key attacks are not applicable and that no weak or semi-weak keys exist. In our report we noted that there are keys which produce sets of subkeys with many common elements. This feature was not exploited in any attacks and might not appear to be serious for SEED, however they do illustrate properties for keys which are not usually seen in modern block ciphers.

In [1] it was concluded that SEED is not vulnerable to a differential attack. To facilitate an easier analysis the authors first consider a simplified version of SEED. For this version a search was made for the best differential characteristics and subsequently these characteristics were considered for (the unmodified) SEED. However we demonstrated that this approach might not always lead to the best characteristics. There are cases where the elements in the unmodified SEED not present in the simplified SEED have an important influence on the existence of certain differential characteristics. However a full differential analysis of unmodified SEED is a very difficult, if not impossible, task. Our estimates of bounds on the success of differential attacks are more crude than those of [1] but still with a sufficiently wide security margin.

The previous statement applies also linear attacks. The analysis in [1] is

more detailed that our analysis, but our estimates more crude. Both analyses conclude that SEED does not seem vulnerable to a linear attack.

Finally, one reason for the linear transformations applied to the power polynomials in the S-boxes of SEED was to remove the two fixed points 0 and 1. However, even though the chosen linear transformations remove these fixed points, they introduce three other fixed points. It is not clear why the authors are more concerned about the fixed points 0 and 1 than about other fixed points.

A Block Ciphers in General

In the following we give a compressed overview of the state-of-the-art of block cipher cryptanalysis, and outline the following known attacks.

1. Exhaustive Key Search
2. Matching Ciphertext Attacks
3. Differential Cryptanalysis
4. Truncated Differential Attacks
5. Higher-order Differential Attacks
6. Linear Cryptanalysis
7. Related-key Attacks
8. Non-surjective Attacks
9. Interpolation Attacks
10. Mod-n Attacks
11. Slide Attacks
12. Integral Attacks

A.1 Exhaustive key search

This attack needs only a few known plaintext-ciphertext pairs. An attacker simply tries all keys, one by one, and checks whether the given plaintext encrypts to the given ciphertext. For a block cipher with a k-bit key and n-bit blocks the number of pairs of texts needed to determine the key uniquely is approximately $\lceil k/n \rceil$. Also, if the plaintext space is redundant, e.g., consists of English or Japanese text, the attack will work if only some ciphertext blocks is available. The number of ciphertext blocks needed depends on the redundancy of the language.

A.2 The matching ciphertext attack

The *matching ciphertext attack* is based on the fact that for block ciphers of m bits used in the modes of operations for the DES [26] after the encryption of $2^{m/2}$ blocks, equal ciphertext blocks can be expected and information is leaked about the plaintexts [7, 15, 24].

A.3 Differential cryptanalysis

The most well-known and general method of analysing conventional cryptosystems today is *differential cryptanalysis*, published by Biham and Shamir in 1990. Differential cryptanalysis is universal in the sense that it can be used against any cryptographic mapping which is constructed from iterating a fixed round function. One defines a **difference** between two bit strings, X and X' of equal length as

$$\Delta X = X \otimes (X')^{-1}, \tag{5}$$

where $\otimes$ is the group operation on the group of bit strings used to combine the key with the text input in the round function and where $(X)^{-1}$ is the inverse element of X with respect to $\otimes$. The idea behind this is, that the differences between the texts before and after the key is combined are equal, i.e., the difference is independent of the key. To see this, note that

$$(X \otimes K) \otimes (X' \otimes K)^{-1} = X \otimes K \otimes K^{-1} \otimes X'^{-1} = X \otimes (X')^{-1} = \Delta X.$$

In a differential attack one exploits that for certain input differences the distribution of output differences of the non-linear components is non-uniform.

Definition 1 *An s-round* characteristic *is a series of differences defined as an $s+1$-tuple $\{\alpha_0, \alpha_1, \ldots, \alpha_s\}$, where $\Delta P = \alpha_0$, $\Delta C_i = \alpha_i$ for $1 \leq i \leq s$.*

Here ΔP is the difference in the plaintexts and ΔC_i is the difference in the ciphertexts after i rounds of encryption. Thus, the characteristics are lists of expected differences in the intermediate ciphertexts for an encryption of a pair of plaintexts. In essence one specifies a characteristic for a number of rounds and searches for the correct key in the remaining few rounds. In some attacks it is not necessary to predict the values $\alpha_1, \ldots, \alpha_{s-1}$ in a characteristic. The pair (α_0, α_s) is called a *differential*. The complexity of a differential attack is approximately the inverse of the probability of the characteristic or differential used in the attack.

A.4 Truncated differentials

For some ciphers it is possible and advantageous to predict only the values of parts of the differences after each round of the cipher. The notion of truncated differentials was introduced by Knudsen [17]:

Definition 2 *A differential that predicts only parts of an n-bit value is called a* truncated differential. *More formally, let (a, b) be an i-round differential. If a' is a subsequence of a and b' is a subsequence of b, then (a', b') is called an i-round truncated differential.*

A truncated differential can be seen as a collection of differentials. As an example, consider an n-bit block cipher and the truncated differential (a', b), where a' specifies the least $n' < n$ significant bits of the plaintext difference and b specifies the ciphertext difference of length n. This differential is a collection of all $2^{n-n'}$ differentials (a, b), where a is any value, which truncated to the n' least significant bits is a'.

A.5 Impossible differentials

A special type of differentials are those of probability zero. The attack was first applied to the cipher DEAL [18] and later to Skipjack [4]. The main idea is to specify a differential of probability zero over some number of rounds in the attacked cipher. Then by guessing some keys in the rounds not covered by the differential one can discard a wrong value of the key if it would enable the cipher to take on the differences given in the differential.

A.6 Higher-order differentials

An sth-order differential is defined recursively as a (conventional) differential of the function specifying an $(s-1)$st order differential. In order words, an sth order differential consists of a collection of 2^s texts of certain pairwise, predetermined differences. We refer to [20, 17] for a more precise definition of higher order differentials.

In most cases one considers differences induced by the exclusive-or operation and the field of characteristic 2. The *nonlinear order* of a function $f : GF(2^n) \to GF(2^n)$ is defined as follows. Let the output bits y_j be expressed as multivariate polynomials $q_j(x) \in GF(2)[x_1, \ldots, x_n]$, where $x_1, \ldots, x_n$ are the input bits. The nonlinear order of f is then defined to be the minimum total degree of any linear combination of these polynomials. The higher order differential attacks exploit the following result.

Corollary 1 *Let $f : GF(2^n) \to GF(2^n)$ be a function of nonlinear order d. Then any dth order differential is a constant. Consequently, any $(d+1)$st order differential is zero.*

The boomerang attack [28] can be seen as a special type of a second-order differential attack. This variant applies particularly well to ciphers for which one particular (first-order) differential applies well to one half of the cipher, and where another particular (first-order) differential applies well to the other half of the cipher.

A.7 Linear cryptanalysis

Linear cryptanalysis was proposed by Matsui in 1993 [21]. A preliminary version of the attack on FEAL was described in 1992 [23]. Linear cryptanalysis [21] is a known plaintext attack in which the attacker exploits linear approximations of some bits of the plaintext, some bits of the ciphertext and some bits of the secret key. In the attack on the DES (or on DES-like iterated ciphers) the linear approximations are obtained by combining approximations for each round under the assumption of independent round keys. The attacker hopes in this way to find an expression

$$(P \cdot \alpha) \oplus (C \cdot \beta) = (K \cdot \gamma) \tag{6}$$

which holds with probability $p_L \neq \frac{1}{2}$ over all keys [21], such that $|p_L - \frac{1}{2}|$, called the bias, is maximal. In (6) $P, C, \alpha, \beta, \gamma$ are m-bit strings and '$\cdot$' denotes the dot product. The bit strings α, β, γ are called *masks*.

Definition 3 *An s-round* linear characteristic *is a series of masks defined as an $(s+1)$-tuple $\{\alpha_0, \alpha_1, \ldots, \alpha_s\}$, where α_0 is the mask of the plaintexts and α_i is the mask of the ciphertexts after i rounds of encryption for $1 \leq i \leq s$.*

As for differential cryptanalysis one specifies a linear characteristics for a number of rounds and searches for the keys in the remaining rounds, we refer to [21] for more details. A linear attack needs approximately about b^{-2} known plaintexts to succeed, where b is the bias of the linear characteristic used.

Also, the concepts of linear hulls, the analogue to differentials as opposed to characteristics in differentials cryptanalysis, has been defined in [25].

Finally, in [22] it has been shown that if one defines the quantity $q = (2p-1)^2$ where p is the probability of a linear characteristic or hull, then when combining several linear characteristics one can multiply their q values to get the q-value of the combination. Sometimes the q values are referred to as the "linear probability", which is somewhat misleading, but nevertheless seems to be widely used.

A.8 Mod n cryptanalysis

In [13] a generalisation of the linear attacks is considered. This attack is applicable to ciphers for which some words (in some intermediate ciphertext) are biased modulo n, where n typically is a small integer. It has been shown that ciphers which uses only bitwise rotations and additions modulo 2^{32} are vulnerable to these kinds of attacks.

A.9 Related-key attacks

There are several variants of this attack depending on how powerful the attacker is assumed to be.

1. Attacker gets encryptions under one key.
2. Attacker gets encryptions under several keys.

(a) Known relation between keys.

(b) Chosen relation between keys.

Knudsen used the methods of 1 by giving a chosen plaintext attack of the first kind on LOKI'91 [14], reducing an exhaustive key search by almost a factor of four. The concept "related-key attack" was introduced by Biham [3], who also introduced the attack scenarios of 2, where the encryptions under several keys are requested. Knudsen later described a related key attack on SAFER K [16] and Kelsey, Schneier, and Wagner [12] applied the related key attacks to a wide range of block ciphers. It may be argued that the attacks with a chosen relation between the keys are unrealistic. The attacker need to get encryptions under several keys, in some attacks even with chosen plaintexts. However there exist realistic settings, in which an attacker may succeed to obtain such encryptions. Also, there exists quite efficient methods to preclude the related key attacks [12, 10].

A.10 Interpolation attack

In [11] Jakobsen and Knudsen introduced the interpolation attack on block ciphers. The attack is based on the following well-known formula. Let R be a field. Given $2n$ elements $x_1, \ldots, x_n$, $y_1, \ldots, y_n \in R$, where the x_is are distinct. Define

$$f(x) = \sum_{i=1}^{n} y_i \prod_{1 \le j \le n, j \ne i} \frac{x - x_j}{x_i - x_j}. \tag{7}$$

$f(x)$ is the only polynomial over R of degree at most $n-1$ such that $f(x_i) = y_i$ for $i = 1, \ldots, n$. Equation (7) is known as the *Lagrange interpolation formula* (see e.g.,[6, page 185]). In the *interpolation attack* an attacker constructs polynomials using pairs of plaintexts and ciphertexts. This is particularly easy if the components in the cipher can be expressed as easily described mathematical functions. The idea of the attack is, that if the constructed polynomials have a small degree, only few plaintexts and their corresponding ciphertexts are necessary to solve for the (key-dependent) coefficients of the polynomial, e.g., using Lagrange's interpolation. To recover key bits one expresses the ciphertext before the last round as a polynomial of the plaintext.

A.11 Non-surjective attack

In [27] Rijmen-Preneel-De Win described the non-surjective attack on iterated ciphers. It is applicable to Feistel ciphers where the round function is not surjective and therefore statistical attacks become possible. In a Feistel cipher one can compute the exclusive-or of all outputs of the round functions from the plaintexts and the corresponding ciphertexts. Thus, if the round functions are not surjective this gives information about intermediate values in the encryptions, which can be used to get information about the secret keys.

A.12 Slide attacks

In [5] the "slide attacks" were introduced, based on earlier work in [3, 14]. In particular it was shown that iterated ciphers with identical round functions, that is, equal structures plus equal subkeys in the rounds, are susceptible to slide attacks. Let $F_r \circ F_{r-1} \circ \cdots \circ F_1$ denote an r-round iterated cipher, where all F_is are identical. The attacker tries to find pairs of plaintext P, P^* and their corresponding ciphertexts C, C^*, such that $F_1(P) = P^*$ and $F_r(C) = C^*$. Subsequently, an attacker has twice both the inputs and outputs of one round of the cipher. If the round function is simple enough, this can lead to very efficient attacks. To find such pairs of texts, one can in the worst case apply the birthday paradox, such that one such pair is expected from a collection of $2^{n/2}$ texts, where n is the block size.

A.13 Integral Attacks

These attacks are sometimes referred to as the "Square attack", since it was first applied to the block cipher Square [8]. The attack on Square slightly modified also applies to the block ciphers Crypton and Rijndael [9].

In [19] these attacks are generalised under the name of "integral cryptanalysis". In differential attacks one considers differences of texts, in integral cryptanalysis one considers sums of texts. In ciphers where all nonlinear functions are bijective, it is sometimes possible to predict a sum of texts, even in the cases where differential attacks are not applicable. The main observations are that in a collection of texts which in a particular word take all values exactly equally many times, the value of the words after a bijective function also take all values exactly equally many times. Also, assume that s words have this property and that in the cipher a linear combination of the s words are computed (with respect to the group operation considered). Then it is possible to determine also the sum of all linear combinations in a collection of texts. This attack is still today the best attack reported on Rijndael which has been the selected for the Advanced Encryption Standard.

References

[1] ANNEX. The analyses of SEED. Document.

[2] KOREAN CONTRIBUTION ON NP 18033 "ENCRYPTION ALGORITHM". Document.

[3] E. Biham. New types of cryptanalytic attacks using related keys. In T. Helleseth, editor, *Advances in Cryptology: EUROCRYPT'93, LNCS 765*, pages 398–409. Springer Verlag, 1993.

[4] E. Biham, A. Biryukov, and A. Shamir. "Impossible" cryptanalysis. Presented at the rump session of CRYPTO'98.

[5] A. Biryukov and D. Wagner. Slide attacks. In L. R. Knudsen, editor, *Fast Software Encryption, Sixth International Workshop, Rome, Italy, March 1999, LNCS 1636*, pages 245–259. Springer Verlag, 1999.

[6] P.M. Cohn. *Algebra, Volume 1.* John Wiley & Sons, 1982.

[7] D. Coppersmith, D.B. Johnson, and S.M. Matyas. Triple DES cipher block chaining with output feedback masking. Technical Report RC 20591, IBM, October 1996. Presented at the rump session of CRYPTO'96.

[8] J. Daemen, L. Knudsen, and V. Rijmen. The block cipher Square. In E. Biham, editor, *Fast Software Encryption, Fourth International Workshop, Haifa, Israel, January 1997, LNCS 1267*, pages 149–165. Springer Verlag, 1997.

[9] J. Daemen and V. Rijmen. AES proposal: Rijndael. Submitted as an AES Candidate Algorithm. Description available from NIST, see `http://www.nist.gov/aes`.

[10] I.B. Damgård and L.R. Knudsen. Two-key triple encryption. *The Journal of Cryptology*, 11(3):209–218, 1998.

[11] T. Jakobsen and L. Knudsen. The interpolation attack on block ciphers. In E. Biham, editor, *Fast Software Encryption, Fourth International Workshop, Haifa, Israel, January 1997, LNCS 1267*, pages 28–40. Springer Verlag, 1997.

[12] J. Kelsey, B. Schneier, and D. Wagner. Key-schedule cryptanalysis of IDEA, G-DES, GOST, SAFER, and triple-DES. In Neal Koblitz, editor, *Advances in Cryptology: CRYPTO'96, LNCS 1109*, pages 237–251. Springer Verlag, 1996.

[13] J. Kelsey, B. Schneier, and D. Wagner. Mod n cryptanalysis, with applications against RC5P and M6. In L. Knudsen, editor, *Fast Software Encryption, Sixth International Workshop, Rome, Italy, March 1999, LNCS 1636*, pages 139–155. Springer Verlag, 1999.

[14] L.R. Knudsen. Cryptanalysis of LOKI'91. In J. Seberry and Y. Zheng, editors, *Advances in Cryptology, AusCrypt 92, LNCS 718*, pages 196–208. Springer Verlag, 1993.

[15] L.R. Knudsen. *Block Ciphers – Analysis, Design and Applications.* PhD thesis, Aarhus University, Denmark, 1994.

[16] L.R. Knudsen. A key-schedule weakness in SAFER K-64. In Don Coppersmith, editor, *Advances in Cryptology - CRYPTO'95, LNCS 963*, pages 274–286. Springer Verlag, 1995.

[17] L.R. Knudsen. Truncated and higher order differentials. In B. Preneel, editor, *Fast Software Encryption - Second International Workshop, Leuven, Belgium, LNCS 1008*, pages 196–211. Springer Verlag, 1995.

[18] L.R. Knudsen. DEAL - a 128-bit block cipher. Technical Report 151, Department of Informatics,University of Bergen, Norway, February 1998. Submitted as an AES candidate by Richard Outerbridge.

[19] L.R. Knudsen and D. Wagner. Integral cryptanalysis. In *FSE 2002*.

[20] X. Lai. Higher order derivatives and differential cryptanalysis. In R. Blahut, editor, *Communication and Cryptography, Two Sides of One Tapestry*. Kluwer Academic Publishers, 1994. ISBN 0-7923-9469-0.

[21] M. Matsui. Linear cryptanalysis method for DES cipher. In T. Helleseth, editor, *Advances in Cryptology - EUROCRYPT'93, LNCS 765*, pages 386–397. Springer Verlag, 1993.

[22] M. Matsui. New structure of block ciphers with provable security against differential and linear cryptanalysis. In D. Gollman, editor, *Fast Software Encryption, Third International Workshop, Cambridge, UK, February 1996, LNCS 1039*, pages 205–218. Springer Verlag, 1996.

[23] M. Matsui and A. Yamagishi. A new method for known plaintext attack of FEAL cipher. In R. Rueppel, editor, *Advances in Cryptology - EUROCRYPT'92, LNCS 658*, pages 81–91. Springer Verlag, 1992.

[24] U.M. Maurer. New approaches to the design of self-synchronizing stream ciphers. In D.W. Davies, editor, *Advances in Cryptology - EUROCRYPT'91, LNCS 547*, pages 458–471. Springer Verlag, 1991.

[25] K. Nyberg. Linear approximations of block ciphers. In A. De Santis, editor, *Advances in Cryptology - EUROCRYPT'94, LNCS 950*, pages 439–444. Springer Verlag, 1995.

[26] National Bureau of Standards. DES modes of operation. Federal Information Processing Standard (FIPS), Publication 81, National Bureau of Standards, U.S. Department of Commerce, Washington D.C., December 1980.

[27] V. Rijmen, B. Preneel, and E. De Win. On weaknesses of non-surjective round functions. *Designs, Codes, and Cryptography*, 12(3):253–266, 1997.

[28] D. Wagner. The boomerang attack. In L. R. Knudsen, editor, *Fast Software Encryption, Sixth International Workshop, Rome, Italy, March 1999, LNCS 1636*, pages 156–170. Springer Verlag, 1999.