Differential Cryptanalysis of Snefru, Khafre, REDOC-II, LOKI and Lucifer

(Extended Abstract)

Eli Biham *Adi Shamir*

The Weizmann Institute of Science
Department of Applied Mathematics and Computer Science
Rehovot 76100, Israel

Abstract

In [1,2] we introduced the notion of differential cryptanalysis based on chosen plaintext attacks. In [3,4] we described the application of differential cryptanalysis to Feal[13,12] and extended the method to known plaintext attacks. In this paper we apply differential cryptanalytic methods to the hash function Snefru[10] and to the cryptosystems Khafre[11], REDOC-II[6,7], LOKI[5] and Lucifer[8].

1 Introduction

The notion of differential cryptanalysis was introduced in [1,2,3,4]. In this paper differential cryptanalytic methods are applied to Snefru[10], Khafre[11], REDOC-II[6,7], LOKI[5] and Lucifer[8]. Due to space limitations only brief descriptions of the specific attacks can be given in this extended abstract. More extensive descriptions of the attacks will be given in the full paper.

Snefru[10] is a one way hash function suggested by Merkle as the Xerox secure hash function. In March 1990 a $1000 reward was offered to the first person to break the two-pass variant of Snefru by finding two messages which hash to the same value. A similar reward was later announced for breaking the four-pass variant of Snefru.

Khafre[11] is a fast software oriented cryptosystem suggested by Merkle. Although the number of rounds is not yet determined, the designer expects that almost all applications

Copyright (c) 1998, Springer-Verlag

will use 16, 24 or 32 rounds.

REDOC-II[6,7] is a high speed confusion/diffusion/arithmetic cryptosystem suggested by Cryptech. REDOC-II has ten rounds, but even the one-round variant is claimed to be sufficiently strong since the round-function is very complicated. A reward of $5000 was offered for the best theoretical attack performed on the one-round variant and a reward of $20000 was offered for a practical known plaintext attack on the two-round variant.

LOKI[5] is a 64-bit key/64-bit block cryptosystem similar to DES which uses one twelve-bit to eight-bit S box based on irreducible polynomials in four S box entries. Two new modes of operation which convert LOKI into a hash function are defined.

Lucifer[8] is a S-P network cryptosystem designed by IBM prior to the design of DES. The variant of Lucifer with eight rounds has 128-bit blocks and 256-bit keys.

The main results described in this paper are:

Two-pass Snefru is easily breakable within three minutes on a personal computer. Our attack can find many pairs which hash to the same value and can even find several messages hashing to the same hashed value as a given message. The attack is also applicable to three-pass and four-pass Snefru with complexities which are much better than of the birthday attack. The attack is independent of the actual choice of the S boxes and one of its variants can even be used as a black box attack in which the choice of the S boxes is not known to the attacker.

Khafre with 16 rounds is breakable by a differential cryptanalytic chosen plaintext attack using about 1500 encryptions within about an hour on a personal computer. By a differential cryptanalytic known plaintext attack it is breakable using about 2^{38} encryptions. Khafre with 24 rounds is breakable by a chosen plaintext attack using about 2^{53} encryptions and using a differential cryptanalytic known plaintext attack it is breakable using about 2^{59} encryptions.

REDOC-II with one round is breakable by a differential cryptanalytic chosen plaintext attack using about 2300 encryptions within less than a minute on a personal computer. For REDOC-II with up to four rounds it is possible to find three bytes of the masks (created by 1280 byte key tables) faster than via an exhaustive search of the key. The three masks can even be found by a known plaintext attack.

LOKI with up to eleven rounds is breakable faster than exhaustive search by differential cryptanalytic attacks, either chosen plaintext or known plaintext attacks. We further show that every key of LOKI has 15 equivalent keys due to a key complementation property and thus the complexity of a known plaintext attack on the full 16-round version can be reduced to 2^{60}. Another complementation property can reduce the complexity of a chosen plaintext attack by another factor of 16 to 2^{56}. The two hash function modes of LOKI are shown to be insecure.

Lucifer with eight rounds is breakable within 2^{21} steps using 24 ciphertexts pairs.

Copyright (c) 1998, Springer-Verlag

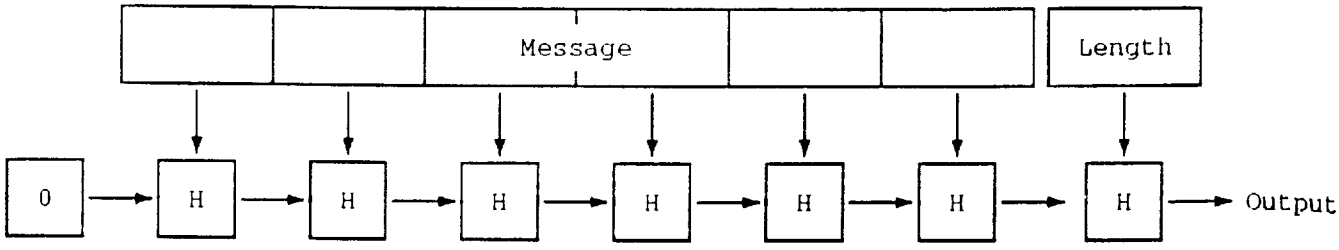


Figure 1: Outline of Snefru

2 Cryptanalysis of Snefru

Snefru[10] is designed to be a cryptographically strong hash function which hashes messages of arbitrary length into m-bit values (typically 128 bits). The messages are divided into $(512 - m)$-bit chunks and each chunk is mixed with the hashed value computed so far by a randomizing function H. The function H takes a 512-bit input and calculates an m-bit output. The new hashed value is the output of H. The outline of Snefru is given in figure 1.

The function H is based on a (reversible) 512-bit to 512-bit function E and returns a XOR combination of the first m bits of the input and the last m bits of the output of E. The function E randomizes the data in several passes. Each pass is composed of 64 randomizing rounds where in each one of them a different byte of the data is used as an input to an S box whose output word is XORed with the two neighboring words.

A cryptographically strong hash function is broken if two different messages which hash to the same value are found. In particular, we break Snefru by finding two different chunk-sized messages which hash to the same value, or in other words, finding two inputs of the function H which differ only in the chunk part and have the same output. Unless specified otherwise, we concentrate in the following discussion on two-pass Snefru with $m = 128$ (whose chunks are 384-bit long).

A universal attack on hash functions is based on the birthday paradox. If we hash about $2^{m/2}$ random messages (2^{64} when $m = 128$) then with a high probability we find a pair of messages which hash to the same value. This attack is applicable to any hash function and is independent of its details.

For Snefru we designed a differential cryptanalytic attack which is also independent of the choice of the S boxes and can even be used when the hash function is viewed as a black box which hides the choice of the S boxes themselves from the attacker.

The basic attack is as follows: choose a random chunk-sized message and prepend the 128-bit zero vector (or any other 128-bit vector with the hash value from the previous chunk) to get the input of the function H. We create a second message from the first one by modifying the two bytes in the eighth and the ninth words which are used as inputs to

Copyright (c) 1998, Springer-Verlag

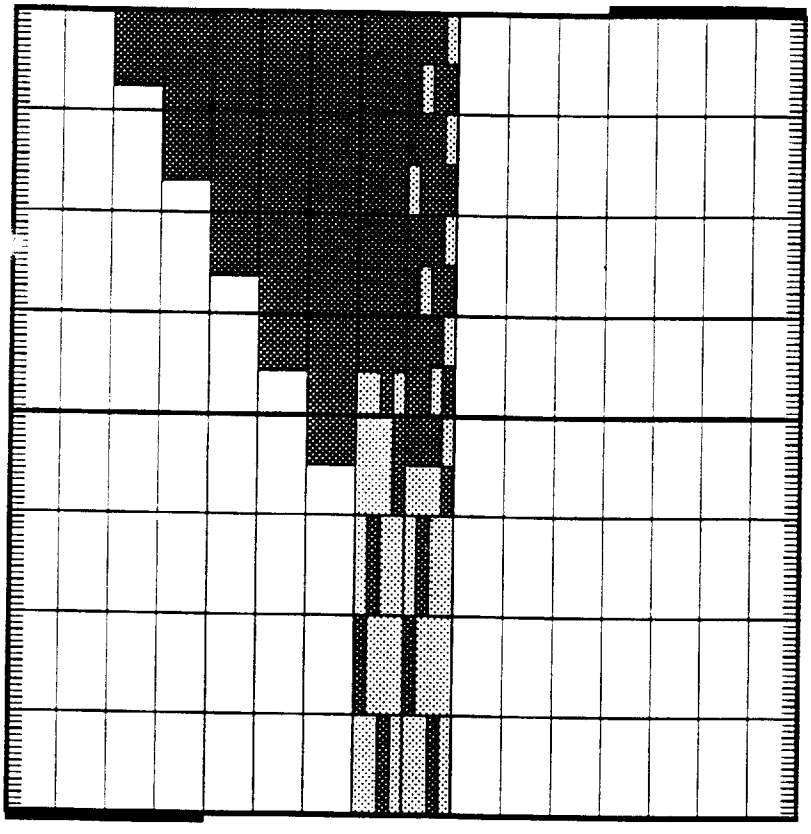


Figure 2: Graphic description of the characteristic

the S boxes at rounds 56 and 57 (the fourth use of these words). We hash both messages by the function H and compare the outputs of the two executions. A fraction of 2^{-40} of these pairs of messages are hashed to the same value. Therefore, by hashing about 2^{41} messages we can break Snefru.

In this attack we use a characteristic which differentiates only zero XOR values from non-zero XOR values and does not a priori fix the values of the non-zero XORs. In round 56 the byte from word eight is used to garble words seven and nine. In a fraction of about 1/256 of the pairs the garbled version of the byte in the ninth word cancels its chosen XOR between pairs. Therefore, for this fraction the XOR of this byte after round 56 is zero and the same values are XORed to the tenth word in both executions. The same values are used as inputs to the S boxes in both executions till the next time a byte of word seven is used at round 71. Round 71 garbles words six and eight by a different value for each execution and so does round 72 to words seven and nine. In a fraction of about 1/256 of the pairs the garbled version of the byte used as input to round 73 in the ninth word again cancels its previous XOR value. Therefore, for this fraction the XOR of this byte after round 72 is zero and the same values are XORed to the tenth word in both executions. The same values are used as inputs to the S boxes in both executions till the next time a byte of word six is used at round 86. The same cancellation should take place five times in rounds 56, 72, 88, 104 and 120. Therefore, the characteristic's probability is about $(1/256)^5 = 2^{-40}$. Each right pair with respect to this characteristic has zero XORs at the first m bits of the input and at the last m bits of the output and thus both messages are hashed to the same value. Figure 2 is a graphic description of the characteristic. In the figure each column represents a word of data and each row

Copyright (c) 1998, Springer-Verlag

represents 16 rounds (represented by the thin lines along the edged). The gray area in the middle represents the modified words in the characteristic. The brighter gray area represets the bytes which are not modified in these words. The two black areas in the top-right and the bottom-left corners point to the words which are used in the calculation of the hash value by the function H. This same attack can break two-pass Snefru with m up to 224 bits. Similar attacks with modification of bytes of three to seven consecutive words of the input XOR of the characteristic are possible with the same characteristic's probability.

The attacks on Snefru can be enhanced in three directions. In one direction black box attacks are developed in which the choice of the S boxes is not known to the attacker. A summary of the results of the black box attacks on Snefru is given in table 1. In the second direction the complexity of the attack is further reduced using the knowledge of the choice of the S boxes. A summary of the attacks on Snefru with known S boxes is given in table 2. The S boxes should be known but the general approach is independent of their choice. In the third direction the attack can find many messages which hash to the same value as a given message. A summary of the attacks on Snefru which find many partners of given messages is given in table 3. All the attacks on Snefru are applicable even if different S boxes are used in different rounds.

A personal computer implementation of this attack on two-pass Snefru finds a pair of messages within three minutes. It finds a partner of a given message in about an hour.

3 Cryptanalysis of Khafre

Khafre[11] is a software oriented cryptosystem with 64-bit blocks whose number of rounds (which should be a multiple of eight) is not yet determined. Each block is divided into two halves called the right half and the left half. In each round the lowest byte of the right half is used as index to an S box with 32-bit output. The left half is XORed with the output of the S box, the right half is rotated and the two halves are exchanged. The rotation is such that every byte is used once every eight rounds as an input to an S box. Before the first round and after every eighth round the data is XORed with 64-bit subkeys. These subkeys are the only way the key is involved in the cryptosystem.

The differential cryptanalysis of Khafre is based upon the observation that the number of output bits of an S box is more than twice the number of input bits. Therefore, given an output XOR of an S box in a pair, the input pair is (usually) unique and it is easy to find the two inputs. Moreover, there are about $(2^8)^2/2 = 2^{15}$ possible input pairs for each S box, thus, only about 2^{-17} of the 32-bit values are possible output XORs of some pair.

A second observation is that there are characteristics in which only one even numbered round (or only one odd numbered round) has non-zero input XOR to the S box. The output XOR of this round in a right pair is easily derivable from the plaintext XOR and

Copyright (c) 1998, Springer-Verlag

No. of passes	m	Char prob	#mod bytes	Complexity of attack	Birthday complexity	Comments
2	128–192	2^{-40}	3	$2^{20.5}$	2^{64}–2^{96}	
	224		2	2^{25}	2^{112}	
	128–192	2^{-40}	4	$2^{20.5}$	2^{64}–2^{96}	Alphanumeric
	224		2	2^{29}	2^{112}	messages
3	128	2^{-72}	3	2^{49}	2^{64}	
	160		2	2^{57}	2^{80}	

Table 1: Summary of the results of the black box attacks on Snefru

No. of passes	m	Char prob	#mod bytes	Complexity of attack	Birthday complexity	Comments
2	128–192	2^{-40}	4·1	$2^{12.5}$	2^{64}–2^{96}	
	224	2^{-56}	2·3	$2^{12.5}$	2^{112}	
	128–192	2^{-40}	4·1	2^{17}	2^{64}–2^{96}	Alphanumeric
	224		2·1	2^{29}	2^{112}	messages
3	128–160	2^{-72}	6·1	$2^{28.5}$	2^{64}–2^{80}	
	192	2^{-80}	4·2	$2^{28.5}$	2^{96}	
	224	2^{-96}	2·4	2^{33}	2^{112}	
4	128–192	2^{-160}	4·4	$2^{44.5}$	2^{64}–2^{96}	
	224	2^{-112}	2·2	2^{81}	2^{112}	

Table 2: Summary of the results of the attacks on Snefru with known S boxes

No. of passes	m	Char prob	#mod bytes	Complexity of attack	Brute force	Comments
2	128–160	2^{-40}	6·1	2^{24}	2^{128}–2^{160}	
	128–160		6·1	2^{40}	2^{128}–2^{160}	Black box
	128–224	2^{-64}	2·4	2^{24}	2^{128}–2^{224}	
	128–160	2^{-40}	7·1	2^{32}	2^{128}–2^{160}	Alphanumeric
	128–160		7·1	2^{40}	2^{128}–2^{160}	Alphanumeric, black box
3	128–224	2^{-96}	2·4	2^{56}	2^{128}–2^{224}	
4	128–192	2^{-160}	4·4	2^{88}	2^{128}–2^{192}	

Table 3: Summary of the results of the attacks which find partners of given messages

Copyright (c) 1998, Springer-Verlag

Rnd	Left Half				Right Half					Output XOR			
Ω_P	0	0	A	0	0	0	B	0					
1	0	0	A	0	0	0	B	0	$\rightarrow$	0	0	0	0
2	B	0	0	0	0	0	A	0	$\rightarrow$	0	0	0	0
3	A	0	0	0	B	0	0	0	$\rightarrow$	0	0	0	0
4	0	B	0	0	A	0	0	0	$\rightarrow$	0	0	0	0
5	0	A	0	0	0	B	0	0	$\rightarrow$	0	0	0	0
6	0	0	0	B	0	A	0	0	$\rightarrow$	0	0	0	0
7	0	0	0	A	0	0	0	B	$\rightarrow$	C	D	E	$A^\dagger$
8	0	0	B	0	C	D	E	0	$\rightarrow$	0	0	0	0
9	D	E	0	C	0	0	B	0	$\rightarrow$	0	0	0	0
10	B	0	0	0	D	E	0	C	$\rightarrow$	$F \oplus B^\ddagger$	G	H	I
11	0	C	D	E	F	G	H	I	$\rightarrow$	J	K	$D \oplus L^\ddagger$	$E^\dagger$
12	I	F	G	H	J	M^0	L	0	$\rightarrow$	0	0	0	0
13	0	J	M^0	L	I	F	G	H	$\rightarrow$	N	$P \oplus J^\ddagger$	Q	$L^\dagger$
14	G	H	I	F	N	P	R^0	0	$\rightarrow$	0	0	0	0
15	R^0	0	N	P	G	H	I	F	$\rightarrow$	S	T	U	$P^\dagger$
16	H	I	F	G	V	T	W^0	0	$\rightarrow$	0	0	0	0
Ω_T	T	W^0	0	V	H	I	F	G					

Table 4: A characteristic of Khafre

the ciphertext XOR. Given this output XOR we can discard most of the wrong pairs by the first observation, leaving only a small fraction of about 2^{-17} of them.

The characteristics of Khafre are described by templates which choose between zero XORs and non-zero XORs. Each right pair may have its own value of the non-zero XORs. The characteristic described in table 4 is used as an example of the cryptanalysis of Khafre with 16 rounds. Each value 0 describes a byte which has equal values in both executions of the encryptions of the pair (zero XOR). Each letter denotes a XOR value which is not zero. A letter with a superscript ^{0} denotes a XOR value which can be either zero or non-zero. The exact values of the non-zero XOR values vary for every right pair. The superscript † means that the byte of the output XOR must be equal to the corresponding byte of the left half in order to cause the input XOR byte of the S box in the next round to be zero. Each occurrence of † causes a reduction of the probability of the characteristic by $\frac{1}{255}$. The superscript ‡ means that the byte of the output XOR must not be equal to the corresponding byte of the left half in order to prevent a zero value in the corresponding byte in the next round, so that it can become zero in one of the following rounds, after XORing with another non-zero value. Each occurrence of ‡ causes a reduction of the probability of the characteristic by $\frac{254}{255}$. Therefore, the probability of this characteristic is $\left(\frac{1}{255}\right)^4 \cdot \left(\frac{254}{255}\right)^3 \approx 2^{-32}$. The input XOR Ω_P of the characteristic has two degrees of freedom: each one of A and B can have 255 possible values. Therefore, the characteristic points to $255^2 \approx 2^{16}$ possible plaintext XORs. Using this characteristic we can break Khafre with 16 rounds using about 2^{34} pairs.

Copyright (c) 1998, Springer-Verlag

Rounds	Char prob	Pairs needed	Chosen plaintexts	Known plaintexts
16	2^{-16}	$3 \cdot 2^{16}$	1536	$2^{37.5}$
24	2^{-56}	2^{60}	2^{53}	$2^{58.5}$

Table 5: Summary of the results of the attacks on Khafre

Differential cryptanalytic attacks can be converted to known plaintext attacks[3,4]. This attack can be converted to a known plaintext attack using about $2^{41.5}$ plaintext/ciphertext pairs. In such an attack, the $2^{41.5}$ plaintexts can form $(2^{41.5})^2/2 = 2^{82}$ pairs. Since there are only 2^{64} possible plaintext XORs (the block size is 64 bits), about $2^{82}/2^{64} = 2^{18}$ pairs occur with each plaintext XOR. There are about 2^{16} usable input XORs of the characteristic and thus we get about $2^{16} \cdot 2^{18} = 2^{34}$ candidate pairs which can be used to break khafre with 16 rounds.

A summary of the best results we obtained for 16-round Khafre and 24-round Khafre is given in table 5 which describes the number of pairs needed for the attack, the number of chosen plaintexts needed, and the number of known plaintexts needed. Note that these complexities are independent of the actual choice of the S boxes, although the S boxes themselves should be known. The attacks are applicable even if different S boxes are used in different rounds. Our personal computer implementation of the chosen plaintext attack on 16-round Khafre takes about an hour.

4 Cryptanalysis of REDOC-II

REDOC-II[6,7] is a ten-round cryptosystem with 70-bit blocks (arranged as ten bytes of seven bits). Each round contains six phases: (1) First variable substitution, (2) Second variable substitution, (3) First variable key XOR, (4) Variable enclave, (5) Second variable key XOR and (6) Variable permutation.

An important property of the enclave tables is that they are linear operations in terms of addition which can be described as the product of a fixed matrix with the current vector of bytes. By modifying only upper bits in the input, only upper bits in the output are modified. Moreover, the linear modification table of the upper output bits by the upper input bits uniquely identifies the enclave table used. This property can even be used in the variable enclave phase. The left half of the input with two of the bytes of the right half affect the choice of the enclave tables used in this phase. However, three of the bytes of the right half do not affect the choice of the enclave tables (in the first round they are the eighth, ninth and the tenth bytes), and thus the modifications of the upper bits of the output are linear functions of the modifications of the upper bits of the input. Note that since we XOR the left half with the right half as the last step in the variable enclave phase we get a symmetric modification in both halves and therefore, an even number of modi

Copyright (c) 1998, Springer-Verlag

In this attack we use the following characteristic:

After Phase	Data XOR										
Ω_P	0	0	0	0	0	0	0	A	0	0	
First Subst	0	0	0	0	0	0	0	B	0	0	For some B
Second Subst	0	0	0	0	0	0	0	64	0	0	with probability about 1/128
Key XOR	0	0	0	0	0	0	0	64	0	0	
Enclave	C	0	D	E	F	C	0	D	E	F	with probability about 1/2
Key XOR	C	0	D	E	F	C	0	D	E	F	
Permutation	Some permutation of C,0,D,E,F,C,0,D,E,F										
Ω_T	Some permutation of C,0,D,E,F,C,0,D,E,F										

where $A, B \in \{1, \ldots, 127\}$ and $C, D, E, F \in \{0, 64\}$ (not all of them zero). In total this characteristic has probability about $\frac{1}{256}$. The ciphertext XOR has 60 zero bits (six in each byte) and the XORed value of the uppermost bits is zero as well. Similar characteristics exist in which the difference is at the ninth and tenth bytes rather than at the eighth byte. Differences in more than one of these three bytes is also possible with smaller probabilities, but if the difference is the same in all the differing bytes and the values of all the differing bytes in the plaintexts are equal then the probability stays about $\frac{1}{256}$.

Given sufficiently many pairs encrypted by one-round REDOC-II with the plaintext differences specified in the characteristics we can discard (almost) all the wrong pairs by verifying that the 61 bits of the ciphertext XORs $(60 + 1)$ are really zero. Only a negligible fraction of 2^{-61} of the wrong pairs may remain. In practice, only right pairs remain.

For each of the $16 \cdot 16 = 256$ possible values of the masks of the substitution phases we count the number of pairs whose differing byte after the two substitutions resulting from the masks differ only by the uppermost bit. For each one of the 128 possible values of the mask of the permutation phase we count the number of pairs whose ciphertext XOR permuted by the resulting inverse permutation is symmetric and has zeroes in the second and the seventh bytes. The right values of these mask bytes are likely to be the ones counted most frequently and thus can be identified. This attack needs about 1000 pairs and finds three masks of the processed key.

The attack can be enhanced by using structures of 32 encryptions with identical nine bytes and whose tenth byte gets 32 different values. In such a structure there are 496 pairs. There are only 128 possible differences after the second substitution and thus there are about four pairs which differ only by one uppermost bit after the substitution phases. These four pairs use the same enclave tables and thus with probability about half the structure contains four right pairs, and with probability about half does not contain any right pair. Using three such structures with identical eight bytes, 32 plaintexts differ at the ninth byte, 32 differ at the tenth byte and 32 differ at both the ninth and the tenth bytes with equal values in both bytes in each plaintext, we are guaranteed to have at least one structure whose choosing byte of the second key XOR has no difference and thus to have about four right pairs. This enhanced attack needs only 96 chosen plaintexts and their corresponding ciphertexts. REDOC-II with up to four rounds is also vulnerable to this attack. The attack can again be converted to a known plaintext attack. Chosen

Copyright (c) 1998, Springer-Verlag

Rnds	Char prob	Pairs needed	Chosen plains	Chosen ciphers	Known plains	Comments
1	2^{-8}	–	2300	–	–	All masks + key tables
1	2^{-8}	1000	96	40	$2^{35.5}$	Three masks
2	2^{-29}	2^{31}	2^{25}	2^{24}	2^{46}	Three masks
3	2^{-50}	2^{52}	2^{46}	2^{45}	$2^{56.5}$	Three masks
4	2^{-71}	2^{73}	2^{67}	2^{66}	2^{67}	Three masks

Table 6: Summary of the results of the attacks on REDOC-II

ciphertext attacks on these variants which find the three masks are also possible and need about a third of the data compared to the chosen plaintext attacks.

A summary of the results on REDOC-II is given in table 6.

The three masks of the substitution and the permutation phases of the one-round variant can be found within less than a second on a personal computer using 96 encryptions. A more complicated attacking program on one-round REDOC-II (which cannot be described in this extended abstract) finds all the masks and all the key tables in about a minute using about 2300 encryptions with more than 90% success rate. Using about 3900 encryptions the success rate becomes better than 99%.

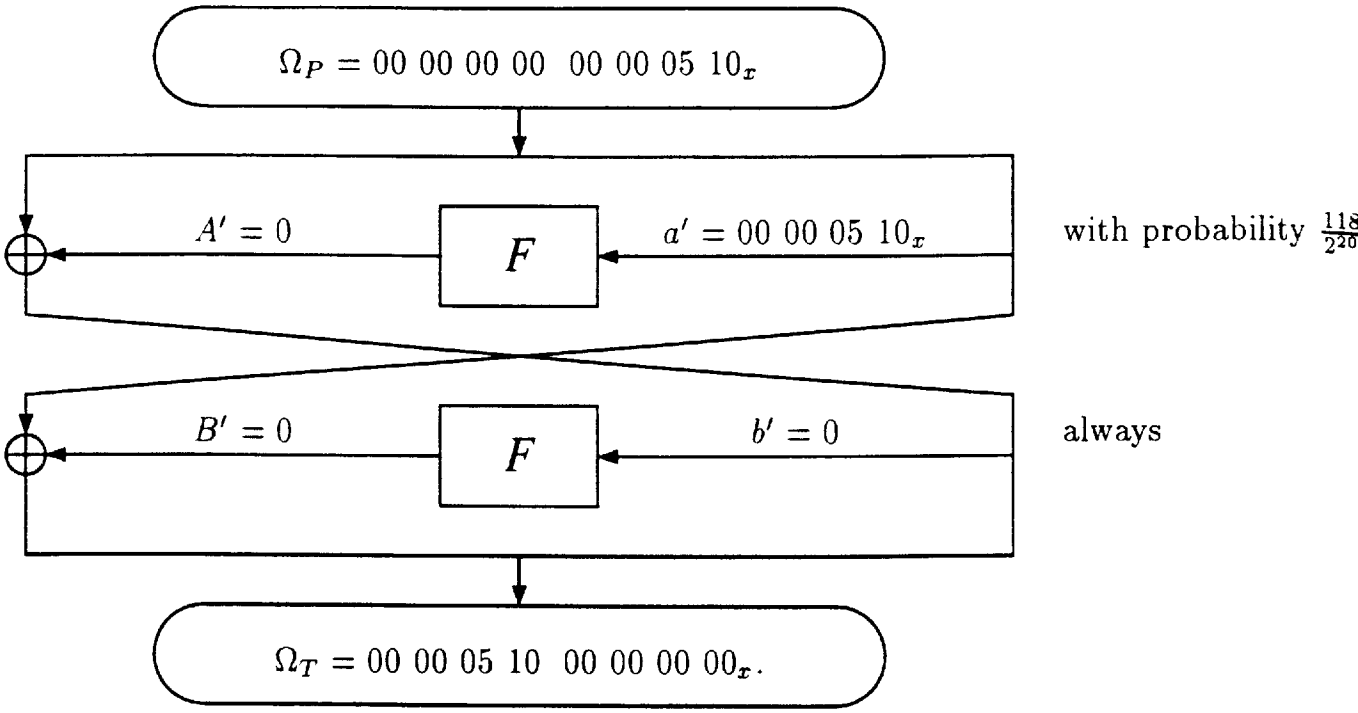
5 Cryptanalysis of LOKI

LOKI[5] is a 64-bit key/64-bit block cryptosystem similar to DES which uses one twelve-bit to eight-bit S box (based on irreducible polynomials) replicated four times in each round. The expansion and the permutation are replaced by new choices and the initial and final transformations are replaced by XORs with the key. The bit permutations in the key scheduling are replaced by rotations and the subkeys become 32-bit long. The XOR of the input of the F function with the key is done before the expansion and therefore neighboring S boxes receive common bits. Two new modes of operation which convert LOKI into a hash function are defined.

The pairs XORs distribution table of the larger S box of LOKI has much smaller probabilities than the ones of DES (average $\frac{1}{256}$ and maximum $\frac{1}{64}$). However, it is possible to have non-zero XORs in the inputs of two S boxes resulting with the same output, whereas in DES this requires at least three S boxes. We have found the following two-round iterative characteristic with probability $\frac{118}{2^{20}} \approx 2^{-13.12}$ (this probability is calculated using the observation that two neighboring S boxes have four common input

Copyright (c) 1998, Springer-Verlag

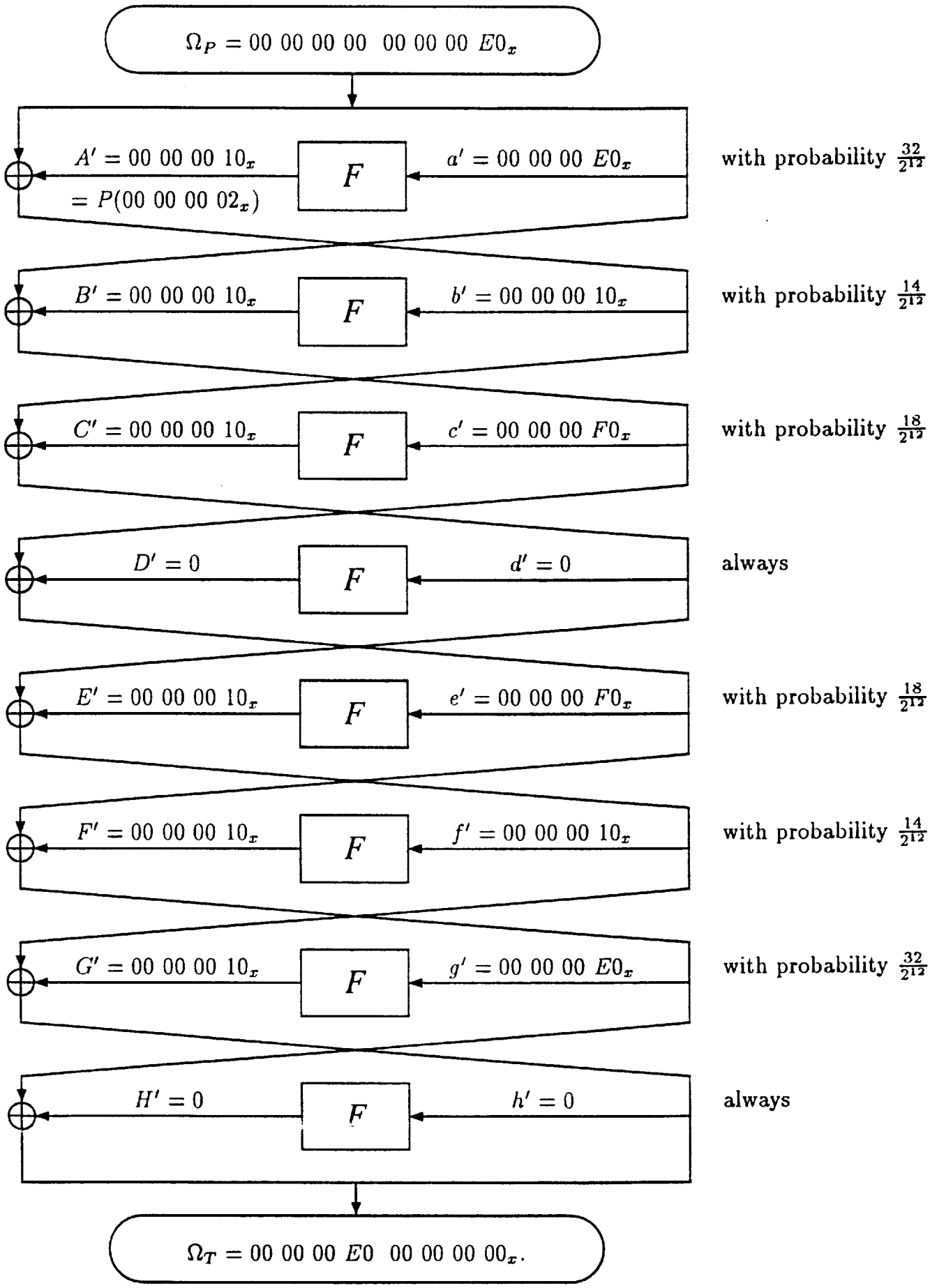
bits, otherwise we get slightly smaller probability):



This characteristic can be iterated to nine rounds with probability about $2^{-52.5}$ and to eleven rounds with probability about $2^{-65.5}$. Since all the four S boxes of LOKI are the same and all the output XORs in this characteristic are zero, there are three similar characteristics in which the XOR pattern is rotated by units of eight bits. There is another eight-round iterative characteristic in which only non-replicated bits of some S

Copyright (c) 1998, Springer-Verlag

box are different and the outputs differ only by one bit. This characteristic is:



Copyright (c) 1998, Springer-Verlag

This iterative characteristic has probability about 2^{-46} and its extension to nine rounds has the same probability. Using this characteristic it is possible to break LOKI with up to eleven rounds with less than 2^{64} chosen or known plaintexts.

Careful analysis of the structure of LOKI has revealed that any key has 15 equivalent keys which encrypt any plaintext to the same ciphertext due to a key complementation property. These 15 keys are the key XORed with the 15 possible 64-bit hexadecimal numbers whose digits are equal (i.e., $hhhhhhhhhhhhhhhh_x$ where $h \in \{1_x, \ldots, F_x\}$). Encryption with these keys results with the same inputs to the F functions in all the 16 executions. Therefore, most of the keys are redundant and a known plaintext attack can have complexity 2^{60} rather than 2^{64}.

Another complementation property is due to the observation that XORing the key with an hexadecimal value $gggggggghhhhhhhh_x$ (or with $hhhhhhhhgggggggg_x$) and XORing the plaintext by $iiiiiiiiiiiiiiii_x$ where $g \in \{0_x, \ldots, F_x\}$, $h \in \{0_x, \ldots, F_x\}$ and $i = g \oplus h$ results in XORing the ciphertext by $iiiiiiiiiiiiiiii_x$. This property can be used to reduce the complexity of a chosen plaintext attack by a further factor of 16 to 2^{56}.

These observations result in major weaknesses when LOKI is used as a hash function. For any message it is easy to find 15 additional messages which hash to the same value by the Single Block Hash (SBH) mode of LOKI: the other messages are the given message XORed with each of the 15 hexadecimal values $hhhhhhhhhhhhhhhh_x$. Since the messages are used as the key of the LOKI primitive (XORed with the previous hash value which can be viewed as a fixed value) and the plaintext of LOKI is fixed, the outputs of all the executions are the same by the first complementation property.

For any message it is easy to find 255 other messages which hash to the same value by the Double Block Hash (DBH) mode of LOKI provided the initial value is changed. This is done by XORing H_{-1} and M_2 by $gggggggghhhhhhhh_x$ and XORing M_1 by $hhhhhhhhgggggggg_x$ without changing H_0 (where $g \in \{0_x, \ldots, F_x\}$ and $h \in \{0_x, \ldots, F_x\}$).

LOKI has 256 simple fixpoints $\mathrm{LOKI}_K(X) = X$ where $K = gggggggghhhhhhhh_x$, $X = iiiiiiiiiiiiiiii_x$, $g, h \in \{0_x, \ldots, F_x\}$ and $i = g \oplus h$. In particular, LOKI encrypts the plaintext zero by the key zero to the ciphertext zero: $\mathrm{LOKI}_0(0) = 0$. Therefore, the two hash function modes hash the zero messages with the zero initial value to zero. This observation shows that the zero initial value should be avoided since any number of zero-blocks (or any even number in the DBH mode) can be prepended to the message without modifying the hash value. Moreover, in the SBH mode all the 16 initial values $H_0 = hhhhhhhhhhhhhhhh_x$ should be avoided since the message $00000000hhhhhhhh_x$ and 15 others hash to the initial value $H_1 = hhhhhhhhhhhhhhhh_x$. In the DBH mode all the 256 initial values $H_{-1} = 0$ and $H_0 = gggggggghhhhhhhh_x$ should be avoided since the messages $M_1 = hhhhhhhhgggggggg_x$ and $M_2 = iiiiiiiiiiiiiiii_x$ where $i = g \oplus h$ hash to the initial value and can be prepended any number of times without affecting the hash value.

After this research was completed, Matthew Kwan[9] found a three-round iterative

Copyright (c) 1998, Springer-Verlag

Input	Output of S_0	Output of S_1	Equal bits
0000	0100	1111	.1..
0001	0001	1100	..0.
0010	1110	1000	1..0
0011	1000	0010	.0.0
0100	1101	0100	.10.
0101	0110	1001	....
0110	0010	0001	00..
0111	1011	0111	..11
1000	1111	0101	.1.1
1001	1100	1011	1...
1010	1001	0011	.0.1
1011	0111	1110	.11.
1100	0011	1010	.01.
1101	1010	0000	.0.0
1110	0101	0110	01..
1111	0000	1101	..0.

Table 7: Output bits that are equal for both S boxes (left table)

Input	Equal bits
.000	.1..
0.00	.1..
001.	...0
.110	0...
10.0	...1
110.	.0..

Table 8: Output bits that are equal for both S boxes and two input values (right table)

characteristic of LOKI with probability $2^{-14.4}$ which can be use to break LOKI with up to 14 rounds. He also found many more fixpoints of LOKI.

6 Cryptanalysis of Lucifer

Lucifer[8] is a substitution/permutation network cryptosystem designed by IBM prior to the design of DES. In Lucifer the input of the S boxes is the bit permuted output of the S boxes of the previous round. The input of the S boxes of the first round is the plaintext itself. A key bit is used to choose the actual S box at each entry out of two possible four-bit to four-bit invertible S boxes.

Given an input of an S box the outputs of the two possible S boxes are known. Each output bit may be equal in the two S boxes or may be different. Usually only one or two output bits are equal in the two S boxes. In few cases one output bit is equal in all the four output values obtained when two input values differing by one bit (for example 8_x and A_x) enter the two possible S boxes. In particular, there are pairs for which three output bits are equal and the fourth bit differ when using different S box.

For the sake of concreteness, we use the third and fourth lines of S1 of DES as the S boxes S_0 and S_1 of Lucifer. Other choices of the S boxes give similar results. Table 7 describes the S boxes Copyright (c) 1998, Springer-Verlag outputs of the two S boxes. We see that

eleven inputs cause two equal bits, four cause one equal bit and one input does not cause any equal bit. Table 8 describes the equal bits of two input values which differ by one bit using both S boxes. A binary notation is used in the tables.

By consulting these tables we can create many plaintexts whose particular (chosen) bit at an interior round has a chosen fixed value, regardless of the choice of the key. We can also create pairs of plaintexts which differ in a later round only at a particular bit. Lucifer with eight rounds can be attacked using the encryptions of such plaintexts. The attack on Lucifer with 128-bit blocks with eight rounds needs about 24–30 encryptions and takes about 2^{21} steps. Its personal computer implementation requires a few seconds with almost 100% success rate.

Another variant of Lucifer which is described in [14] is more similar to DES. Its variant with eight rounds is weaker than the eight-round variant of Lucifer described in [8].

References

[1] Eli Biham, Adi Shamir, *Differential Cryptanalysis of DES-like Cryptosystems (extended abstract)*, Advances in cryptology, proceedings of CRYPTO 90, 1990.

[2] Eli Biham, Adi Shamir, *Differential Cryptanalysis of DES-like Cryptosystems*, Journal of Cryptology, Vol. 4, No. 1, 1991, to appear.

[3] Eli Biham, Adi Shamir, *Differential Cryptanalysis of Feal and N-Hash (extended abstract)*, Advances in cryptology, proceedings of EUROCRYPT 91, 1991.

[4] Eli Biham, Adi Shamir, *Differential Cryptanalysis of Feal and N-Hash*, in preperation.

[5] Lawrence Brown, Josef Pieprzyk, Jennifer Seberry, *LOKI - A Cryptographic Primitive for Authentication and Secrecy Applications*, Advances in cryptology, proceedings of AUSCRYPT 90, pp. 229–236, 1990.

[6] Michael C. Wood, technical report, Cryptech inc., Jamestown, NY, July 1990.

[7] Thomas W. Cusick, Michael C. Wood, *The REDOC-II Cryptosystem*, Advances in cryptology, proceedings of CRYPTO 90, 1990.

[8] H. Feistel, *Cryptography and Data Security*, Scientific American, Vol. 228, No. 5, pp. 15–23, May 1973.

[9] Matthew Kwan, private communications.

[10] Ralph C. Merkle, *A Fast Software One-Way Hash Function*, Journal of Cryptology, Vol. 3, No. 1, pp. 43-58, 1990.

[11] Ralph C. Merkle, *Fast Software Encryption Functions*, Advances in cryptology, proceedings of CRYPTO 90, 1990.

[12] Shoji Miyaguchi, Akira Shiraishi, Akihiro Shimizu, *Fast Data Encryption Algorithm Feal-8*, Review of electrical communications laboratories, Vol. 36, No. 4, pp. 433–437, 1988.

Copyright (c) 1998, Springer-Verlag

[13] Akihiro Shimizu, Shoji Miyaguchi, *Fast Data Encryption Algorithm Feal*, Advances in cryptology, proceedings of EUROCRYPT 87, pp. 267–278, 1987.

[14] Arthur Sorkin, *Lucifer, a Cryptographic Algorithm*, Cryptologia, Vol. 8, No. 1, pp. 22–41, January 1984.

Copyright (c) 1998, Springer-Verlag