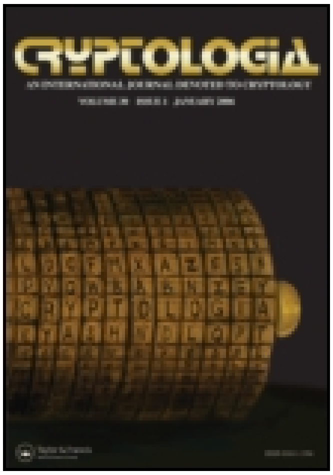

Cryptologia
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/ucry20

CIPHERTEXT-ONLY ATTACK ON AKELARRE

Lars R. Knudsen MSc and Phd [a] & Vincent Rijmen Phd [b]

[a] University of Bergen, Department of Informatics , N-5020, Bergen, NORWAY E-mail:

[b] Katholieke Universiteit Leuven , ESAT-COSIC, K. Mercierlaan 94, B-3001, Heverlee, BELGIUM E-mail:

Published online: 22 Aug 2007.

CIPHERTEXT-ONLY ATTACK ON AKELARRE

Lars R. Knudsen[1] and Vincent Rijmen[2]

ADDRESS: (1) University of Bergen, Department of Informatics, N-5020 Bergen NORWAY. lars.knudsen@ii.uib.no and (2) Katholieke Universiteit Leuven, ESAT-COSIC, K. Mercierlaan 94, B-3001 Heverlee BELGIUM. vincent.rijmen@esat.kuleuven.ac.be.

ABSTRACT: At the SAC'96 the iterated block cipher, Akelarre, was proposed. Akelarre uses components of the block ciphers RC5 and IDEA and is conjectured strong with four rounds. This paper shows that Akelarre with any number of rounds is weak even under a ciphertext only attack. This illustrates that mixing two (presumably) strong ciphers is not always a good idea.

KEYWORDS: Secret-key cryptosystem, block cipher, cryptanalysis.

1 INTRODUCTION

At the SAC'96 the block cipher, Akelarre, was proposed [1]. Akelarre is an iterated cipher, which uses components of the block ciphers RC5 [13] and IDEA [10]. A comparison is made to these block ciphers in favor of Akelarre. In the following we will show that Akelarre is a weak block cipher with any number of rounds. The paper is organized as follows. Section 2 starts with a short description of the structure of IDEA and RC5 and the currently known attacks on both block ciphers. We proceed with a short description of Akelarre containing the details necessary for our attacks and refer to [1] for the full description. In particular, the details of the key scheduling algorithm are not relevant for our attack and will not be discussed here. In Section 3 we describe the main weakness of Akelarre, which forms the basis of our attacks. In Section 4 a known plaintext attack and a ciphertext only attack are given and the Conclusion contains our concluding remarks.

Independent of our work is the analysis of Akelarre [6], which contains a chosen plaintext attack with a full recovery of the secret key. Our analysis enables us to deduct a substantial amount of information about the plaintext alone from the ciphertext without doing a full recovery of the cipher key. A preliminary version of this paper appears in [9].

2 DESCRIPTION OF AKELARRE

Before we delve into the details of Akelarre, we give a short description of the structure of IDEA and RC5 and give a short overview of the cryptanalytic results on both ciphers.

2.1 A Short Description of IDEA

The block cipher IDEA (International Data Encryption Algorithm) was proposed by X. Lai, J. Massey and S. Murphy. IDEA is an iterated block cipher, consisting of 8 rounds followed by an output transformation. It has a 128-bit key and operates on data blocks of 64 bits. Figure 1 shows the round transformation structure of IDEA. The round transformation divides the data into four 16-bit blocks and uses three different operations: bitwise exor (shown as '$\oplus$' on the figure), addition modulo 2^{16} ('+') and multiplication modulo $2^{16}+1$ ('$\bullet$'), where 0 represents the element 2^{16}. The two multiplications and the two additions in the middle of the round transformation are called the MA-structure.

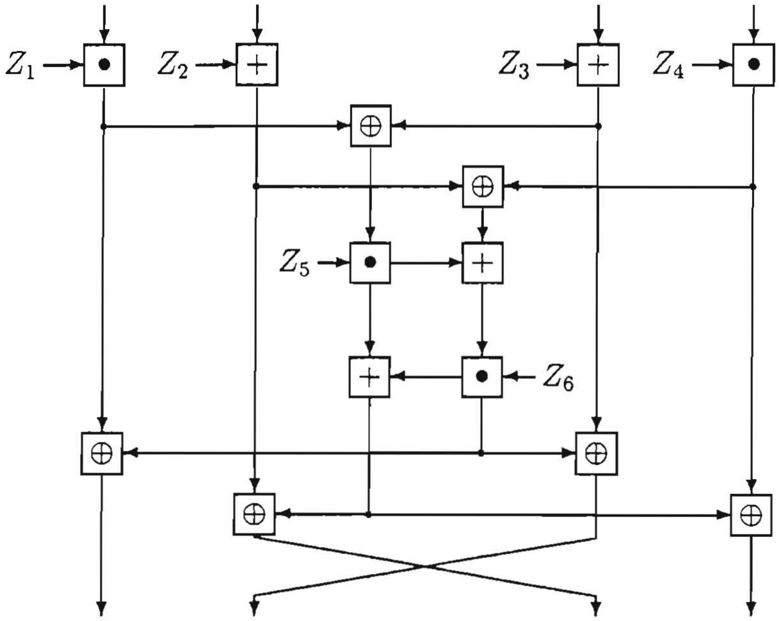

Figure 1. The Round transformation of IDEA.

Attempts to cryptanalyse IDEA have resulted in the following attacks.

Partial distributivity: W. Meier cryptanalysed IDEA reduced to two rounds, with 2^{10} chosen plaintexts and a computational effort of about 2^{42} encryptions [12].

Weak keys: J. Daemen discovered that the strength of IDEA depends on the specific key that is being used. He developed an attack that works for a subset of 2^{51} keys. The attack uses 16 chosen plaintexts [4] and has a workload of 2^{17} encryptions.

Differential attack: J. Daemen cryptanalysed IDEA reduced to two rounds with the output transformation, in a differential attack. The attack requires 2^{10} chosen plaintexts and has a workload of about 2^{30} encryptions [5].

Differential-linear attack: J. Borst, L. Knudsen and V. Rijmen cryptanalysed IDEA reduced to three rounds, with a differential-linear attack. The attack requires 2^{29} chosen plaintexts and has a workload of about 2^{44} encryptions [3]

Truncated differential attack: J. Borst, L. Knudsen and V. Rijmen also described a truncated differential attack for IDEA reduced to three rounds and the output transformation [3]. The amount of required plaintexts and the workload depend on the actual key that is used. 1% of the keys can be recovered with 2^{40} chosen plaintexts and an effort of 2^{51} encryptions.

We can conclude that until now IDEA has withstood fairly well all public attempts of cryptanalysis.

2.2 A Short Description of RC5

RC5 is a block cipher designed by R. Rivest. It has a variable block size, a variable number of rounds and a variable key length. In a nominal choice [13] these parameters are chosen as follows: 64-bit blocks, 12 rounds and 128-bit keys. The algorithm uses exor operations ('$\oplus$'), modular additions ('+') and data-dependent rotations ('<<<'). Let the plaintext be denoted with (L_0, R_0) and the ciphertext with (L_{2r+1}, R_{2r+2}), where r is the number of rounds. The key scheduling converts the cipher key into an array S with $2r+2$ 32-bit entries. The algorithm executes the following operations.

```
$L_1 = L_0 + S_0$
$R_1 = R_0 + S_1$
for $i = 2$ to $2r + 1$ do
        $L_i = R_{i-1}$
        $R_i = ((L_{i-1} \oplus R_{i-1}) <<< R_{i-1}) + S_i$
```

Analysis of RC5 has resulted in the following attacks.

Differential attack: B. Kaliski and Y. Yin described a differential attack on RC5 with 12 rounds. The attack requires 2^{63} chosen plaintexts and has a negligible workload [7].

Linear attack: In the same paper, a linear is described on RC5, reduced to 6 rounds. This attack requires 2^{57} known plaintexts.

Improved differential attack: L. Knudsen and W. Meier showed a technique to improve the differential attack on RC5 with a large factor. Their attack on RC5 with 12 rounds needs only 2^{54} chosen plaintexts [8].

Further improved differential attack: Biryukov and Kushilevitz further improved the differential attack on RC5 with 12 rounds to 2^{44} chosen plaintexts [2].

The number of rounds of RC5 can be increased to thwart the above attacks, e.g., to 16 rounds [8]. On the contrary, the attack on Akelarre that we describe in Section 4 can be applied to *any* number of rounds.

2.3 Akelarre

Akelarre [1] is a 128-bit block cipher. The key length is variable, but always a multiple of 64. Akelarre has an input transformation, an output transformation and a variable number of rounds. It is proposed with four rounds. The round transformation of Akelarre has a structure similar to that of IDEA. Figure 2 gives a picture of Akelarre with one round. To simplify notation a different numbering for the round keys has been adopted. Our attacks are independent of the 12 round keys in each AR-structure, which therefore are denoted Z_r for round r.

The similarity between the round transformations of IDEA and Akelarre is obvious from a visual inspection. The main differences between the round transformations are the following:

- Akelarre uses 32-bit words instead of 16-bit words.
- Akelarre does not swap the inner two words.
- The multiplication-addition structure (MA-structure) of IDEA is replaced by a complex addition-rotation structure (AR-structure).
- No modular multiplications are used.

- In the round transformation of IDEA there are key additions inside and outside the MA-structure, in Akelarre there are only key additions in the AR-structure.

- The key scheduling is more complicated and difficult to invert.

The AR-structure of Akelarre consists of 12 31-bit data dependent rotations and 12 key additions. This structure is reminiscent of the RC5 round operation.

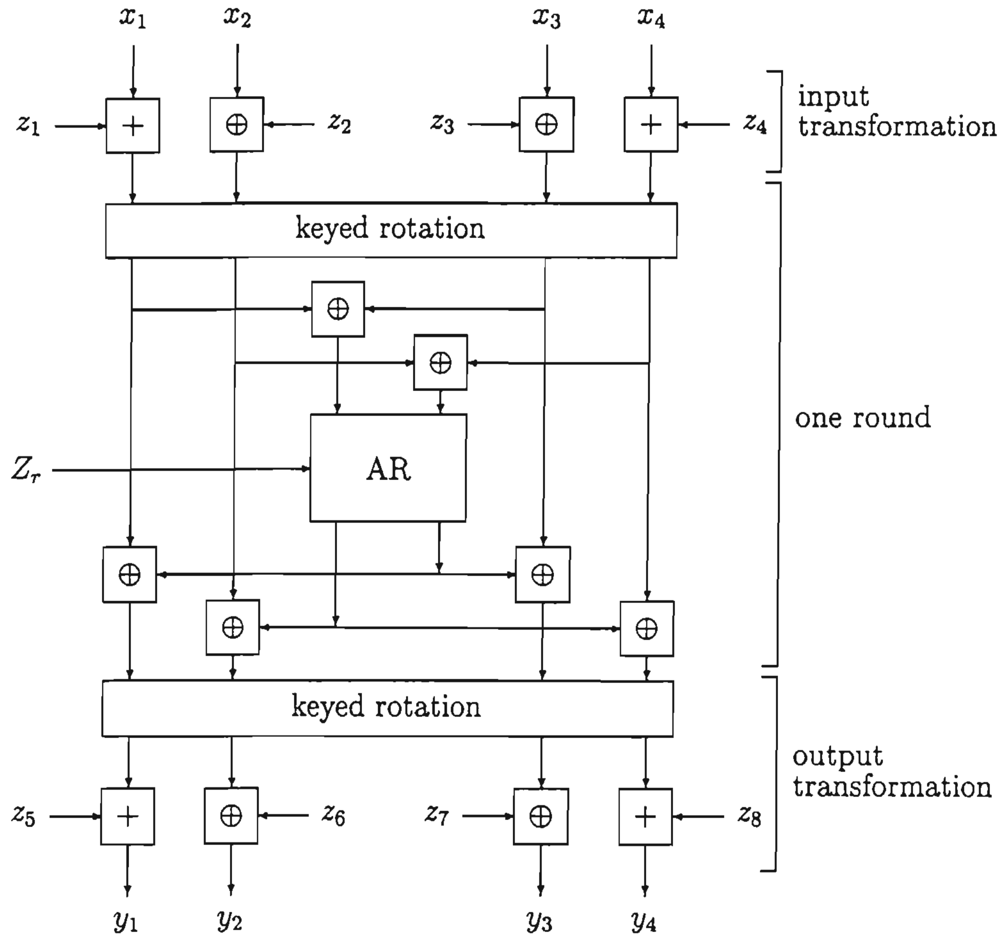

Figure 2. Computational graph of Akelarre.

3 A WEAKNESS OF THE ROUND TRANSFORMATION

Before explaining the weakness of the round transformation, we fix a notation for the relevant variables. Let the plaintext be denoted with $x_1 \,\|\, x_2 \,\|\, x_3 \,\|\, x_4$, where '||' is the concatenation of bit strings. The input transformation of Akelarre consists of adding modulo 2^{32} respectively exoring the key words z_1, z_2, z_3 and z_4 to the plaintext (cf. Figure 2). We denote the input of round i with $x_1^i \,\|\, x_2^i \,\|\, x_3^i \,\|\, x_4^i, i = 1, \dots R$. The round transformation for round i is as follows.

$$\begin{aligned} u_1^i \,\|\, u_2^i \,\|\, u_3^i \,\|\, u_4^i &= \mathrm{rot}_{r^i}(x_1^i \,\|\, x_2^i \,\|\, x_3^i \,\|\, x_4^i) &\quad (1)\\ (t_1, t_2) &= \mathrm{AR}(u_1^i \oplus u_3^i, u_2^i \oplus u_4^i) &\quad (2)\end{aligned}$$

$$\begin{aligned} x_1^{i+1} &= u_1^i \oplus t_1 &\quad (3a)\\ x_2^{i+1} &= u_2^i \oplus t_2 &\quad (3b)\\ x_3^{i+1} &= u_3^i \oplus t_1 &\quad (3c)\\ x_4^{i+1} &= u_4^i \oplus t_2 &\quad (3d)\end{aligned}$$

where rot_{r^i} is the key dependent 128-bit rotation by r^i positions. The inputs of the output transformation are denoted x_i^{R+1}. The output transformation consists of adding modulo 2^{32} respectively exoring the key words z_5, z_6, z_7 and z_8.

The weakness of the round transformation of Akelarre is that it exhibits the following invariant relation between the input and the output.

$$\begin{aligned} (x_1^{i+1} \| x_2^{i+1}) \oplus (x_3^{i+1} \| x_4^{i+1}) &= (u_1^{i+1} \| u_2^{i+1}) \oplus (u_3^{i+1} \| u_4^{i+1})\\ &= \mathrm{rot}_{r^i \bmod 64}((x_1^i \| x_3^i) \oplus (x_3^i \| x_4^i)) &\quad (4)\end{aligned}$$

This relation can be extended over any number of rounds. After R rounds we have

$$(x_1^{R+1} \| x_2^{R+1}) \oplus (x_3^{R+1} \| x_4^{R+1}) = \mathrm{rot}_{s^{R-1}}((x_1^1 \| x_2^1) \oplus (x_3^1 \| x_4^1)),$$

where $s^R = (\sum_{i=1}^{R} r^i) \bmod 64$. This relation does not hold during the input transformation and the output transformation. Let $s^\rho = s^R + r^{R+1}$, where the last term denotes the rotation amount in the output transformation. Then we can express the following relation between the input and the output of the cipher:

$$\begin{aligned} &((y_1 - z_5) \| (y_2 \oplus z_6)) \oplus ((y_3 \oplus z_7) \| (y_4 - z_8))\\ &= \mathrm{rot}_{s^\rho}(((x_1 + z_1) \| (x_2 \oplus z_2)) \oplus ((x_3 \oplus z_3) \| (x_4 + z_4))). \quad (5)\end{aligned}$$

All variables in this equation are concatenations of two 32-bit words. It can therefore be seen as a set of 64 nonlinear relations between bits of the plaintext, ciphertext and key.

Suppose now that the keys $z_i, i = 1, \ldots 8$ and s^ρ are known. Then, given a 128-bit ciphertext block, it is possible to determine the class of 2^{64} 128-bit plaintexts that are a solution of (5). In other words, 64 bits of information about the plaintext are leaked. This situation is comparable to an encryption with a one time pad where the key is used twice. If the plaintext contains enough redundancy, it can be uniquely determined.

As we will show in the following section, it is possible to determine the keys of (5). Once these keys have been determined the attacker gets immediate information about the plaintexts from intercepted ciphertexts. We first describe a known plaintext attack, then a ciphertext only attack. Both attacks assume that some statistics of the plaintext are known. If the encrypted text is an English text, a LaTeX document, or even consists of random ASCII-characters this gives enough redundancy to recover the key.

4 CRYPTANALYSIS OF AKELARRE

We describe two attacks that do not recover the complete Akelarre key, but give enough information about the key to allow the cryptanalyst to recover the plaintexts from the ciphertexts.

4.1 A Known Plaintext Attack

In a known plaintext attack, we assume that the attacker has some ciphertexts and the corresponding plaintexts at his disposal. The attacker cannot choose the specific value of the ciphertexts or the plaintexts.

4.1.1 Recovering the keys

We first look at a simplified version of (5), where all modular additions have been replaced by exors:

$$\begin{aligned}&((y_1 \oplus z_5)\|(y_2 \oplus z_6)) \oplus ((y_3 \oplus z_7)\|(y_4 \oplus z_8))\\ &\quad = \mathrm{rot}_{s^\rho}(((x_1 \oplus z_1)\|(x_2 \oplus z_2)) \oplus ((x_3 \oplus z_3)\|(x_4 \oplus z_4))),\end{aligned} \tag{6}$$

or,

$$\begin{aligned}&(y_1\|y_2) \oplus (y_3\|y_4) \oplus (z_5\|z_6) \oplus (z_7\|z_8)\\ &\quad = \mathrm{rot}_{s^\rho}((x_1\|x_2) \oplus (x_3\|x_4) \oplus (z_1\|z_2) \oplus (z_3\|z_4)).\end{aligned} \tag{7}$$

Given two or three known plaintexts, it is easy to solve this equation for s^ρ and

$$(z_5\|z_6) \oplus (z_7\|z_8) \oplus \mathrm{rot}_{s^\rho}((z_1\|z_2) \oplus (z_3\|z_4)),$$

which accounts for 64 bits information on the key.

We now return to the case with modular additions. The equations for the key bits are now nonlinear: on a bit level, the exor operation is linear and the modular addition is nonlinear because of the carry bits. The nonlinearity will allow us to determine more than 64 key bits.

The nonlinear equations can be solved in the following way. Suppose that $s^\rho = 0$. We rewrite (5) on bit level, starting with the least significant bits of the right 32-bit word:

$$\begin{aligned}
y_2[0] \oplus z_6[0] \oplus y_4[0] \quad &\oplus \quad (-z_8)[0] \\
&= \quad x_2[0] \oplus z_2[0] \oplus x_4[0] \oplus z_4[0] \\
y_2[1] \oplus z_6[1] \oplus y_4[1] \quad &\oplus \quad (-z_8)[1] \oplus y_4[0] \cdot (-z_8)[0] \\
&= \quad x_2[1] \oplus z_2[1] \oplus x_4[1] \oplus z_4[1] \oplus x_4[0] \cdot z_4[0] \\
\ldots \quad &= \quad \ldots
\end{aligned}$$

By collecting a few plaintext-ciphertext pairs with different values in x_4 and y_4, we can solve these equations for $z_4, z_8, z_2 \oplus z_6$ and in the same way we determine z_1, z_5 and $z_3 \oplus z_7$. Since there is no carry from the most significant bit, we cannot determine uniquely the most significant bits of z_1, z_4, z_5 and z_8.

In the more general case, s^ρ can take on any value between 0 and 63. To solve the equations we first adopt a guess for s^ρ. Then we write the equations on bit level, starting from the least significant bits of the y_i-variables. Since we do not necessarily begin with the least significant bit of the x_i-variables, we will have to guess two incoming carry bits. When the equations are solved, we collect a few more plaintext-ciphertext pairs to verify our solution, wrong guesses for s^ρ will fail this test. For every value of s^ρ, there will be some subkey bits that we cannot determine uniquely because only exor information is available. We will be able to determine 195 of the 263 subkey bits (8 32-bit z_i's and one 7-bit s^ρ). This attack requires at most four or five known plaintexts and has a very small work factor.

4.1.2 Recovering plaintexts

After recovering the keys as described in the previous section, the cryptanalyst can examine new ciphertexts and try to recover the plaintexts from them. This will only be possible if the plaintext contains some redundancy. To simplify the discussion we assume that $s^\rho = 0$. In that case (5) reduces to the following:

$$\begin{aligned}
(x_1 + z_1) \oplus x_3 \quad &= \quad (y_1 - z_5) \oplus y_3 \oplus z_7 \oplus z_3 &&(8a)\\
(x_4 + z_4) \oplus x_2 \quad &= \quad (y_4 - z_8) \oplus y_2 \oplus z_6 \oplus z_2\,. &&(8b)
\end{aligned}$$

The right hand sides of (8) are known. A cryptanalyst who tries to determine $x_1 \,\|\, x_2 \,\|\, x_3 \,\|\, x_4$, faces a problem that has a strong resemblance to the decryption of a one time pad where the key has been used twice. The situation is a bit more complicated, because there are actually two pads used, one for the even numbered words and one for the odd numbered words. If the right hand sides are denoted k_1 and k_4, the plaintexts are given by

$$x_1\|x_2\|x_3\|x_4 = a\|b\|((a + z_1) \oplus k_1)\|((b \oplus k_4) - z_4)\,,$$

where the two 32-bit values a and b are not known by the cryptanalyst, and different for each 128-bit plaintext block. If the plaintext contains enough redundancy, this problem can be solved. Even if the redundancy of the plaintext is small, there is a leak of plaintext information to the ciphertext.

4.2 A Ciphertext Only Attack

In some cases it is possible to recover the eight key words z_i and s^ρ using only statistical information on the distribution of the plaintext. After these keys have been recovered, the approach of the previous section can be used to recover plaintexts.

A ciphertext only attack can only succeed if some information about the distribution of the plaintexts is available. In this section we consider two cases. In the first case we try to decrypt a message that consists of an ASCII-encoded text in English. We assume here that only characters are used with the most significant bit equal to zero. While this is not true if an extended set of ASCII characters is used, it is still a good approximation. The second case is much harder: we try to decrypt a message where the bytes have a uniform distribution, except for the fact that the most significant bit of every byte is zero. We will call this type of text 'random ASCII'. Our attack is very succesful in the first case, but in the last case we can only recover some of the key bits.

4.2.1 Recovering s^ρ

If the plaintext consists of ASCII characters, the most significant bit of every byte will be zero. Exoring of some bytes with the bytes of the keys z_2, z_3, z_6 and z_7 will keep these most significant bits constant. Even after the addition of z_1, z_4, z_5 and z_8 these bits will be still be heavily biased. By observing the ciphertexts it is possible to see where the almost constant bits have moved to. In this way s^ρ mod 8 can be determined. Computer experiments have shown that with 100 ciphertexts the success rate is close to 1. This attack applies with the same level of success to the case of random ASCII as to the case of English text.

# texts	English text	Random ASCII
100	0.98	0.98
1000	1.00	1.00

Table 1. Success probability for the recovery of s^ρ mod 8 on Akelarre when the plaintext is known to be English ASCII-coded text; and when the plaintext consists of random ASCII.

Once s^ρ is known modulo eight, there are four possibilities left modulo 32, which is all we need for the recovery of the z_i keys. The rest of the attack is simply repeated for every possible value. Note that the addition becomes less and less linear for more significant bits. This probably allows to determine s^ρ modulo 32 in a more efficient way than just guessing.

4.2.2 Recovering the z_i keys

Once the rotation modulo 32 is known, it can be partially compensated for by applying the inverse rotation to the ciphertexts and the keys of the output transformation. In the further analysis we assume that $s^\rho = 0$ mod 32 because this makes the discussion much easier to follow. Equation (5) can then be written as follows.

$$
\begin{aligned}
(x_v + z_v) \oplus x_{v_2} &= (y_1 - z_5) \oplus y_3 \oplus z_7 \oplus z_{v_2} && (9a)\\
(x_w + z_w) \oplus x_{w_2} &= (y_4 - z_8) \oplus y_2 \oplus z_6 \oplus z_{w_2} && (9b)
\end{aligned}
$$

The parameters (v, v_2, w, w_2) can take the values $(1,3,4,2)$ and $(4,2,1,3)$. The exact value depends on s^ρ. We assume that $(v, v_2, w, w_2) = (1,3,4,2)$. Since the attack uses only information on the distribution of the plaintexts and not on their actual value, it will also work if this assumption is wrong. The only visible effect will be that the keys have been labeled erroneously. When the keys are used to recover plaintext both possibilities should be checked.

The attack takes one equation of (9) at a time. Like with the known plaintext attack, we cannot determine z_3, z_2, z_7 and z_6 separately, but only $z_3 \oplus z_7$ and $z_2 \oplus z6$. This does not pose any hinder in the plaintext recovery phase. Also, we cannot determine the most significant bit of z_5, z_8, z_1 and z_4. Since the most significant bit of every plaintext byte is zero, it is possible to recover the keys byte by byte, starting with the least significant byte. In the remainder of the analysis the variables y_i, z_i, x_i will stand for the byte that is examined.

The attack is based on a statistical technique, called *Maximum Likelihood.* The first stage of the attack is off-line. The cryptanalyst uses the information he has on the distribution of the plaintext bytes x_1, x_3 to build for every possible value of z_1 a table $p_t[z_1]$ that contains the distribution of $(x_1 + z_1) \oplus x_3$. These distributions are called the *theoretical distributions*.

Afterwards the cryptanalyst collects ciphertexts and builds for every possible value of z_5 and $z_7 \oplus z_3$ a table $p_p[z_5, z_7 \oplus z_3]$ that contains the distribution of the values $(y_1 - z_5) \oplus y_3 \oplus z_7 \oplus z_3$. These tables give a set of experimentally measured distributions, called the *observed distribution*. The value of the keys can then be obtained by comparing every theoretical distribution $p_t[z_1]$ and every practical distribution $p_p[z_5, z_7 \oplus z_3]$. If enough ciphertexts are used, the closest match between a theoretical and an observed distribution will occur when the values for z_1, z_5 and $z_3 \oplus z_7$ are the correct key values.

This attack is very succesful if the message consists of English texts. Table 3 shows that 1000 to 5000 ciphertexts suffice to determine the correct key bytes with a very high probability. Using 1000 ciphertexts, we can recover the correct key byte in 80% of the cases, with 5000 ciphertexts this increases to 92%. The success of the attack becomes even more apparent if the key ranking technique [11] is used: not only the most probable key is given as output, but a small set of key values with high probability to contain the correct value. For instance, the probability that the correct key byte is one of the three most suggested values, equals 92% for 1000 ciphertexts, and 100% for 5000 ciphertexts. It is reasonable to assume that the probabilities of success will be the same for the attacks on the remaining key bytes. Thus, using 1000 ciphertexts, we estimate that the correct values of all four bytes of each of z_1, z_5 and $z_3 \oplus z_7$ will be amongst the $3^4 = 81$ most suggested values (out of 2^{32} values) with success probability $0.92^4 \simeq 0.72$.

# texts	most suggested	3 most suggested
1000	80%	92%
5000	92%	100%

Table 2. Success probability for the ciphertext only attack on Akelarre when the plaintext is known to be English ASCII-coded text.

For the case of random ASCII, the attack is much less successful. It is possible to narrow down the number of candidates for the key significantly, but the attack seldom succeeds in recovering the complete key. The fact that the number of candidates is reduced, can be seen as a determination of some key bits: we say that the attack finds t equivalent key bits if the number of candiates is divided by a factor of 2^t and the correct key value is still among the candidate values. The attack recovers two bytes at a time, one byte from each of $z_1 \oplus z_5$ and $z_3 \oplus z_7$. The highest order bit of $z_1 \oplus z_5$ cannot be determined uniquely, so the attack can recover at most 15 key bits at a time. Using 200 000 ciphertexts, we can recover 6 equivalent key bits with probability 99%, 13 equivalent key bits are recovered correctly with probability 23%.

# texts	number of equivalent key bits recovered (t)			
	13	11	8	6
50 000	2%	15%	48%	82%
100 000	10%	23%	65%	94%
200 000	23%	37%	72%	99%

Table 3. Success probability for the ciphertext only attack on Akelarre when the plaintext consists of random bytes, with the most significant bit set to zero.

When the keys have been recovered, the approach of the previous section can be used to recover plaintexts. However, note that the attacker must repeat this attack for both sets of values for (v, v_2, w, w_2) of Equation (5).

5 CONCLUSION

We presented realistic attacks on the block cipher Akelarre, which mixes features of the block cipher IDEA and RC5. Our attacks are independent of the number of rounds used in the cipher and enable the recovery of a limited set of key bits. Once these bits have been found an attacker can obtain the plaintexts of any intercepted ciphertexts, provided that the plaintext space is redundant. Akelarre and our attacks illustrate that mixing the components of two presumably secure block ciphers does not always yield a strong new block cipher.

REFERENCES

1. Alvarez, G., D. de la Guiaía, F. Montoya and A. Peinado. 1996. Akelarre: a new block cipher algorithm. *Proceedings of SAC'96, Third Annual Workshop on Selected Areas in Cryptography.* Kingston, Ontario, CANADA: Queen's University.

2. Biryukov, A. and E. Kushilevitz. 1998. Improved cryptanalysis of RC5. *Advances in Cryptology, Proceedings Eurocrypt'98, LNCS 1403,* K. Nyberg, Ed. New York: Springer-Verlag. pp. 85–99.

3. Borst, J., L. R. Knudsen, V. Rijmen. 1997. Two attacks on reduced IDEA. *Advances in Cryptology, Proceedings Eurocrypt '97, LNCS 1233,* W. Fumy, Ed. New York: Springer-Verlag. pp. 1–13.

4. Daemen, J., R. Govaerts and J. Vandewalle. 1994. Weak keys for IDEA. *Advances in Cryptology - Proceedings Crypto'93, LNCS 773,* D. Stinson, Ed. New York: Springer-Verlag. pp. 224–231.

5. Daemen, J., R. Govaerts and J. Vandewalle. 1994. Cryptanalysis of 2,5 rounds of IDEA. *Technical Report ESAT-COSIC Report 94-1,* Department of Electrical Engineering, Katholieke Universiteit Leuven. March.

6. Ferguson, N. and B. Schneier. 1997. Cryptanalysis of Akelarre. *Proceedings of the Workshop on Selected Areas in Cryptography - SAC'97, Ottawa, 11-12 Augustus.* pp. 201–212.

7. Kaliski, B. S. and Y.L. Yin. 1995. On differential and linear cryptanalysis of the RC5 encryption algorithm," *Advances in Cryptology, Proceedings Crypto '95, LNCS 963*, D. Coppersmith, Ed.. New York: Springer-Verlag. pp. 171–184.

8. Knudsen, L. R. and W. Meier. 1996. Improved differential attacks on RC5. *Advances in Cryptology, Proceedings Crypto '96, LNCS 1109.* N. Koblitz, Ed. New York: Springer-Verlag. pp. 216–228.

9. Knudsen, L. and V. Rijmen. 1997. Two Rights Sometimes Make a Wrong. *Proceedings of the Workshop on Selected Areas in Cryptography - SAC'97, Ottawa, 11-12 Augustus.* pp. 213–223.

10. Lai, X., J.L. Massey and S. Murphy. 1991. Markov ciphers and differential cryptanalysis. *Advances in Cryptology, Proceedings Eurocrypt'91, LNCS 547*, D.W. Davies, Ed. New York: Springer-Verlag. pp. 17–38.

11. Matsui, M. 1994. The first experimental cryptanalysis of the Data Encryption Standard. *Advances in Cryptology, Proceedings Crypto'94, LNCS 839.* Y. Desmedt, Ed. New York: Springer-Verlag. pp. 1–11.

12. Meier, W. 1994. On the security of the IDEA block cipher. *Advances in Cryptology - Proceedings Eurocrypt'93, LNCS 765*, T. Helleseth, Ed. New York: Springer-Verlag. pp. 371–385.

13. Rivest, R. L. 1995. The RC5 encryption algorithm. *Fast Software Encryption, LNCS 1008*, B. Preneel, Ed. New York: Springer-Verlag. pp. 86–96.

BIOGRAPHICAL SKETCHES

Lars Knudsen was born in Denmark on February 21, 1962. He received his MSc and Phd degrees in Computer Science and Mathemathics in 1992 and 1994, both from the University of Aarhus, Denmark. Since 1998 he is an associate professor at the University of Bergen, Norway.

Vincent Rijmen was born in Belgium on October 16, 1970. In 1993, he obtained the degree of electronics engineer at the K. U. Leuven (Belgium), and in 1997 he obtained his Phd. Since 1993, Vincent has been working at the department of electrotechnic engineering of the K. U. Leuven. Currently, Vincent is a postdoctoral researcher, sponsored by the Fund for Scientific Research - Flanders (Belgium).