Two Rights Sometimes Make a Wrong

Article · September 1997
Source: CiteSeer

2 authors:

Lars Ramkilde Knudsen
Technical University of Denmark
184 PUBLICATIONS **6,910** CITATIONS
SEE PROFILE

Vincent Rijmen
University of Leuven
220 PUBLICATIONS **10,273** CITATIONS
SEE PROFILE

Two Rights Sometimes Make a Wrong

Lars R. Knudsen Vincent Rijmen*

Katholieke Universiteit Leuven, ESAT-COSIC
K. Mercierlaan 94, B-3001 Heverlee, Belgium
lars.knudsen@esat.kuleuven.ac.be
vincent.rijmen@esat.kuleuven.ac.be

Abstract

At the SAC'96 a new 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.

1 Introduction

At the SAC'96 a new block cipher, Akelarre, was proposed [1]. Akelarre is an iterated cipher, which uses components of the block ciphers RC5[4] and IDEA[2]. 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 organised as follows. § 2 contains a short description of Akelarre with the details necessary for our attacks; we refer to [1] for the full description. In § 3 we describe the main weakness of Akelarre, which forms the basis of our attacks. In § 4 a known plaintext attack and a ciphertext only attack are given and § 5 contains our concluding remarks.

2 Description of Akelarre

Akelarre [1] is a 128-bit block cipher. The key length is variable, but always a multiple of 64 bits. Akelarre has a structure similar to IDEA [2]. The main differences are the following.

- Akelarre uses 32-bit words instead of 16-bit words.
- The multiplication-addition structure 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.

*F.W.O. research assistant, sponsored by the Fund for Scientific Research - Flanders (Belgium)

Appeared in *Selected Areas in Cryptography, 4th Annual International Workshop, SAC 1997*, Lecture Notes in Computer Science, Springer-Verlag, pp. 213–223, 1997.

- 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 [4] round operation. Akelarre has an input transformation, an output transformation and a variable number of rounds. It is proposed with four rounds. Figure 1 shows 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.

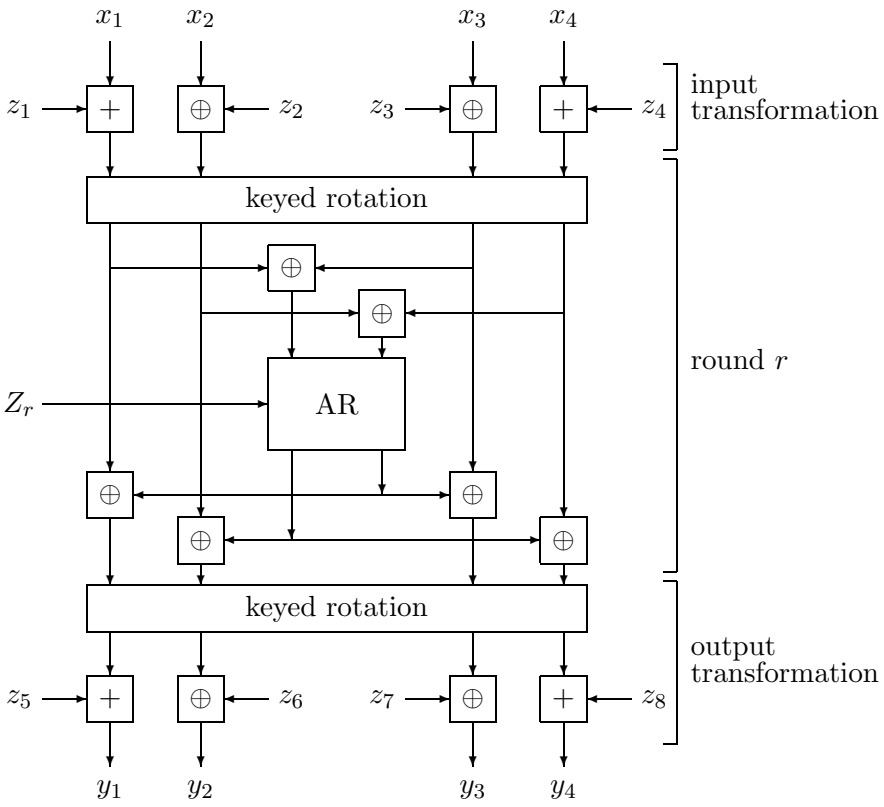

Figure 1: Computational graph of Akelarre.

The plaintext is denoted $x_1 \,\|\, x_2 \,\|\, x_3 \,\|\, x_4$. where '$\|$' denotes concatenation of bit strings. The input transformation consists of adding modulo 2^{32} respectively exoring the key words $z_1, \ldots, z_4$ to the plaintext, according to Figure 1. The input of round r will be denoted $x_1^r \,\|\, x_2^r \,\|\, x_3^r \,\|\, x_4^r, r = 1, \ldots R$, The round transformation for round i is as follows.

$$u_1^i \,\|\, u_2^i \,\|\, u_3^i \,\|\, u_4^i = \mathrm{rot}_{r^i}(x_1^i \,\|\, x_2^i \,\|\, x_3^i \,\|\, x_4^i) \quad (1)$$

$$(t_1, t_2) = \mathrm{AR}(u_1^i \oplus u_3^i, u_2^i \oplus u_4^i) \quad (2)$$

$$x_1^{i+1} = u_1^i \oplus t_1 \quad (3a)$$

$$x_2^{i+1} = u_2^i \oplus t_2 \quad (3b)$$

$$x_3^{i+1} = u_3^i \oplus t_1 \quad (3c)$$

$$x_4^{i+1} = u_4^i \oplus t_2 \quad (3d)$$

Appeared in *Selected Areas in Cryptography, 4th Annual International Workshop, SAC 1997*, Lecture Notes in Computer Science, Springer-Verlag, pp. 213–223, 1997.

where rot_{r^i} is a 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 $z_5, \ldots, z_8$.

3 Weakness of the Round Transformation

In this section we describe the main observation to be used in our attacks. The round transformation of Akelarre exhibits an invariant relation between the input and the output.

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

After R rounds this is

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

where $s^R = (\sum_{i=1}^{R} r^i) \bmod 64$. The input and output transformation destroy this invariant. Let $s^\rho = s^R + r^{R+1}$, where the last term denotes the rotation amount in the output transformation. For the whole block cipher the relation is

$$\begin{aligned}&((y_1 - z_5) \oplus y_3 \oplus z_7) \,\|\, ((y_4 - z_8) \oplus y_2 \oplus z_6) \\ &\quad = \mathrm{rot}_{s^\rho}(((x_1 + z_1) \oplus x_3 \oplus z_3) \,\|\, ((x_4 + z_4) \oplus x_2 \oplus z_2))\,, \end{aligned} \tag{4}$$

If the keys $z_i, i = 1, \ldots 8$ and s^ρ are known, it is possible to calculate the exor of two halves of any plaintext block from the exor of the two halves of the corresponding ciphertext block. 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 (4). 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

4.1.1 Recovering the keys

Equation (4) is an equation in 64 bits, containing one unknown rotation and eight unknown 32-bit key words. Five known plaintexts and their ciphertexts give enough information to solve the equations for the unknown key bits. This can be done very efficiently by guessing a value for s^R and then solve for the z_i's, starting from the least significant bits. Four known

Appeared in *Selected Areas in Cryptography, 4th Annual International Workshop, SAC 1997*, Lecture Notes in Computer Science, Springer-Verlag, pp. 213–223, 1997.

plaintexts are used to solve the equations and the fifth to verify. Since s^R can take only 128 different values, the work factor of this approach is very small.

The keys z_1, z_4, z_5, z_8 and s^R can be determined uniquely. Since only exor information is available, it is not possible to determine z_2, z_3, z_6 and z_7 uniquely. Depending on the value of s^R it is possible to determine $z_2 \oplus z_6$ and $z_3 \oplus z_7$, or $z_2 \oplus z_7$ and $z_3 \oplus z_6$.

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 $s^R = 0$. Equation (4) is then:

$$(x_1 + z_1) \oplus x_3 = (y_1 - z_5) \oplus y_3 \oplus z_7 \oplus z_3 \tag{5a}$$
$$(x_4 + z_4) \oplus x_2 = (y_4 - z_8) \oplus y_2 \oplus z_6 \oplus z_2\,. \tag{5b}$$

The right hand sides of (5) 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 complex, 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 a and b can take every value. 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

It is possible to recover the eight key words z_i and s^R using only statistical information on the distribution of the plaintext. Afterwards the approach of the previous section can be used to recover plaintexts.

4.2.1 Recovering s^R

If the plaintext consists of ASCII characters, the most significant bit of every byte will be zero (or with probability close to one for an extended set of ASCII characters). In the following we assume that the most significant bit of every byte is a zero bit. 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 biased with a high probability. By observing the ciphertexts it is possible to see where the almost constant bits have moved to. In this way $s^R \bmod 8$ can be determined. Computer experiments have shown that with 5000 ciphertexts the success rate is close to 1.

Once s^R is known modulo eight, there are four possibilities left modulo 32. 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^R modulo 32 in a more efficient way than just guessing.

Appeared in *Selected Areas in Cryptography, 4th Annual International Workshop, SAC 1997*, Lecture Notes in Computer Science, Springer-Verlag, pp. 213–223, 1997.

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 $s^R = 0 \bmod 32$ because this makes the discussion much easier to follow. Equation (4) can be written as follows.

$$(y_1 - z_5) \oplus y_3 \oplus z_7 = (x_v + z_v) \oplus x_{v_2} \oplus z_{v_2} \quad (6a)$$

$$(y_4 - z_8) \oplus y_2 \oplus z_6 = (x_w + z_w) \oplus x_{w_2} \oplus z_{w_2} \quad (6b)$$

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^R. 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 (6) at a time. We assume here that the plaintext consists of bytes with the most significant bit equal to 0. In this case 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 cryptanalyst uses the information he has on the distribution of the plaintext bytes to build an off-line table that contains for every value of z_1 the distribution of $(x_1+z_1)\oplus x_3$. These distributions are called the theoretical distributions. Afterwards the cryptanalyst collects ciphertexts and calculates for every possible value of z_5 and $z_7 \oplus z_3$ the values $(y_1 - z_5) \oplus y_3 \oplus z_7 \oplus z_3$. This gives a set of experimentally measured distributions. If enough ciphertexts are used the correct values for z_1, z_5 and $z_3 \oplus z_7$ will yield the closest match between a theoretical and an experimental distribution.

The number of required ciphertexts depends on the distribution of the plaintexts. We did experiments with two distributions: real English text and random bytes (with the most significant bit set to zero). The ciphertext requirements can be reduced significantly if the key ranking technique [3] 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 the case of English text, using 1000 ciphertexts, $s^R \bmod 8$ can be determined with probability 0.7 and the correct values for one byte of each of z_1, z_5, and $z_3 \oplus z_7$ are in 90% of the cases among the 4 most suggested values (out of 2^{24} possible values). It is reasonable to assume that the probability of successes will be the same for the attacks on the remaining key bytes. Thus, for the case of English text with 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 $4^4 = 256$ most suggested values (out of 2^{96} values) with success probability $0.9^4 \simeq 0.66$. The results are summarized in Table 1. As can be seen the recovery of the keyed rotation works better for random ASCII bytes than for English text, whereas the recovery of the keys z_i works worse. We expect that success probability of the key recovering part of our attack will increase significantly when using more texts.

Subsequently, 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 6.

Appeared in *Selected Areas in Cryptography, 4th Annual International Workshop, SAC 1997*, Lecture Notes in Computer Science, Springer-Verlag, pp. 213–223, 1997.

	# texts	recovering $s^R \bmod 8$	recovering z_i
English text	1000	0.70	0.66
Random bytes	1000	0.90	0.00
English text	5000	0.78	1.00
Random bytes	5000	1.00	0.03

Table 1: Success probability for the ciphertext only attack on Akelarre when the plaintext is known to be English ASCII-coded text; and when the plaintext consists of random bytes, with the most significant bit set to zero.

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

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

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

[4] R.L. Rivest, "The RC5 encryption algorithm," *Fast Software Encryption (FSE'94), LNCS 1008*, B. Preneel, Ed., Springer-Verlag, 1995, pp. 86–96.

Appeared in *Selected Areas in Cryptography, 4th Annual International Workshop, SAC 1997*, Lecture Notes in Computer Science, Springer-Verlag, pp. 213–223, 1997.