MacDES: a new MAC algorithm based on DES

L.R. Knudsen and B. Preneel

February 13, 1998

Indexing terms: Cryptography, Message Authentication.

The authors propose a new MAC algorithm which improves the popular retail MAC based on DES; it has the same complexity, but provides better resistance against key recovery attacks. In addition, a new key recovery attack on the retail MAC is presented, that requires a single known text-MAC pair and 2^{56} on-line MAC verifications.

Introduction: Message authentication code (MAC) algorithms are widely used to provide data integrity and data origin authentication. They are preferred over digital signatures for applications with inexpensive processors, such as smart cards, or involving high data rates, such as IP level security. MACs are used in settings where sender and receiver share a secret key (uppercase) K, of bitlength (lowercase) k. The sender sends together with the message x an m-bit string $\mathrm{MAC}_K(x)$, that is a complex and non-invertible function of every bit of K and x. Typically m is between 32 and 64 bits. The receiver can verify that the message x indeed comes from the claimed sender by recomputing the value $\mathrm{MAC}_K(x)$, and by verifying it against the transmitted MAC value. An active eavesdropper can modify the message, but faces the task of determining the MAC value without knowing K.

The main security property for a MAC is that it should be resistant to *forgery*, i.e., it must be computationally infeasible for someone who does not know the secret key to find an arbitrary *new* message x and the corresponding value $\mathrm{MAC}_K(x)$. One strategy is to guess the MAC value; it has success probability $1/2^m$, which means that it can be defeated easily by choosing m large enough. A more clever strategy consists of asking the MAC value for a number of chosen texts, or of verifying a number of text-MAC values before coming up with a forgery that is correct with high probability.

A second attack on a MAC is a *key recovery* attack; recovering the key allows for an arbitrary forgery. One approach is to search the key space exhaustively. This requires about k/m text MAC pairs (to define the key uniquely), and 2^{k-1} MAC evaluations. It can be precluded by choosing k appropriately; 80 to 90 bits should be sufficient for long term security.

Appeared in *Electronics Letters 34(9)*, pp. 871–873, 1998.

CBC-MAC: The standard MAC algorithm for banking applications is CBC-MAC [1, 2, 3] based on DES [4]. The block length n and key length k of CBC-MAC are equal to block and key length of the block cipher on which it is based ($n = 64$ and $k = 56$ for DES). The input is padded unambiguously to a multiple of the block length, and then divided into t blocks x_1 through x_t. The following iteration is performed:

$$H_i = E_K(H_{i-1} \oplus x_i)\,, \quad 1 \le i \le t\,.$$

Here $E_K(x)$ denotes the encryption of x using key K with an n-bit block cipher E and $H_0 = 0$. The MAC value is then computed as $\mathrm{MAC}_K(x) = g(H_t)$, where g is the output transformation. The mapping g is intended to preclude a simple forgery attack (see e.g., [5]). One approach is for g to select the leftmost m bits, but it was shown in [6] that this is less secure than expected.

The ANSI retail MAC [1] selects as output transformation a decryption with a second key K_2 followed by an encryption with K_1 (such that the last block undergoes a two-key triple encryption):

$$g(H_t) = E_{K_1}(D_{K_2}(H_t)) = E_{K_1}(D_{K_2}(E_{K_1}(x_t \oplus H_{t-1})))\,.$$

Here D denotes decryption. This alternative is widely used because it requires very little overhead (2 encryptions), and offers the additional advantage that it precludes an exhaustive search against the 56-bit DES key [7]. With a 112-bit key, one can expect that the retail MAC based on DES is resistant to key recovery attacks.

The best known forgery attack on CBC-MAC based on DES requires about $2^{32.5}$ known text-MAC pairs and a single chosen text if $m = 64$. For $m = 32$, an additional 2^{32} chosen texts are required [5]. In [8] a divide and conquer key recovery attack against the retail MAC is described. This attack requires $2^{32.5}$ known text-MAC pairs and $3 \cdot 2^{56}$ off-line computations to find the 112-bit key.

New attack: This section presents a different divide and conquer key recovery attack on the retail MAC, which uses on-line verifications rather than known-text MAC pairs. The attack requires only a single known text-MAC pair, 2^{55} MAC verifications, and about 2^{56} encryptions when DES is used as the underlying block cipher. These numbers are much smaller than what is suggested by the key size of 112 bits and the size of the MAC result of 64 bits. In some environments this attack is more realistic than the attack of [8]: when the MAC generation is performed on a slow smart card, $2^{32.5}$ known texts are out of reach, while typically the MAC verifications are performed centrally on extremely fast and parallel machines. Moreover, its success probability is linear in the number of verifications, while it is quadratic in the number of known text-MAC pairs for the attack of [8].

Proposition 1 *For the retail MAC [1, 3], a key recovery attack yielding both keys K_1 and K_2 requires one known text of t blocks ($t \ge 2$), $1.5 \cdot 2^k$ encryptions, and 2^{k-1} MAC verifications, where $k = |K_1| = |K_2|$, $k \le n$, and $m = n$.*

Appeared in *Electronics Letters 34(9)*, pp. 871–873, 1998.

Proof: The attacker knows a padded message $x = (x_1, x_2, x_3, \ldots, x_t)$ and its corresponding MAC value Y. He creates a second message of the form $x' = (x'_1, x'_2, x_3, \ldots, x_t)$. Then he chooses a value $x'_1 \neq x_1$, guesses the value of K_1, and computes

$$x'_2 = x_2 \oplus E_{K_1}(x_1) \oplus E_{K_1}(x'_1)\,.$$

If the guess for K_1 was correct, this choice implies $H_2 = H'_2$, and thus $\mathrm{MAC}_K(x') = Y$. Hence if the attacker submits the pair (x', Y) for verification, and if the guess was correct, the verification will result in a positive answer. If the guess was incorrect, the pair (x', Y) will only pass the verification with probability $1/2^m$. On the average, K_1 will be found after 2^{k-1} attempts; each attempt requires 2 encryptions and 1 MAC verification. Subsequently, K_2 is computed by exhaustive search, which requires on the average 2^{k-1} encryptions. If a spurious key K_1 arises, it can be eliminated by either exhaustively searching all values of K_2 or by confirming the guess for K_1 with a different choice for x'_1 (and x'_2). ∎

Note that the attack allows for the verification of individual guesses of K_1, which by itself is an undesirable property. Proposition 1 can be generalized to cover the cases $k > n$ and $m < n$.

New construction (MacDES) and its security: An alternative construction is proposed, which requires exactly the same complexity as the retail MAC, but which provides higher security against key recovery attacks. The idea is to start from CBC-MAC and to replace single DES by double DES in the first and in the last iteration (of course any other block cipher may be used). In addition, the padded message should have at least two blocks, or $t \geq 2$. Define $H_1 = E_{K_2}(E_{K_1}(x_1))$, and

$$H_i = E_{K_1}(H_{i-1} \oplus x_i)\,,\;\; 2 \leq i \leq t\,,$$

then the MAC result is computed as $\mathrm{MAC}_K(x) = E_{K'_2}(H_t)$ (see also Figure 1). Here K'_2 is a key derived from K_2 such that $K'_2 \neq K_2$; e.g., $K'_2 = K_2 \oplus \beta$, where β is a non-zero k-bit string. Optionally, the output can be truncated to $m < 64$ bits. The best known attacks on MacDES are the forgery attack of [5], a brute force key search, which requires 2 known text-MAC pairs and 2^{112} encryptions, and a key recovery attack that requires 2^{65} chosen text-MAC pairs, 2^{89} encryptions, 2^{55} MAC verifications, and storage of about 2^{32} 24-byte values. Both key recovery attacks are currently completely unrealistic. The second key recovery attack is described in Proposition 2.

Proposition 2 *For MacDES, there exists a key recovery attack yielding both keys K_1 and K_2 that requires 2^{n+1} chosen texts of 3 blocks each, $2^{k+n/2+1}$ encryptions, 2^{k-1} MAC verifications, and $3 \cdot 2^{n/2}$ n-bit words of storage, where $k = |K_1| = |K_2|$, $k \leq n$, and $m = n$.*

Proof: The chosen texts consists of three blocks, that is,

$$\mathrm{MAC}(x_1, x_2, x_3) = E_{K'_2}(E_{K_1}(E_{K_1}(E_{K_2}(E_{K_1}(x_1)) \oplus x_2) \oplus x_3))\,.$$

Appeared in *Electronics Letters 34(9)*, pp. 871–873, 1998.

First fix an n-bit string $\alpha \neq 0$, then choose X : a set of $2^{n/2}$ texts $X(i) = (x_1, x_2, x_3(i))$ and X' : a set $2^{n/2}$ texts $X'(j) = (x_1, x_2 \oplus \alpha, x'_3(j))$ for $i, j = 1, \ldots, 2^{n/2}$, and get the corresponding MACs; x_1 and x_2 are fixed values. The values of $x_3(i)$ and $x'_3(j)$ are chosen such that for any n-bit value β, there exist i, j such that $x_3(i) \oplus x'_3(j) = \beta$ [6]. It follows that by pairing each element in X with each element in X', one finds exactly one pair of texts $(X(k), X'(l))$ with equal MACs. For the middle encryption with key K_1 this gives the information that an input pair with difference α yields an output pair with difference $x_3(i) \oplus x'_3(j)$. Store this difference together with the values of x_2, $x_3(i)$, and the MAC in a table T. Repeat this part of the attack $2^{n/2}$ times with the same value for x_1, but different values of x_2 (choosing x_3-values as before). At this point one knows that $2^{n/2}$ input pairs with difference α result after a single encryption with K_1 in output pairs with known differences (included in T). For each value of K_1, choose $2^{n/2}$ input pairs with difference α and compute the differences in the outputs. With a probability of 0.63 the difference in an output pair equals one of the differences from table T, say in entry w. (If no such value is found, some additional input pairs can be chosen.) Compute the exact values in the input and output of the middle encryption from the values (a, b) of $(x_2, x_3(i))$ in entry w of T. As input and output values are now known, the value for K_1 can be verified similarly as in the proof of Proposition 1. Once K_1 has been identified, K_2 can be found by exhaustive search, which requires on the average 2^{k-1} encryptions. ∎

The motivation for the use of K'_2 for the last encryption rather than K_2 is the following certificational attack. Select 2^n chosen texts of 2 blocks, where x_1 takes all 2^n values and $x_2 = 0$. With probability 0.63, the permutation $E_{K_2}(E_{K_1}(.))$ has a fixed point, denoted with f. For such a value f one obtains also $\mathrm{MAC}(f) = f$. This results in a plaintext/ciphertext pair for a double encryption; this implies that key recovery requires $1.5 \cdot 2^k$ encryptions and 2^k storage of n-bit values; other trade-offs are possible [9].

Conclusions: This letter demonstrates a new key recovery attack for the ANSI X9.19 retail MAC based on on-line MAC verifications. An alternative scheme is proposed, which provides increased strength against key recovery attacks at the same computational cost.

L.R. Knudsen (Department of Informatics, University of Bergen, N-5020 Bergen, Norway)

B. Preneel[1] (Katholieke Universiteit Leuven, Department Electrical Engineering-ESAT, COSIC, Kardinaal Mercierlaan 94, B–3001 Heverlee, Belgium)

References

[1] ANSI X9.19: 'Financial institution retail message authentication' (American Bankers Association, August 13, 1986)

[1]F.W.O. postdoctoral researcher, sponsored by the National Fund for Scientific Research – Flanders (Belgium). Part of this work was done while visiting the University of Bergen, Norway.

Appeared in *Electronics Letters 34(9)*, pp. 871–873, 1998.

[2] ISO 8731: 'Banking – approved algorithms for message authentication, Part 1, DEA, Part 2, Message authentication algorithm (MAA)' (ISO, 1987)

[3] ISO/IEC 9797: 'Information technology - Data cryptographic techniques - Data integrity mechanisms using a cryptographic check function employing a block cipher algorithm' (ISO/IEC, 1993)

[4] FIPS 46: 'Data encryption standard' (NBS, U.S. Department of Commerce, January 1977)

[5] PRENEEL, B. and VAN OORSCHOT, P.C.: 'MDx-MAC and building fast MACs from hash functions', Advances in Cryptology, CRYPTO'95, Lect. Notes Comput. Sci. 963, (Springer-Verlag, 1995), pp. 1–14

[6] KNUDSEN, L.R.: 'A chosen text attack on CBC-MAC', *Electron. Lett.*, 1997, **33**, (1), pp. 48–49

[7] WIENER, M.J.: 'Efficient DES key search', Technical Report TR-244, School of Computer Science, Carleton University, Ottawa, Canada, May 1994

[8] PRENEEL, B and VAN OORSCHOT, P.C.: 'A key recovery attack on the ANSI X9.19 retail MAC', *Electron. Lett.*, 1996, **32**, (17), pp. 1568–1569

[9] VAN OORSCHOT, P.C. and WIENER, M.: 'Improving implementable meet-in-the-middle attacks by orders of magnitude', Advances in Cryptology, CRYPTO'96, Lect. Notes Comput. Sci. 1109, (Springer-Verlag, 1996), pp. 229–236

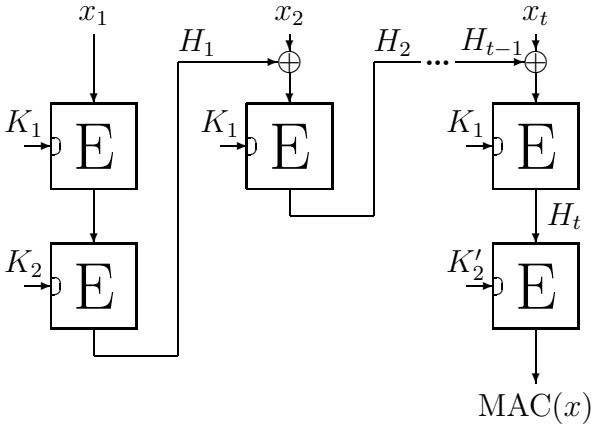

Figure 1: MacDES: a new MAC proposal based on DES.

Appeared in *Electronics Letters 34(9)*, pp. 871–873, 1998.