# New attacks against standardized MACs

Antoine Joux[1], Guillaume Poupard[1], and Jacques Stern[2]

[1] DCSSI Crypto Lab
51 Boulevard de La Tour-Maubourg
75700 Paris 07 SP, France
Antoine.Joux@m4x.org
Guillaume.Poupard@m4x.org
[2] Département d'Informatique
Ecole normale supérieure
45 rue d'Ulm, 75230 Paris Cedex 05, France
Jacques.Stern@ens.fr

**Abstract.** In this paper, we revisit the security of several message authentication code (MAC) algorithms based on block ciphers, when instantiated with 64-bit block ciphers such as DES. We essentially focus on algorithms that were proposed in the norm ISO/IEC 9797–1. We consider both forgery attacks and key recovery attacks. Our results improve upon the previously known attacks and show that all algorithms but one proposed in this norm can be broken by obtaining at most about $2^{33}$ MACs of chosen messages and performing an exhaustive search of the same order as the search for a single DES key.

## 1 Introduction

Message authentication codes (MACs) are secret-key cryptographic primitives designed to ensure the integrity of messages sent between users sharing common keys. MAC algorithms are often based on block ciphers or hash functions. Among the MAC algorithms based on block cipher, the CBC-MAC construction is probably the best known and studied. Initially proposed in [2], it has been studied in many papers, both from the cryptanalytic point of view [12] and from the security point of view [3].

It is well known that this algorithm suffers from birthday paradox based weaknesses, and this fact is reflected both in the known attacks and in the security proofs for this mode of operation of block ciphers. Of course, with 64-bit block ciphers, the birthday paradox hits as soon as $2^{32}$ messages have been authenticated with the same key. With high speed networks and high bandwidth applications, this is clearly not enough. In order to reach higher security, it is possible to use block ciphers with larger blocks, such as the AES, or more complicated MAC mechanisms secure beyond the birthday paradox, such as for example RMAC [11].

However, in many real life applications, developers still use variants of the CBC-MAC such as the algorithms described in ISO/IEC 9797–1 [7]. Of course, none of these algorithms has a known security proof that holds beyond the birthday paradox barrier. Moreover, most of them have known forgery attack with $2^{32}$ known or chosen messages. Yet, these forgery attacks are often seen as academic and impractical, and most developers only care about key recovery attacks. As long as finding the keys requires an exhaustive search of two or more DES keys simultaneously, i.e. as long as the 56-bit DES keys cannot be found one by one, the algorithm is deemed secure. According to this point of view, some algorithms in the norm are still insecure, but others are considered as secure against key recovery attacks. More precisely, according to the informative annex B of ISO/IEC 9797–1, among the 6 MAC algorithms proposed in this standard, the first 3 algorithms have known efficient key recovery attacks, while the 3 others are considered to be secure in that sense. However, Coppersmith, Knudsen and Mitchell [6, 5] have proved that, whatever the padding method may be, the fourth algorithm is also insecure.

In this paper, we show that the fifth algorithm can also be cryptanalysed with efficient key recovery attacks. This algorithm consists in two parallel computations of CBC-MAC. As a consequence, both for efficiency and for security reasons, it is much preferable to use a classical CBC-MAC with retail using a 128-bit block cipher such as AES and, if needed, to truncate the MAC value to 64 bits. We also describe generic key recovery attacks against any MAC based on a single internal CBC chain.

## 2 Description of CBC–MAC and some variants

In this section, we give a general description of MAC algorithm based on a single CBC chain with a key $K$ and quite general and arbitrary initial and final keyed transformation. For the sake of simplicity, we assume that the initial computation $I$ is applied to the first block of message and yields the initial value of the CBC chain which starts at the second block. We also assume that the final computation $F$ is applied to the final value in the CBC chain and results in the MAC tag $m$. Since all MAC algorithms based on a single CBC chain we are aware of, are of this type, we do not lose much generality. Furthermore, this formalism is also used in ISO/IEC 9797–1.

More precisely, for a message $M_1$, $M_2$,..., $M_\ell$ the computation works as follows (see also figure 1):

- Let $C_1 = I(M_1)$.
- For $i$ from 2 to $\ell$, let $C_i = E_K(C_{i-1} \oplus M_i)$.
- Let the MAC tag $m$ be $F(C_\ell)$.

In order to fully specify a MAC algorithm of this type, it suffices to give explicit descriptions of $I$ and $F$. The simplest example of a plain CBC-MAC occurs when $I$ is defined as $E_K$ and $F$ is the identity function. ISO/IEC 9797–1 standard defines 6 MAC algorithms. The first 4 ones are defined in the following table:

| Algorithm ISO/IEC 9797–1 | initial transformation | final transformation | reference |
|---|---|---|---|
| 1 | $E_K$ | $\emptyset$ | [7, 10, 2] |
| 2 | $E_K$ | $E_{K_1}$ | [7] |
| 3 | $E_K$ | $E_K \circ D_{K_1}$ | [7, 1] |
| 4 | $E_{K_2} \circ E_K$ | $E_{K_1}$ | [7, 9] |

The MAC algorithm 5 is defined as the exclusive-or of two MAC values computed using algorithm 1 with different keys. The MAC algorithm 6 is similar but uses the exclusive-or of two MACs computed with algorithm 4.

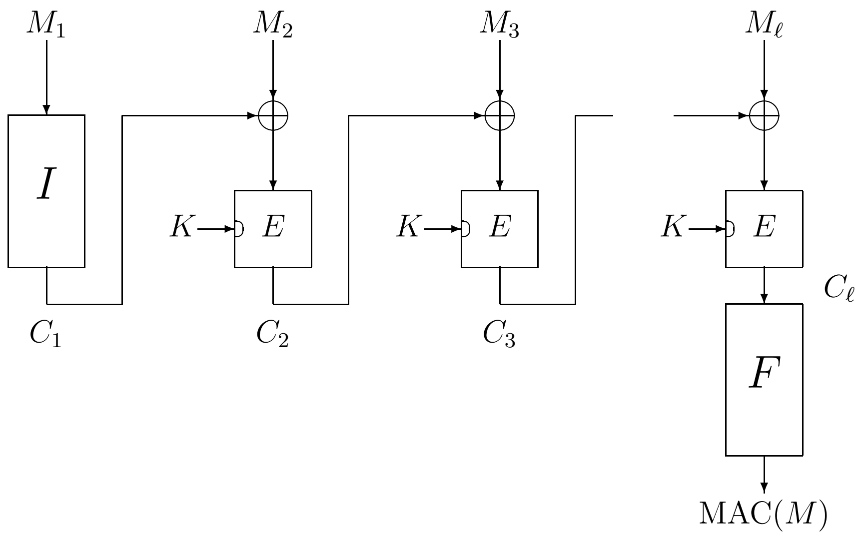


**Fig. 1.** Generic CBC MAC algorithm.

Even if the algorithms of [7] are defined with up to 6 secret keys, it is advised to derive them from only one or two keys. In the following, we propose attacks that do not try to take advantage of such key derivation technique.

It should also be noticed that ISO/IEC 9797–1 defines a complete MAC algorithm by specifying which padding should be used and if a final truncation is applied to the result. Our attacks immediately apply to standard paddings but we do not consider messages that include the bit length as a first block (padding 3 of [7]).

## 3 Overview of classical attacks

| Algorithm | complexity$^{(*)}$ | reference |
|---|---|---|
| algorithm 1 | [$2^{56}$,1,0,0] | |
| algorithms 2 and 3 | [$3 \times 2^{56}$,$2^{32}$,0,0] | |
| algorithm 4<br>with paddings 1 et 2 | [$4 \times 2^{56}$,$2^{32}$,2,0]<br>or [$4 \times 2^{56}$,1,1,$2^{56}$] | [6] |
| algorithm 4<br>with padding 3 | [$4 \times 2^{56}$,0,$2^{64}$,$2^{64}$]<br>or [$8 \times 2^{56}$,$2 \times 2^{32}$,$3 \times 2^{48}$,0] | [5] |

$^{(*)}$ an $[a, b, c, d]$ attack requires

- $a$ off-line block cipher encipherments,
- $b$ known data string/MAC pairs,
- $c$ chosen data string/MAC pairs,
- $d$ on-line MAC verifications.

**Fig. 2.** Complexity of key recovery attacks against ISO/IEC 9797–1 MAC algorithms.

### 3.1 Birthday paradox forgery of any generic CBC MAC algorithm

Let us assume that the final transformation $F$ of a generic CBC MAC algorithm is a permutation. Then, we observe MAC tags of known messages. If we denote by $n$ the block size (which is also the size of the MAC tag), according to the birthday paradox, the observation of $O(2^{n/2})$ MAC tags allows to find a collision, i.e., two different messages $M$ and $M'$ with the same MAC. For a 64-bit block cipher such as DES or triple-DES, this means that a collision occurs after the computation of only about $2^{32}$ MAC tags.

More precisely, we note the blocks of messages $M$ and $M'$ in the following way

$$\begin{aligned} M &= (M_1, M_2, \ldots, M_{\ell_1}, N_1, \ldots, N_{\ell_2}) \\ M' &= (M'_1, M'_2, \ldots, M'_{\ell'_1}, N_1 \ \ldots, N_{\ell_2}) \end{aligned}$$

with $M_{\ell_1} \neq M'_{\ell'_1}$. Then, it is easy to check that, for any blocks $N'_1, \ldots N'_{\ell'_2}$,

$$\mathrm{MAC}(M_1, M_2, \ldots, M_{\ell_1}, N'_1, \ldots, N'_{\ell'_2}) = \mathrm{MAC}(M'_1, M'_2, \ldots M'_{\ell'_1}, N'_1, \ldots, N'_{\ell'_2}).$$

Consequently, we obtain a forgery attack where the query of the MAC tag of a message enables to compute the (same) MAC tag of a different message.

Furthermore, using the same notations, we can also notice the following identity that will be used in the sequel:

$$\begin{aligned} &\mathrm{MAC}(M_1, M_2, \ldots, M_{\ell_1 - 1}, X, N'_1, \ldots, N'_{\ell'_2}) \\ &= \mathrm{MAC}(M'_1, M'_2, \ldots, M'_{\ell'_1 - 1}, X \oplus M_{\ell_1} \oplus M'_{\ell'_1}, N'_1, \ldots, N'_{\ell'_2}). \end{aligned}$$

In conclusion, independently of the key size and of the complexity of initial and final transformations $I$ and $F$, CBC MAC algorithms are vulnerable to forgery attacks if the block size is too small. However, such attacks are often seen as academic and impractical by developers. That is why, in the following, we mainly focus on key recovery attacks.

### 3.2 Attacking algorithm 1 from ISO/IEC 9797–1.

The first and simplest MAC algorithm, that has also been standardized by NIST [10], can be attacked in many different ways. Used with the simple DES block cipher, the knowledge of one MAC tag allows, through an exhaustive search on the key $K$, to recover this key. Furthermore, the algorithm does not have any final "retail" so it is easy to obtain valid MAC tags of concatenated messages using a so-called "xor-forgery".

### 3.3 Attacking algorithms 2 and 3 from ISO/IEC 9797–1.

If the initial transformation is a single application of the block cipher, as in algorithms 2 and 3, and if the final transformation is a permutation, the observation of a collision allows to recover the key $K$ as in the case of algorithm 1. The attack [12] goes as follows. Observe MAC tags of known messages of at least two blocks until a collision occurs. Using the birthday paradox, such an event should appear after the observation of $O(2^{n/2})$ messages, where $n$ is the block size. Since $F$ is a permutation, a collision is also present at the end of the CBC chain so we obtain a test for the exhaustive search on $K$.

When using DES, we need the observation of $2^{32}$ MAC values for known messages followed by an exhaustive search of a single DES key. Then, the key $K_1$ used in $F$ can be recovered by another exhaustive search.

## 4 Devising some tools

Throughout this section, we assume that $n$ denotes the block size used in the MAC algorithms, or equivalently in the basic block cipher $E$ we rely on. We are mostly interested in the case were $n$ is 64 bits. The attacks presented in this paper mostly rely on techniques that allows us to learn the exclusive-or of two intermediate values present in two of the core CBC chains.

### 4.1 Exclusive-Or of intermediate values in CBC chains

We first remind a technique of Coppersmith and Mitchell [6]. We assume that we are given any generic message authentication code algorithm based on a single CBC computation chain with block cipher $E$ and key $K$, as defined in section 2. Clearly, by fixing the last blocks of the message, we transform the final computation into a function of the final output of the CBC chain. In many cases, such as algorithms 1 to 4 of ISO/IEC 9797–1 without final truncation, this function is in fact a permutation and we are able to learn whether the outputs of two different chains are equal or not.

Note that this hypothesis is not essential. Indeed, if the output function is not a permutation, it suffices to observe the output of several computation pairs done with different final blocks. When the output of the two chains are equal, all pairs will contain two identical values whatever the final transformation may be. As a consequence, we may ignore the final computation and assume that we can learn whether the output of two chains are equal or not.

Let $M$ and $N$ be two messages of respective length $\ell_M$ and $\ell_N$. Let $C_{\ell_M}$ denote the final value of the CBC chain computed for message $M$ and $D_{\ell_N}$ denote the final value of chain computed for message $N$. Now form a message $M^{(T)}$ by adding a single block $T$, right at the end of the CBC chain for message $M$. Likewise, add a single block $U$ at the end of $N$ and form $N^{(U)}$. Writing down the equations of the two CBC chains, we find that the final value for messages $M^{(T)}$ and $N^{(U)}$ are respectively:

$$C^{(T)}_{\ell_M+1} = E_K(C_{\ell_M} \oplus T),$$
$$D^{(U)}_{\ell_N+1} = E_K(D_{\ell_N} \oplus U).$$

Given $2^{n/2}$ different messages $M^{(T)}$ and $2^{n/2}$ different $N^{(U)}$ along with their MAC, we find a MAC collision with high probability. Since $E_K$ is a permutation, such a collision implies that:

$$C_{\ell_M} \oplus T = D_{\ell_N} \oplus U.$$

As a consequence, we learn the value of $C_{\ell_M} \oplus D_{\ell_N}$, namely $T \oplus U$.

Note that by structuring our choices for $T$ and $U$, we can find a collision with probability 1. One possible choice is to fix the $n/2$ high order bits of $T$ and let the $n/2$ remaining bits of $T$ cover the $2^{n/2}$ possible choices. Similarly, we fix the low order bits of $U$ and let the high order bits cover the possible choices.

Using the technique presented in this section, requires the MAC computation of $2^{1+n/2}$ chosen messages, i.e., $2^{33}$ chosen messages for 64-bit blocks.

### 4.2 Multiple exclusive-or of intermediate values in CBC chains

We now explain how to efficiently obtain a large number of exclusive-ors of intermediate values in CBC chains, following ideas initially proposed in [5]. This useful tool will be applied in section 6.2 to attack general MAC algorithms based on CBC chains.

We first introduce a notation; for any message $M$ formed with blocks $M_1$, $M_2$, $\ldots M_\ell$, we denote by $\mathrm{Internal}(M, i)$ the intermediate value of the CBC chain at position $i$

$$\mathrm{Internal}(M, i) = E_K(E_K(E_K(I(M_1) \oplus M_2) \oplus M_3) \ldots \oplus M_i)$$

We consider a set of $2^{\alpha n}$ unknown intermediate values $X_j$. For each such value, we build a set $\mathcal{S}_j$ of $2^{\beta n}$ MAC tags where $X_j$ is the penultimate intermediate value. Formally, this means that $X_j = \mathrm{Internal}(M[j], i[j])$ for a fixed message $M[j]$ and an index $i[j]$ smaller than the block length of $M[j]$. Then we choose $2^{\beta n}$ blocks $T_k$ and that we query the MAC tags of messages $(M[j]_1, \ldots, M[j]_{i[j]}, T_k)$, for the $2^{\beta n}$ values of $k$, in order to build the set $\mathcal{S}_j$.

Next, we compare the values of the sets $\mathcal{S}_j$. If the same MAC tag appears in both $\mathcal{S}_a$ and $\mathcal{S}_b$, we learn the exclusive-or of $X_a$ and $X_b$, exactly as in the previous section. The probability to obtain $X_a \oplus X_b$ for fixed indexes $a$ and $b$ is about $2^{\beta n} \times 2^{\beta n}/2^n$.

When both $X_a \oplus X_b$ and $X_b \oplus X_c$ are known, $X_a \oplus X_c$ can be easily deduced. In order to construct many exclusive-ors, we make a graph whose vertices are the $2^{\alpha n}$ unknown values $X_j$ and where an edge links two vertices with known exclusive-or obtained by collision of related sets $\mathcal{S}_j$. When a path exists in this graph from $X_a$ to $X_b$, $X_a \oplus X_b$ can be computed. We claim that this graph behaves like a random graph and well known results [4, 8] on such graphs say that as soon as the number of edges is larger than the number of vertices, a "giant component" (with size linear in the total number of vertices) appears. More precisely, with $s$ vertices and $cs/2$ randomly placed edges, with $c > 1$, there is a single giant component whose size is almost exactly $(1 - t(c))s$, where [4]

$$t(x) = \frac{1}{c} \sum_{k=1}^{\infty} \frac{k^{k-1} (ce^{-c})^k}{k!}$$

Since the number of edges is about $2^{(2\alpha+2\beta-1)n}$, we obtain that, if $\alpha + 2\beta$ is larger than 1, the exclusive-or of all pairs of intermediate values in the giant component can be learned. Moreover, with a number of edges larger than the number of vertices, the giant component covers with probability more than 79% of all vertices. Furthermore, the smallest path between most pairs of vertices in the giant component is logarithmic (i.e. linear in $n$). As a consequence, we can efficiently learn the exclusive-or of a fixed proportion, say one half, of the vertices in time $O(n \times 2^{\alpha n})$.

As a conclusion, we obtain the exclusive-or of $O(2^{\alpha n})$ intermediate values asking $O(2^{(\alpha+\beta)n})$ MAC tags, with $\alpha + 2\beta \geq 1$.

## 5 Advanced attacks against algorithm 4 from ISO/IEC 9797–1

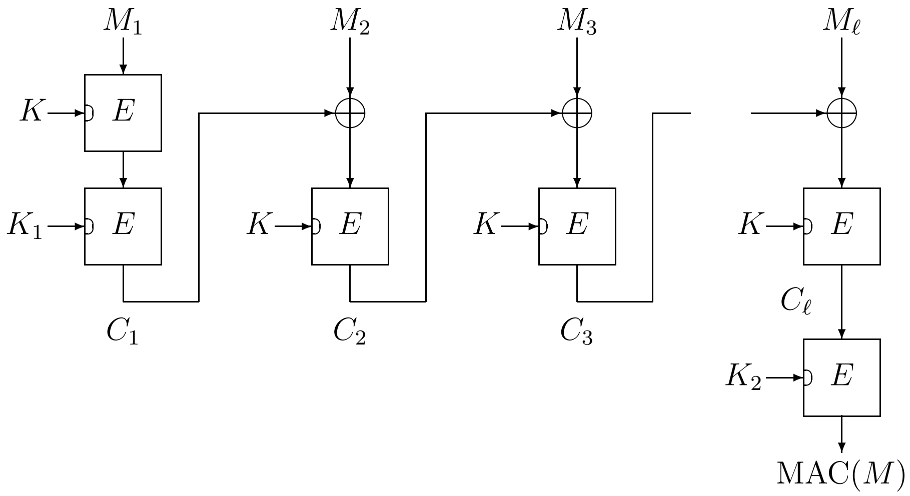


**Fig. 3.** Algorithm 4 from ISO/IEC 9797–1.

Let us recall that in this MAC algorithm, the final value of the CBC chain is encrypted by a final application of the block cipher, with a specific key $K_2$ (see figure 3). Coppersmith and Mitchell [6] have proposed the following attack against this algorithm when padding methods 1 or 2 are used. Notice that even if padding method 3 is preferred, variants based on multiple exclusive-or computation can be applied [5]. The attack goes as follows.

Observe MAC tags of messages of at least two blocks until a collision occurs. Let us note $M$ and $N$ such messages of respective length $\ell_M$ and $\ell_N$ and common MAC tag $m_{coll}$. Since the block cipher $E_{K_2}$ is a permutation, using the notation of section 4.1, we obtain $C_{\ell_M} = D_{\ell_N}$. Consequently, $E_K(C_{\ell_M-1} \oplus M_{\ell_M}) = E_K(D_{\ell_N-1} \oplus N_{\ell_N})$ and $C_{\ell_M-1} \oplus M_{\ell_M} = D_{\ell_N-1} \oplus N_{\ell_N}$, so

$$C_{\ell_M-1} \oplus D_{\ell_N-1} = M_{\ell_M} \oplus N_{\ell_N}$$

Finally, query the MAC tags $m_M$ and $m_N$ of the two truncated messages $M_1 ... M_{\ell_M-1}$ and $N_1 ... N_{\ell_N-1}$. Since $m_M = E_{K_2}(C_{\ell_M-1})$ and $m_N = E_{K_2}(D_{\ell_N-1})$, we obtain

$$E_{K_2}^{-1}(m_M) \oplus E_{K_2}^{-1}(m_N) = M_{\ell_M} \oplus N_{\ell_N}.$$

Then $K_2$ is found by an exhaustive search. We expect that a single value will remain for $K_2$, when the key size is no larger than the block size. Once $K_2$ is known, we can recover $K$ through a second exhaustive search using the test

$$E_K(E_{K_2}^{-1}(m_M) \oplus M_{\ell_M}) = E_{K_2}^{-1}(m_{coll}).$$

Finally, recovering $K_1$ can be done with a final exhaustive search.

This attack requires the observation of $2^{n/2}$ MAC values for known messages, the computation of two MAC values for chosen messages and finally the independent exhaustive searches on a $K_2$, $K$ and $K_1$. When using the DES, we need $2^{32}$ known messages and 2 chosen messages followed by an exhaustive search about four time as expensive as a simple exhaustive search on a single DES key.

## 6 New attacks

### 6.1 Attacking algorithm 5 from ISO/IEC 9797–1

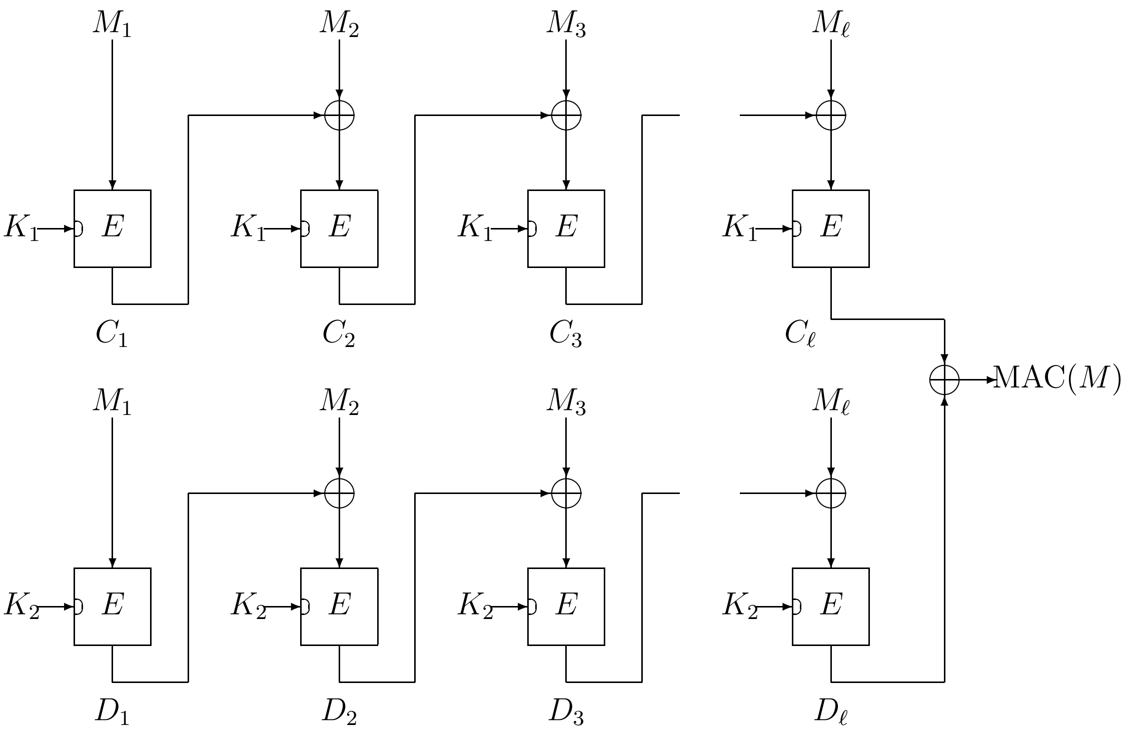


**Fig. 4.** Algorithm 5 from ISO/IEC 9797–1.

Let us recall that in this MAC algorithm, each message goes through two independent plain CBC chain with keys[3] in $K_1$ and $K_2$ (see figure 4). The MAC tag is the

[3] In algorithm 5 from ISO/IEC 9797–1, the keys $K_1$ and $K_2$ are derived from a single key $K$

exclusive-or of the final values of the two chains. We use a variation on the technique from subsection 4.1 to attack this algorithm. The first step of the attack is to find a collision between the MAC tags of two (short, i.e., one block) messages $M$ and $N$. Let $m$ denote the common MAC tag of $M$ and $N$. Moreover, let $\mathcal{E}_{K_1}(M)$, $\mathcal{E}_{K_2}(M)$, $\mathcal{E}_{K_1}(N)$ and $\mathcal{E}_{K_2}(N)$ denote the final values of the 4 CBC chains involved. We have

$$m = \mathcal{E}_{K_1}(M) \oplus \mathcal{E}_{K_2}(M) = \mathcal{E}_{K_1}(N) \oplus \mathcal{E}_{K_2}(N).$$

We would like to learn the value $\delta = \mathcal{E}_{K_1}(M) \oplus \mathcal{E}_{K_1}(N) = \mathcal{E}_{K_2}(M) \oplus \mathcal{E}_{K_2}(N)$. This can be done by computing the MAC values of two long lists of messages $M^{(T)}$ formed by adding a single block $T$ to $M$ and $N^{(U)}$ by adding a block $U$ to $N$. It is easy to check that whenever $T \oplus U = \delta$, we get a collision for both of the CBC chains and of course a collision on the MAC values of the extended messages. Moreover, this kind of double collisions can be distinguished from "ordinary" collisions. Indeed, if we add any block, say the zero block, at the end of both $M^{(T)}$ and $N^{(U)}$ the resulting messages still collide. Once $\delta$, $M^{(T)}$ and $N^{(U)}$ are known, we can proceed either with a forgery attack or a key recovery attack. It should be noted that for this particular algorithm, no efficient forgery attack was previously known.

*Forgery attack.* With a double collision between $M^{(T)}$ and $N^{(U)}$ in hand, making forgery is easy. Indeed, for any message completion $L$, the MAC value of $M^{(T)}$ concatenated with $L$ and the MAC value of $N^{(U)}$ concatenated with $L$ are necessarily equal. Indeed, the double collision propagates along $L$. Thus, it is easy to ask the MAC tag of one of the two extended messages and to guess that the MAC tag of the other extended message has the same value. The cost forgery attack is independent of the size of the keys of $E$, it is only a function of the block size $n$. The attack requires the computation of $2^{1+n/2}$ MAC values.

*Key recovery attack.* Since we know the value of $\delta$, we know the two values $\mathcal{E}_{K_1}(M) \oplus \mathcal{E}_{K_1}(N)$ and $\mathcal{E}_{K_2}(M) \oplus \mathcal{E}_{K_2}(N)$ (both are equal to $\delta$). Thus, we get two independent conditions respectively on $K_1$ and $K_2$, and the keys can be recovered with a simple exhaustive search. Indeed, assume that $M$ and $N$ are different one block messages, then search for a key $K$ that satisfy:

$$E_K(M_1) \oplus E_K(N_1) = \delta.$$

When the key size is no larger than the block size, we expect to find two different solutions during the exhaustive search, $K_1$ and $K_2$. Furthermore, if there are more than two candidate solutions, we simply form all possible candidate pairs and keep the pair which is compatible with all the MAC values we already know. The key recovery attack requires the observation of $2^{n/2}$ MAC tags followed by the computation of $2^{1+n/2}$ MAC values. When using the DES, we need $2^{33}$ message followed by an exhaustive search roughly equivalent to a simple exhaustive search on a single DES key.

### 6.2 General MAC algorithms with a single CBC chain

In this subsection, we consider key recovery attacks against general MAC algorithm based on a single CBC chain with a key $K$. We let $I$ and $F$ denote the initial and final transforms as in section 2. In this subsection, our goal is to recover the key $K$ of the main CBC chain as efficiently as possible. We assume that $I$ and $F$ are both keyed transformations which cannot be computed by the attacker since it (at first) does not know any key material. We first address the special case where $I$ and $F$ are closely related transformations (with identical keys) before considering the general case.

*The special case $I = F \circ E_K$.* For example, a natural MAC scheme could apply the same triple-DES transformation, both at the beginning and at the end of the MAC computation. With our notations, this means that $I = E_K \circ D_{K_1} \circ E_K$ and $F = E_K \circ D_{K_1}$. None of the previously described attacks apply to such a scheme.

Let us consider $2^{\alpha n}$ blocks $M_i$ and the associated internal value

$$X_i = \text{Internal}(M_i, 1) = I(M_i).$$

In order to learn $I(M_i)$, we first remark that $\Delta_I(M_i) = I(M_i) \oplus I(M_i \oplus 1)$ can be seen as an identifier for $M_i$. Of course, this identifier is somewhat ambiguous, since $\Delta_I(M_i)$ and $\Delta_I(M_i \oplus 1)$ are identical. However, given any value for the identifier, it has only a few associated values (unless $I$ is almost linear, in which case simpler attacks are available).

With this in mind, we can apply the technique of subsection 4.2 that computes $\Delta_I(M_i)$ for $O(2^{\alpha n})$ blocks $M_i$ asking $2^{(\alpha+\beta)n}$ MAC tags (with $\alpha + 2\beta$ larger than 1).

Then, we ask for the MAC tags of pairs of messages whose only difference is that the last blocks are respectively $T_j$ and $T_j \oplus 1$. We denote by $Y_j$ the intermediate value after the exclusive-or with $T_j$, i.e., the input of the final transformation $F \circ E_K = I$. Consequently, when the last block is $T_j \oplus 1$, the input of $I$ is $Y_j \oplus 1$. So, the exclusive-or of queried MAC tags reveals $\Delta_I(Y_j)$.

Finally, if we compute $2^{(1-\alpha)n}$ values $\Delta_I(Y_j)$, we obtain, with high probability, a collision with one of the $\Delta_I(M_i)$. As we already explained, $\Delta_I$ is a good identifier so we probably have $M_i = Y_j$. However, there might be false alarms, i.e., apparent collision not resulting from a real one. False alarms are easy to detect by computing a few additional MAC values.

Given a real collision, we know that $I(M_i)$ is equal to the MAC tag whose last intermediate value is $Y_j$. Thus, since we have learned the initial value of the CBC chain, we can compute $K$ through exhaustive search, as we previously explained. This attack requires a total of approximately $2^{(\alpha+\beta)n} + 2^{(1-\alpha)n}$ MAC computations for chosen messages, with $\alpha + 2\beta \geq 1$. The best compromise is obtained with $\alpha = \beta = 1/3$ and the number of queried MAC tags is about $2^{2n/3}$.

When using the DES, we need $2^{43}$ messages followed by an exhaustive search roughly equivalent to a simple exhaustive search on a single DES key.

*The general case with arbitrary I and F.* We finally explain that, even if $I$ and $F$ are arbitrary transformations, the internal key $K$ can still be attacked, even if the complexity is less practical than in previous cases.

Always using the technique of section 4.2 we can consider $2^{\alpha n}$ intermediate values $X_i$ which are unknown but whose pairwise exclusive-or are known. This requires the query of $2^{(\alpha+1)n/2}$ MAC tags of chosen messages.

Then, for each intermediate value $X_i$, the technique of section 4.1 allows to compute $\Delta_i = E_K(X_i) \oplus E_K(X_i \oplus 1)$ asking $2^{n/2}$ MAC computation for each $X_i$. With this list of $\Delta_i$ in mind, we know guess a key $K'$ and a block $X$, and we compute $\Delta = E_{K'}(X) \oplus E_{K'}(X \oplus 1)$. If $K = K'$ and $X$ is one of the $X_i$s, $\Delta$ is in the list of the $\Delta_i$s. We do not know the related $X_i$ value but we know $\delta = X_i \oplus X_j$ for any other $j$. Consequently, we learn the following test

$$E_K(X \oplus \delta) \oplus E_K(X \oplus \delta \oplus 1) = \Delta_j.$$

This allows to know if we have really guessed the correct key $K$ or if it is only a false alarm.

The probability to correctly guess $K = K'$ and that $X$ is one of the $X_i$s is about 1 over $2^k \times 2^{(1-\alpha)n}$, where $k$ is the key size of $K$. The total number of MAC queries is $2^{(\alpha+1)n/2} + 2^{n/2}$ and the complexity of the search on $K'$ and $X$ is $O(2^{k+(1-\alpha)n})$. According to the choice of $\alpha$, we obtain different compromises. The main ones are:

| $\alpha$ parameter | number of MAC queries | search complexity |
|---|---|---|
| $\alpha = 0$ | $O(2^{n/2+1})$ | $O(2^{k+n})$ |
| $\alpha = 1/2$ | $O(2^{3n/4})$ | $O(2^{k+n/2})$ |
| $\alpha = 1$ | $O(2^n)$ | $O(2^k)$ |

## 7 Conclusion

The main conclusion of this paper is that the use of MAC algorithms based on an internal CBC chain using a weak block cipher such as DES must be carefully reconsidered, whatever the initial and final transformation may be. A much more secure approach is to use a strong block cipher such as AES with a provably secure MAC algorithm.


### Acknowledgments

We would like to thank the anonymous referees for pointing out important references.


## References


1. ANSIX9.19, American National Standard–Financial institution retail message authentication, 1986.
2. ANSIX9.9, American National Standard–Financial institution message authentication (wholesale), 1982. Revised in 1986.
3. M. Bellare, J. Kilian, and P. Rogaway. The Security of the Cipher Block Chaining Message Authentication Code. In *Crypto '94*, LNCS 839, pages 362–399. Springer-Verlag, 1994.

4. B. Bollobás. *Random Graphs.* Academic Press, New York, 1985.
5. D. Coppersmith, L.R. Knudsen, and C.J. Mitchell. Key recovery and forgery attacks on the MacDES MAC algorithm. In *Crypto 2000*, LNCS 1880, pages 184–196. Springer-Verlag, 2000.
6. D. Coppersmith and C.J. Mitchell. Attacks on MacDES MAC algorithm. *Electronic Letters*, 35:1626–1627, 1999.
7. ISO/IEC 9797–1, Information technology–Security techniques–Message Authentication Codes (MACs)–Part 1: Mechanisms using a block cipher, 1999.
8. S. Janson, T. Łuczak, and A. Ruciński. *Random Graphs.* John Wiley, New York, 1999.
9. L.R. Knudsen and B. Preneel. MacDES: MAC algorithm based on DES. *Electronic Letters*, 34:871–873, 1998.
10. NIST. Computer Data Authentication, may 1985. Federal Information Processing Standards PUBlication 113.
11. NIST. Recommendation for Block Cipher Modes of Operation: The RMAC Authentication Mode, november 2002. NIST Special Publication 800-38B.
12. B. Preneel and P.C. van Oorschot. On the security of iterated Message Authentication Codes. *IEEE Transactions on Information Theory*, 45(1):188–199, January 1999.