Authenticity Modes

Halil İbrahim Kaplan

TDBY / Kripto Analiz Laboratuvarı

halil.kaplan@tubitak.gov.tr

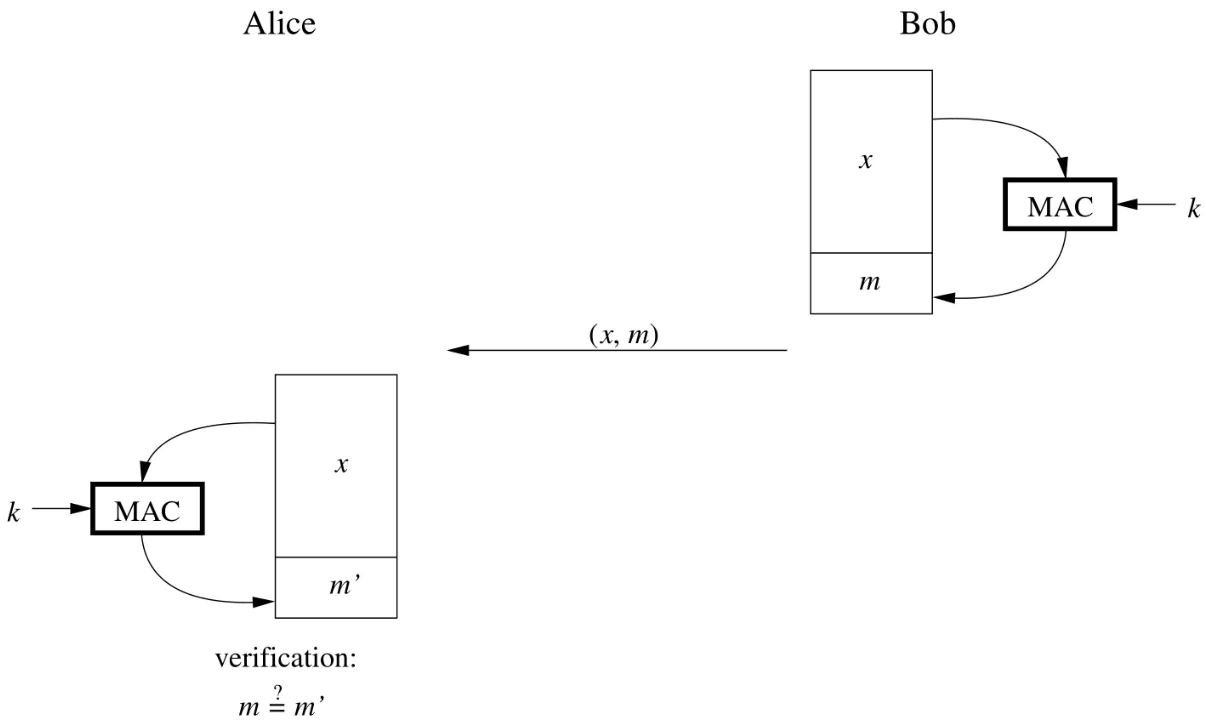
2021

TÜBİTAK

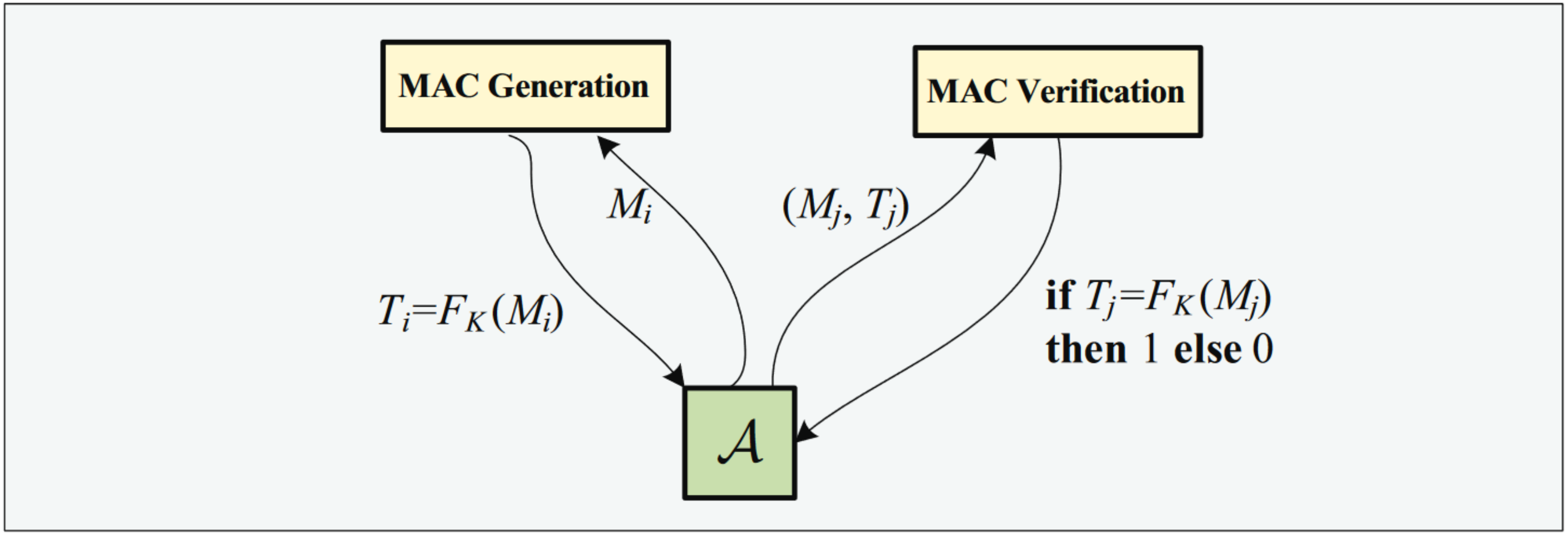
BİLGEM

Overview

1. **Preliminaries**
2. **CBC-MAC Algorithms**
3. **CMAC Mode**
4. **HMAC Mode**



The motivation for using MACs : Alice and Bob want to be assured that any manipulations of a message x in transit are detected.



A **MAC-generation** oracle takes as input a messages $M \in \mathcal{M}$. The oracle returns $T = F_K(M)$, the MAC of the queried string.

A **MAC-verification** oracle takes as input a pair of strings $(M, T) \in \mathcal{M} \times \{0,1\}^\tau$. The oracle returns 1 if $\text{MAC}_K(M) = T$ and 0 otherwise.

The adversary’s aim:

- Ask its MAC-verification oracle a query (M,T) that causes it to output 1 even though the query M was no previously made to the MAC-generation oracle.

- Such a pair (M,T) is called a forgery.

Padding methods :

$$Pad_1 : \mathrm{M} \rightarrow \mathrm{M}\ 0^{i}$$

i : the least nonnegative number such that |M| + i is a positive multiple of n.

$$Pad_2 : \mathrm{M} \rightarrow \mathrm{M}\ 1\ 0^{i}$$

i : the least nonnegative number such that |M| + i + 1 is a positive multiple of n.

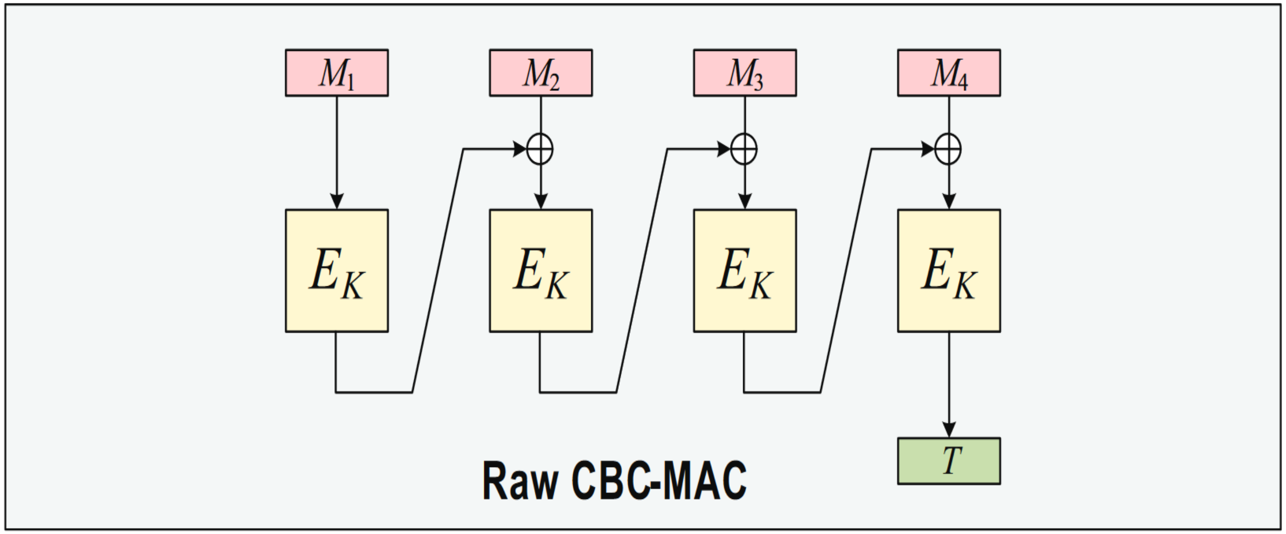
$$Pad_3 : \mathrm{M} \rightarrow \mathrm{L}\ \mathrm{M}\ 0^{i}$$

i : the least nonnegative number such that |M| + i is a positive multiple of n.
L : binary encoding of |M| into a field of n bits.

CBC-MAC Algorithms

ISO/IEC 9797-1

Raw CBC-MAC :

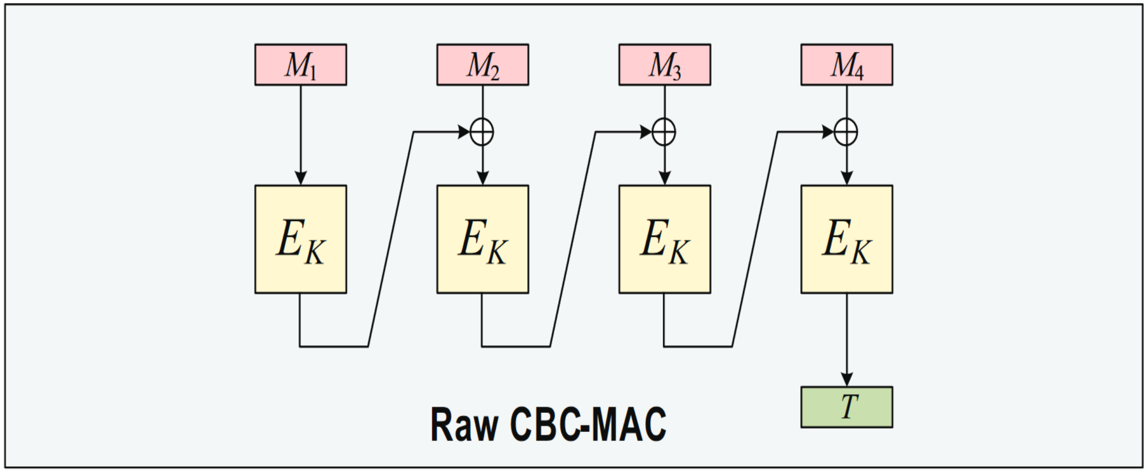


has major restrictions:

1. Restricted domain:

- The input to the raw CBC-MAC must be a positive multiple of n bits, where n is the blocklength of the underlying blockcipher $E : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$
- The algorithm is not defined for other input lengths.
- This makes padding necessary.

2. Cut-and-paste attacks:

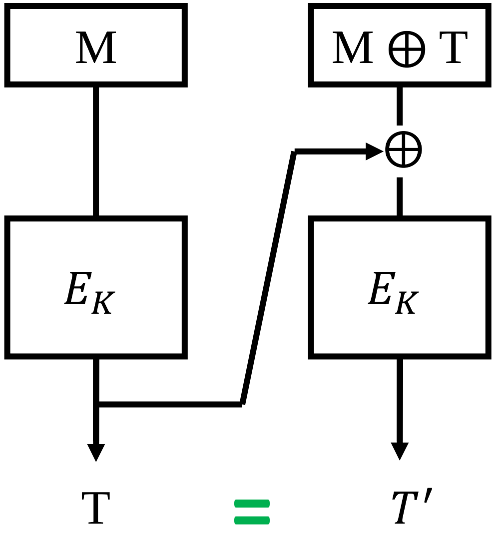


Suppose Adversary obtains T = $CBCMAC_K(M)$ for single block M.

He can forge the message M || (M ⊕ T) with tag of T.

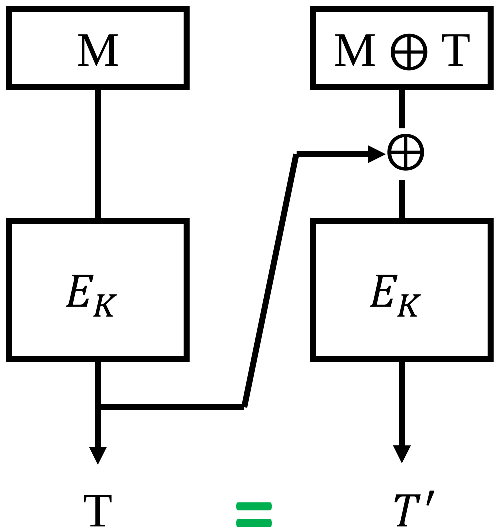
Suppose Adversary obtains T = $CBCMAC_K(M)$ for single block M.

He can forge the message M || (M ⊕ T) with tag of T.



Suppose Adversary obtains T = $CBCMAC_K(M)$ for single block M.

He can forge the message M || (M ⊕ T) with tag of T.



Raw CBC-MAC is not VIL secure.

3. **Birthday attacks:**

   Reduces the complexity of brute force attack from 2^{n} to $2^{\frac{n}{2}}$

4. **Key-guessing attacks:**

   Message can be forged with spending 2^{k} time.

Not much problem when blockcipher is AES.

5. **Efficiency issues:**

   |M| / n blockcipher calls calls needed.

   These calls must be computed serially.

Alg	Pad	Sep	Klen	Mlen	#Calls	Goals	Proof	Attack	Ref
1	1	—	k	$[0..\infty)$	$\lceil\mu/n\rceil$	B F	✓	—	[18]
1	2	—	k	$[0..\infty)$	$\lceil\mu'/n\rceil$	B V	—	✓	folklore
1	3	—	k	$[0..2^n-1]$	$\lceil\mu/n\rceil+1$	B V	✓	—	[167]
2	1	opt	k, $2k$	$[0..\infty)$	$\lceil\mu/n\rceil+1$	B F	✓	—	[167]
2	2	opt	k, $2k$	$[0..\infty)$	$\lceil\mu'/n\rceil+1$	B V	✓	—	[167]
2	3	opt	k, $2k$	$[0..2^n-1]$	$\lceil\mu/n\rceil+2$	B V	✓	—	[167]
3	1	—	$2k$	$[0..\infty)$	$\lceil\mu/n\rceil+2$	B F K	✓	✓	[18]
3	2	—	$2k$	$[0..\infty)$	$\lceil\mu'/n\rceil+2$	B V K	✓	✓	[39]
3	3	—	$2k$	$[0..2^n-1]$	$\lceil\mu/n\rceil+3$	B V K	✓	✓	[167]
4	1	✓	$2k$	$[n+1..\infty]$	$\lceil\mu/n\rceil+2$	B F K	✓	✓	[55]
4	2	✓	$2k$	$[n..\infty]$	$\lceil\mu'/n\rceil+2$	B V K	✓	✓	[55]
4	3	✓	$2k$	$[0..2^n-1]$	$\lceil\mu/n\rceil+3$	B V K	✓	✓	[54]
5	1	✓	k	$[0..\infty)$	$2\lceil\mu/n\rceil$	C F	—	✓	[105]
5	2	✓	k	$[0..\infty)$	$2\lceil\mu'/n\rceil$	C V	—	✓	[105]
5	3	✓	k	$[0..2^n-1]$	$2\lceil\mu/n\rceil+2$	C V	—	—	—
6	1	✓	$2k$	$[n+1..\infty]$	$2\lceil\mu/n\rceil+4$	C F K	✓	—	[206]
6	2	✓	$2k$	$[n..\infty]$	$2\lceil\mu'/n\rceil+4$	C V K	✓	—	[206]
6	3	✓	$2k$	$[0..2^n-1]$	$2\lceil\mu/n\rceil+6$	C V K	✓	—	[206]

B = Provable security up to birthday bound

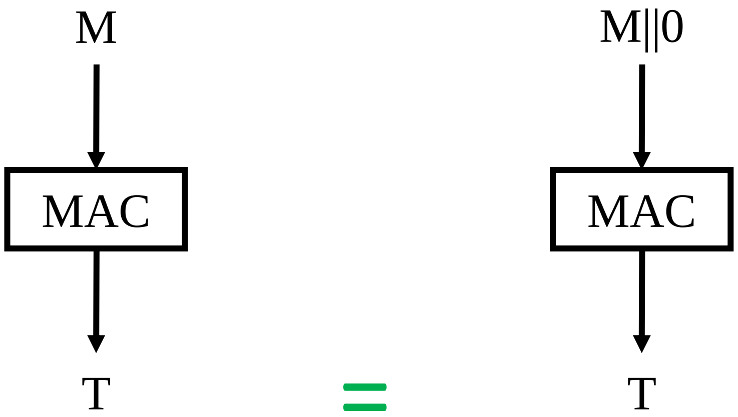
C = Provable security beyond the birthday bound

V = VIL secure

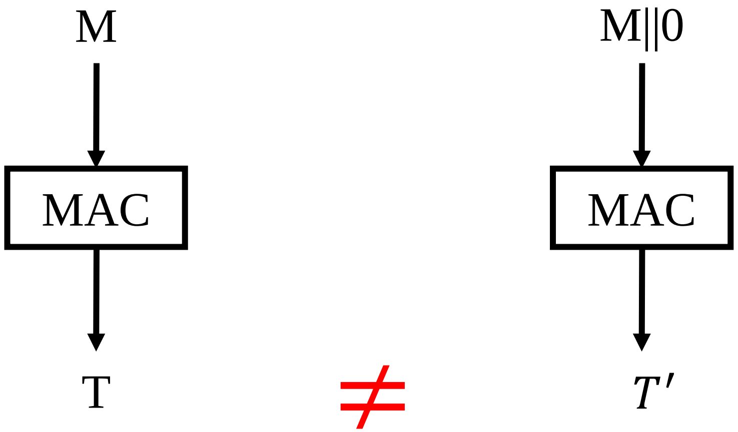
F = FIL secure

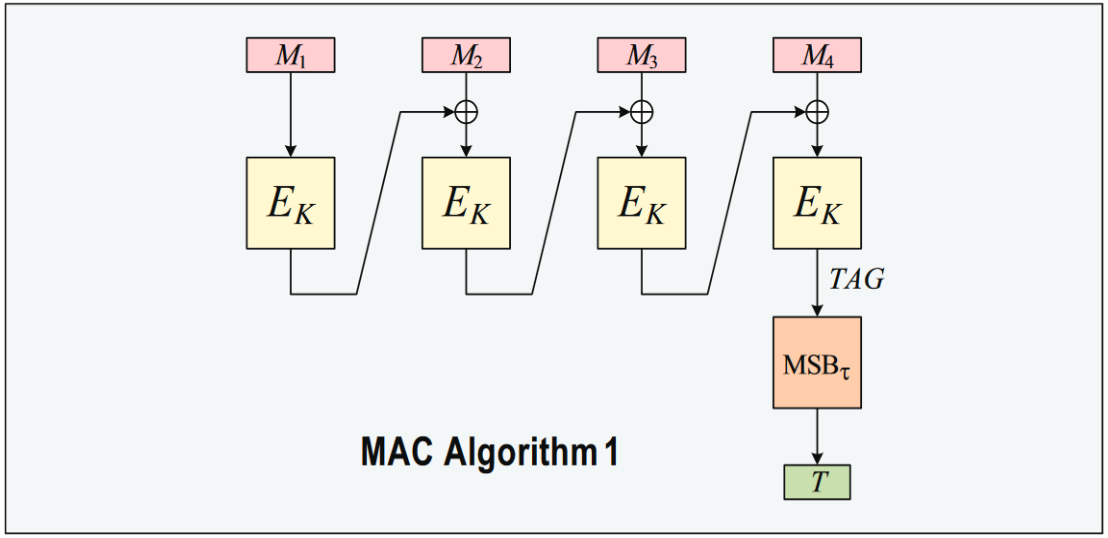
K = Enhanced key-length security

Padding 1 :



Padding 2,3 :

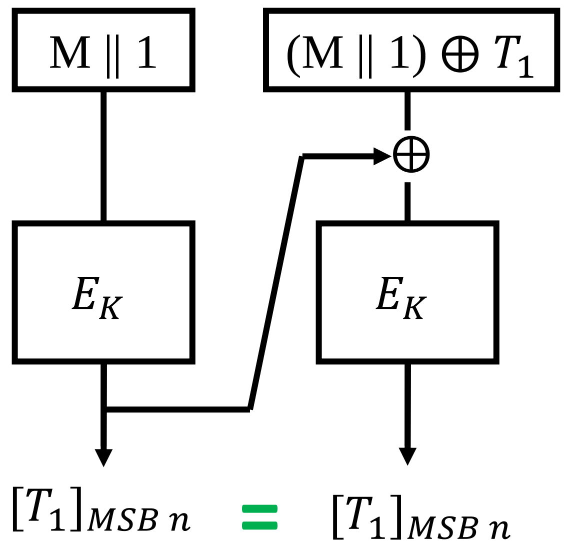




Alg	Pad	Sep	Klen	Mlen	#Calls	Goals	Proof	Attack	Ref
1	1	—	k	$[0..\infty)$	$\lceil \mu/n \rceil$	B F	✓	—	[18]
1	2	—	k	$[0..\infty)$	$\lceil \mu'/n \rceil$	B V	—	✓	folklore
1	3	—	k	$[0..2^n-1]$	$\lceil \mu/n \rceil + 1$	B V	✓	—	[167]

Using Pad_2 if $\tau =$ blocksize (n)

Let M be message with size n-1

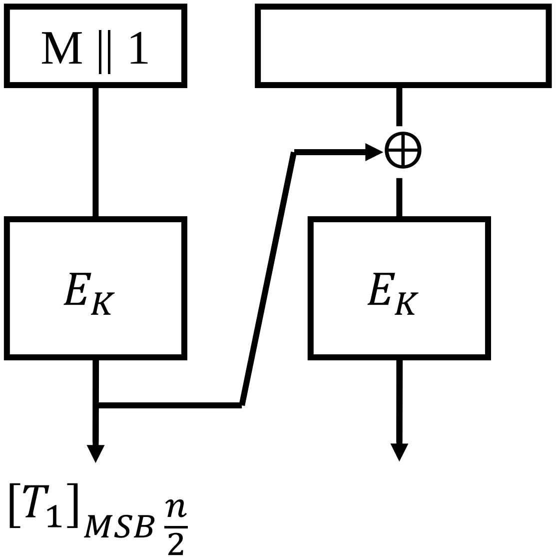


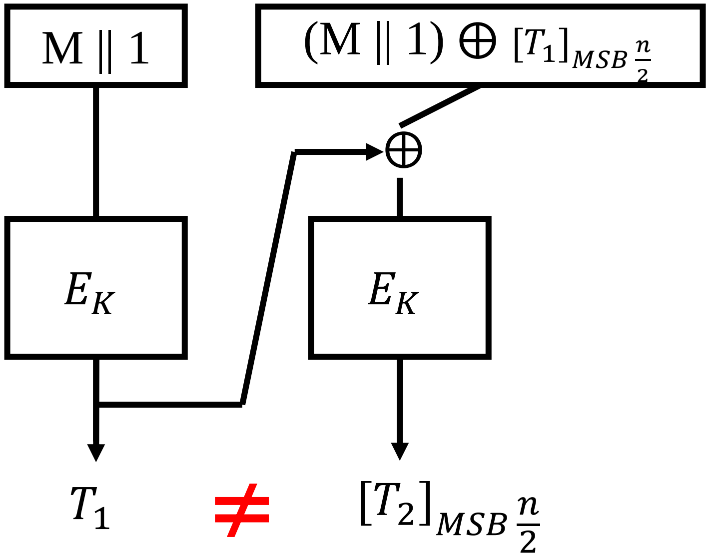
Not VIL secure

Using Pad_2 if $\tau =$ half of the blocksize $(\frac{n}{2})$

Let M be message with size n-1

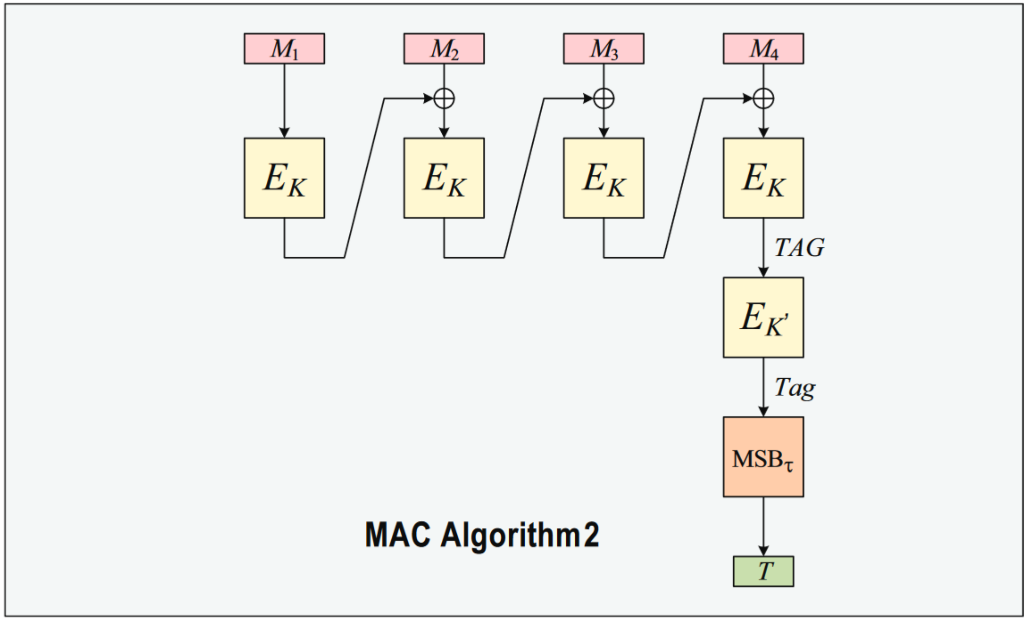
We only know $[T_1]_{MSB\frac{n}{2}}$





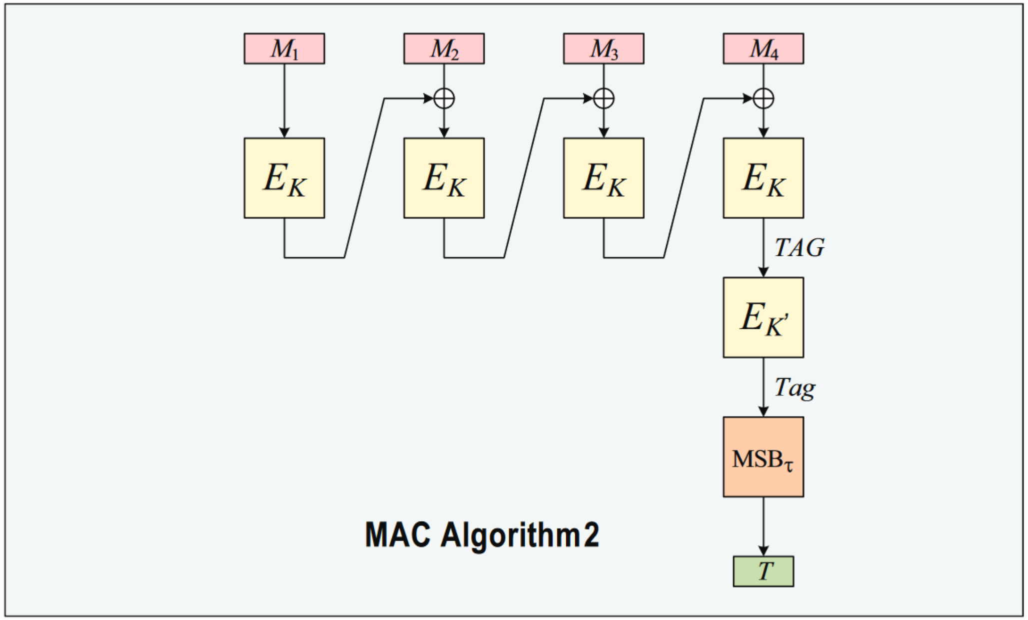
VIL secure

CBC-MAC Algorithms



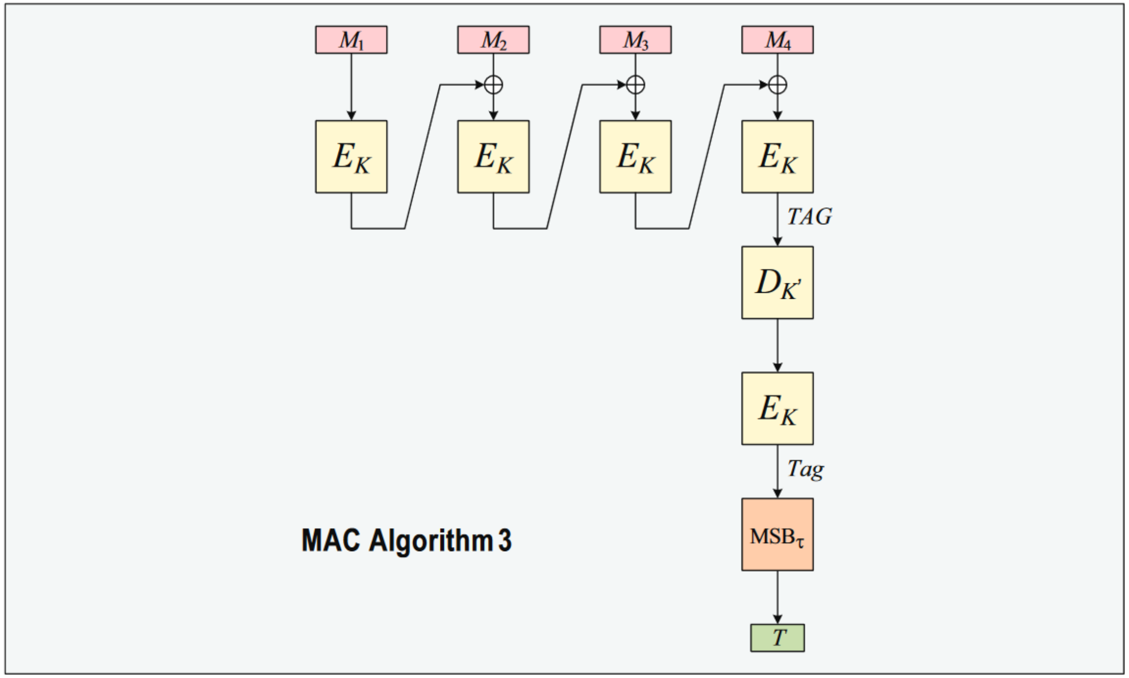
- Has 2 keys.
- Keys must be independent
- The ISO standart says nothing about key seperation.

CBC-MAC Algorithms



Alg	Pad	Sep	Klen	Mlen	#Calls	Goals	Proof	Attack	Ref
2	1	opt	k, $2k$	$[0..\infty)$	$\lceil\mu/n\rceil+1$	B F	✓	—	[167]
2	2	opt	k, $2k$	$[0..\infty)$	$\lceil\mu'/n\rceil+1$	B V	✓	—	[167]
2	3	opt	k, $2k$	$[0..2^n-1]$	$\lceil\mu/n\rceil+2$	B V	✓	—	[167]

CBC-MAC Algorithms

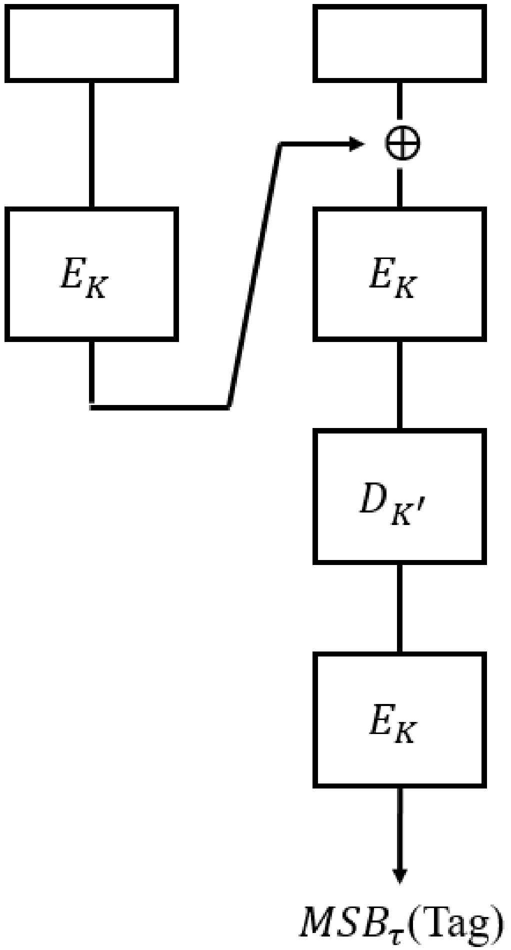


- More secure then Algorithm 2 wrt. exhaustive key search (?)
- The ISO standart says nothing about key seperation.

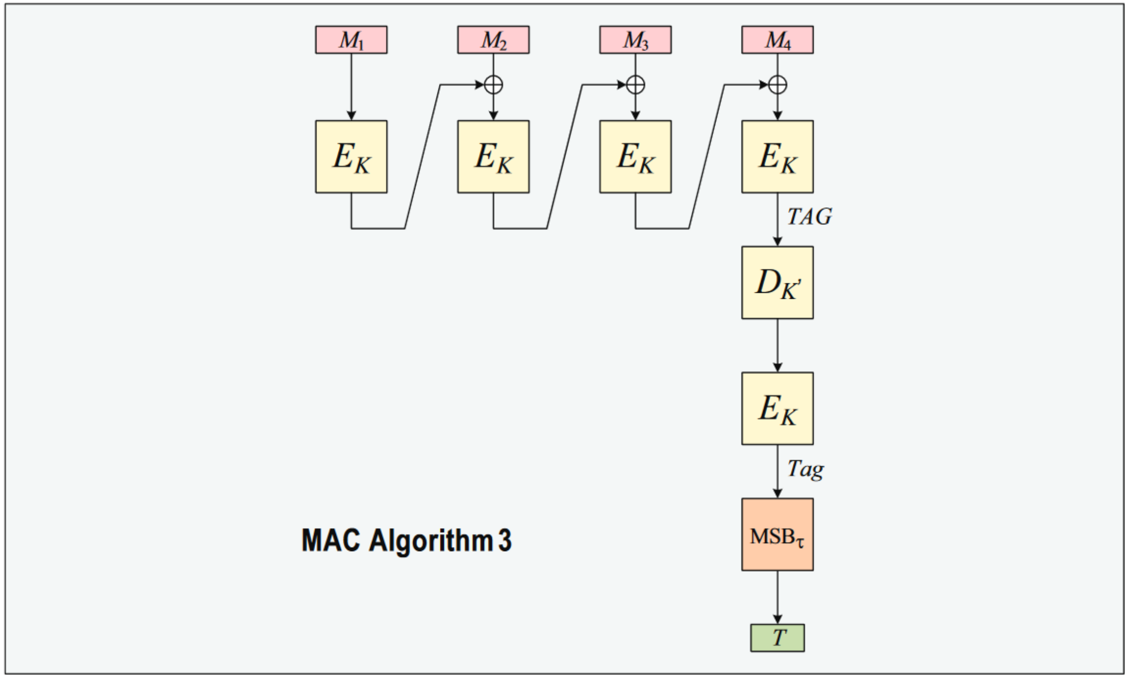
Attack on Algorithm 3(DES) :

- Collect $2^{32.5}$ message / tag pairs for 128 bit messages.
- Get X and X' s.t. MAC(X) = MAC(X')
- Trying all 2^{56} keys, enciphering two blocks with each, find a key K such that the input to the final CBCMAC enciphering call is the same for X and X'
- Spending another 2^{56} time, recover K'using the data in hand.

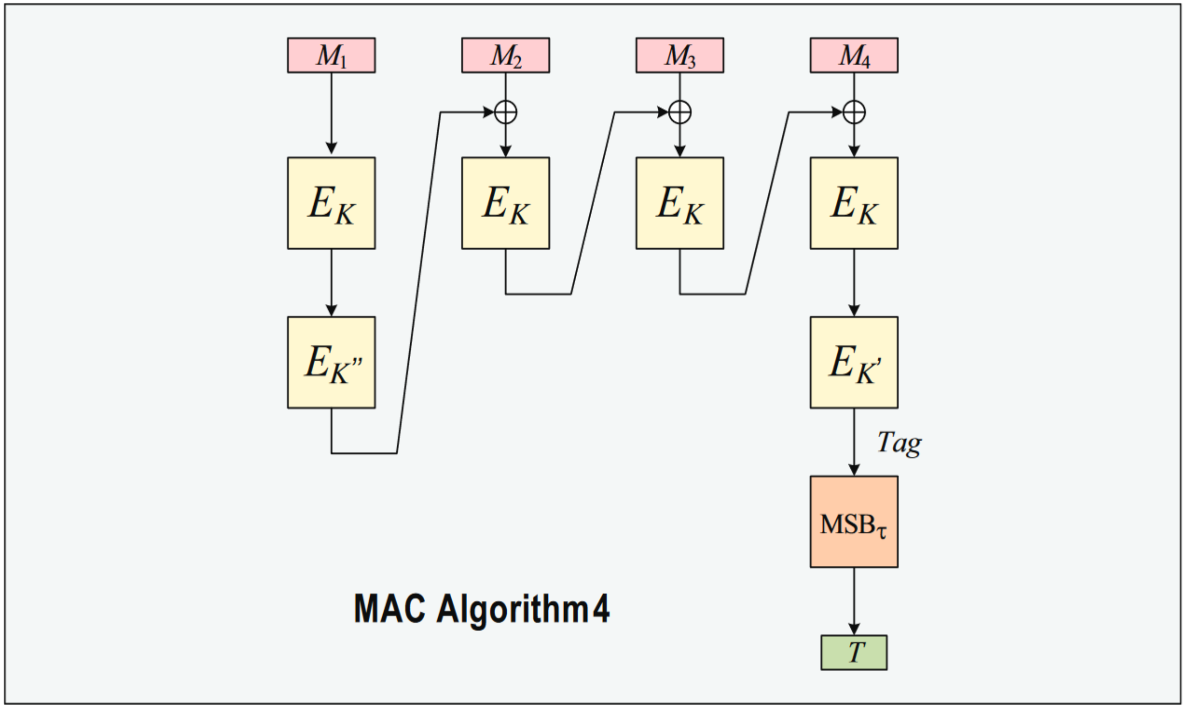
Known message/MAC pairs = $2^{32.5}$
Time = $2^{57.6}$



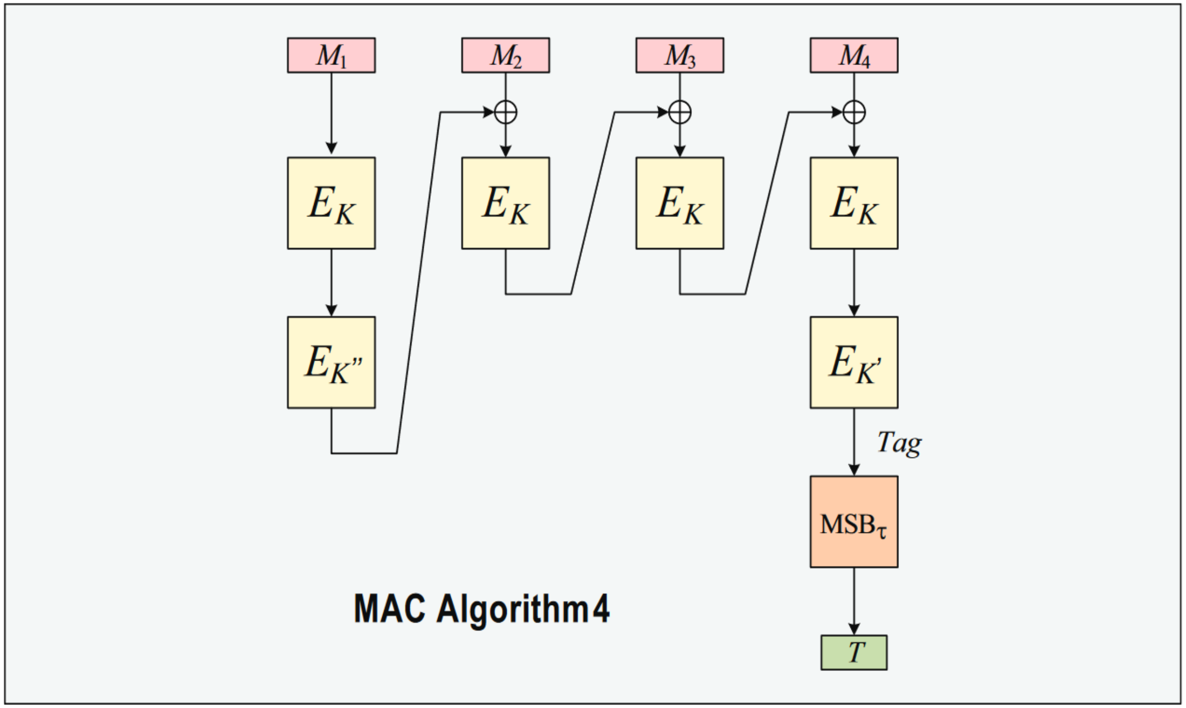
CBC-MAC Algorithms



Alg	Pad	Sep	Klen	Mlen	#Calls	Goals	Proof	Attack	Ref
3	1	—	$2k$	$[0..\infty)$	$\lceil \mu/n \rceil + 2$	B F K	✓	✓	[18]
3	2	—	$2k$	$[0..\infty)$	$\lceil \mu'/n \rceil + 2$	B V K	✓	✓	[39]
3	3	—	$2k$	$[0..2^n - 1]$	$\lceil \mu/n \rceil + 3$	B V K	✓	✓	[167]



- Known also MacDES
- The intent is to be able to use DES in settings where 2^{56} computation is not out of the question.
- The ISO standart says nothing about key seperation.



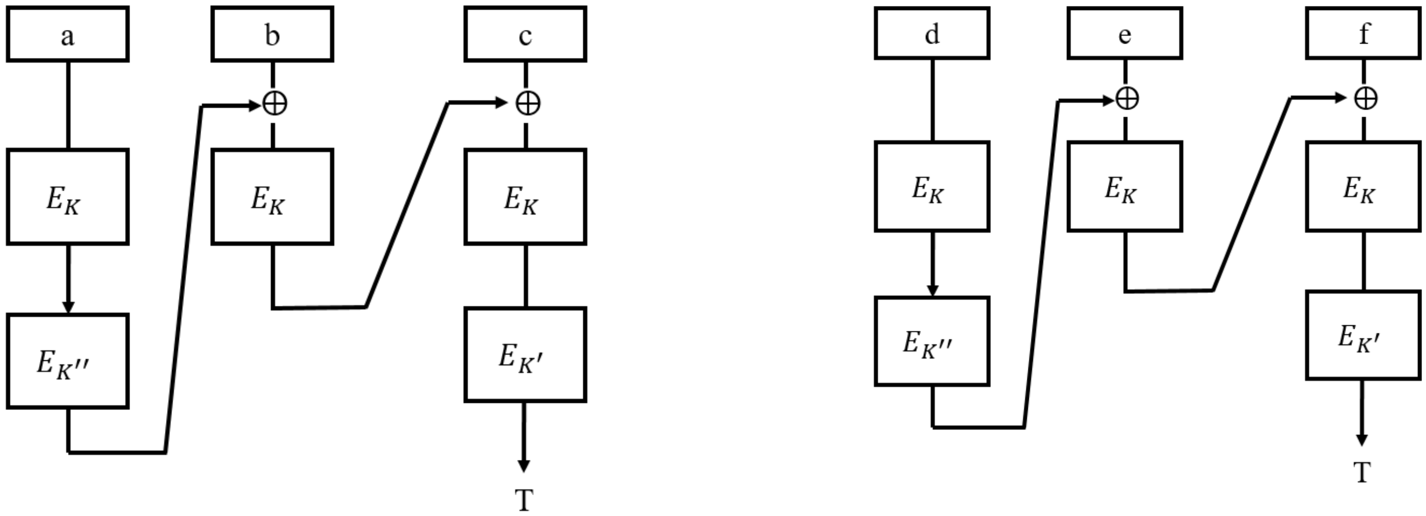
Alg	Pad	Sep	Klen	Mlen	#Calls	Goals	Proof	Attack	Ref
4	1	✓	$2k$	$[n+1 .. \infty]$	$\lceil \mu/n \rceil + 2$	B F K	✓	✓	[55]
4	2	✓	$2k$	$[n .. \infty]$	$\lceil \mu'/n \rceil + 2$	B V K	✓	✓	[55]
4	3	✓	$2k$	$[0 .. 2^n - 1]$	$\lceil \mu/n \rceil + 3$	B V K	✓	✓	[54]

Attack on Algorithm 4 :

Padding : Pad_1
$\tau = n$

Adversary asks messages s.t

$$MAC_4(a|b|c) = MAC_4(d|e|f)$$



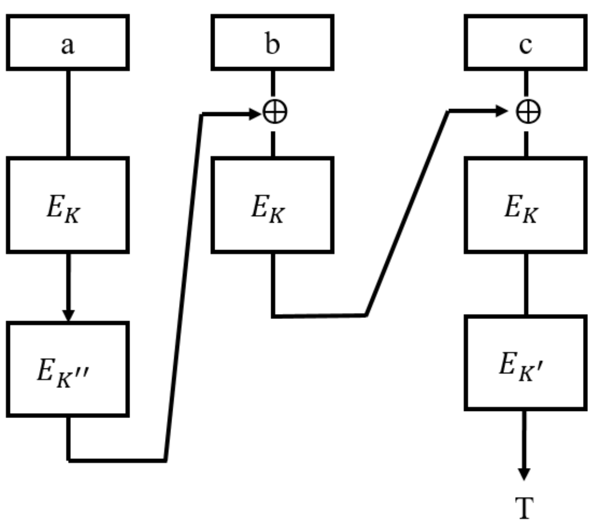
Attack on Algorithm 4 :

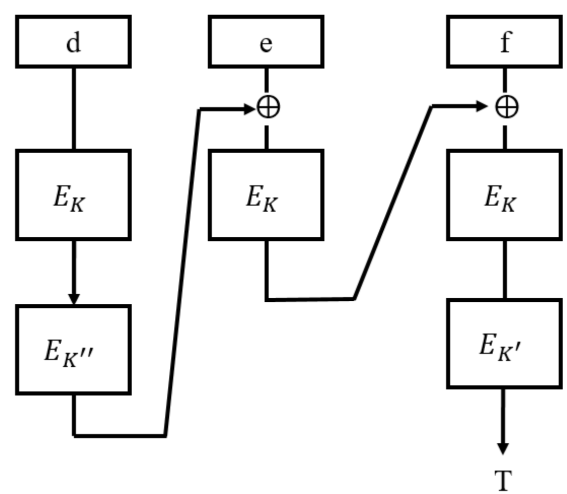
Padding : Pad_1

$\tau = n$

Adversary asks messages s.t

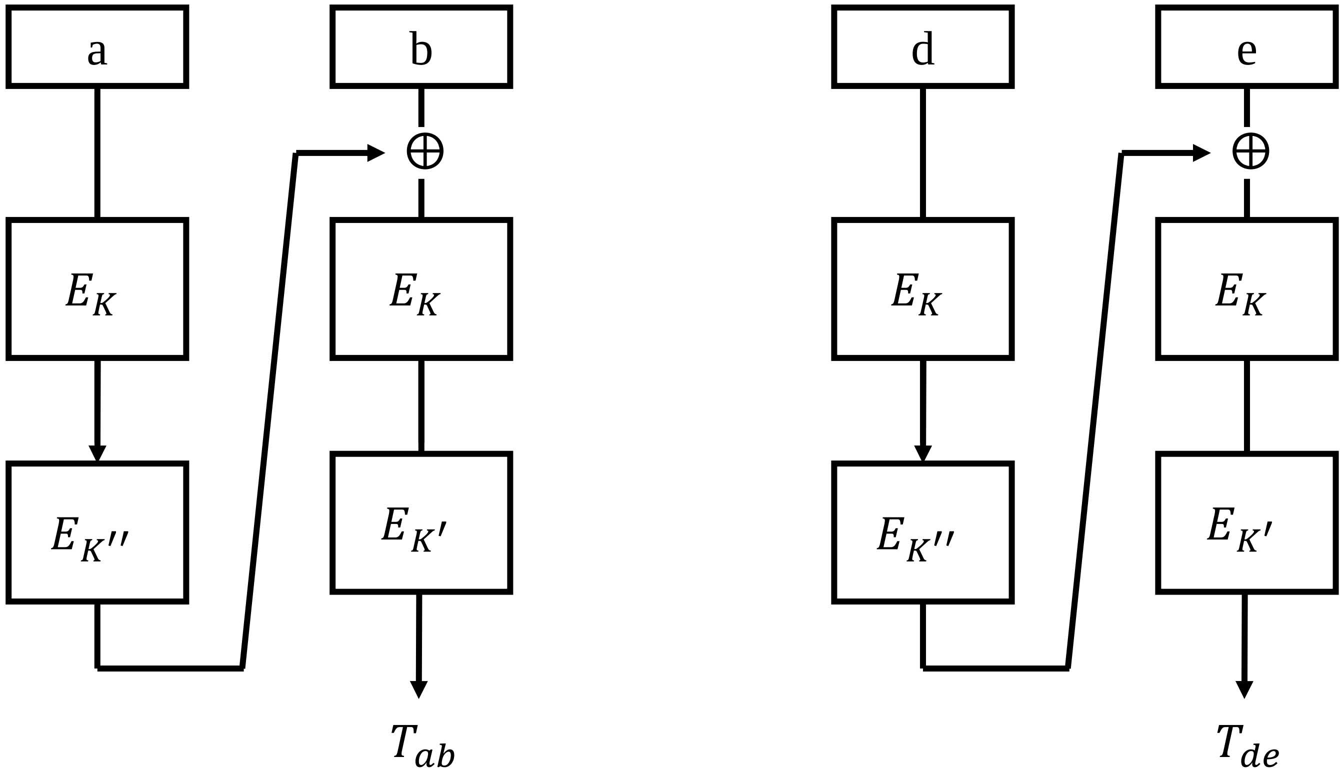
$$MAC_4(a|b|c) = MAC_4(d|e|f)$$





Attack on Algorithm 4 :

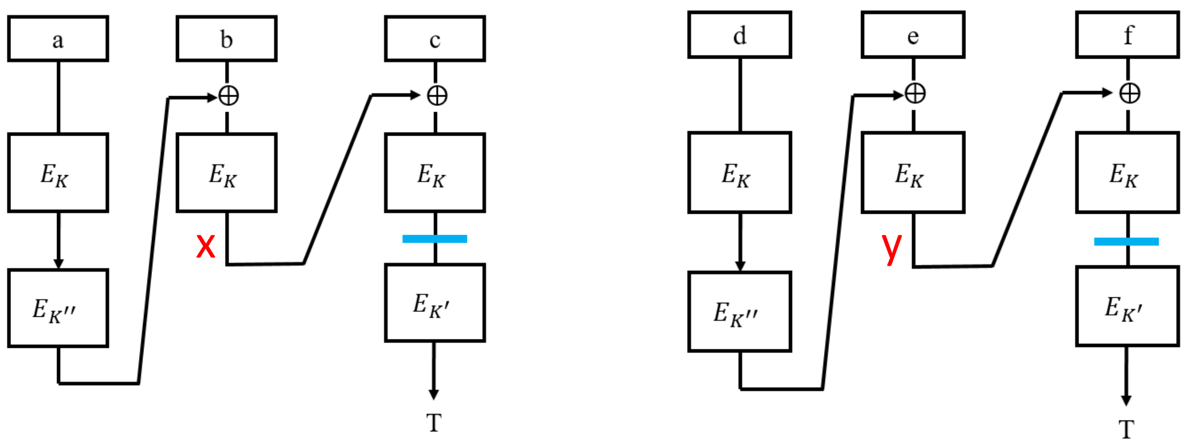
Strip off final block and learn T_{ab} and T_{de} .



Attack on Algorithm 4 :

From first part we have

$$E_K(x \oplus c) = E_K(y \oplus f)$$

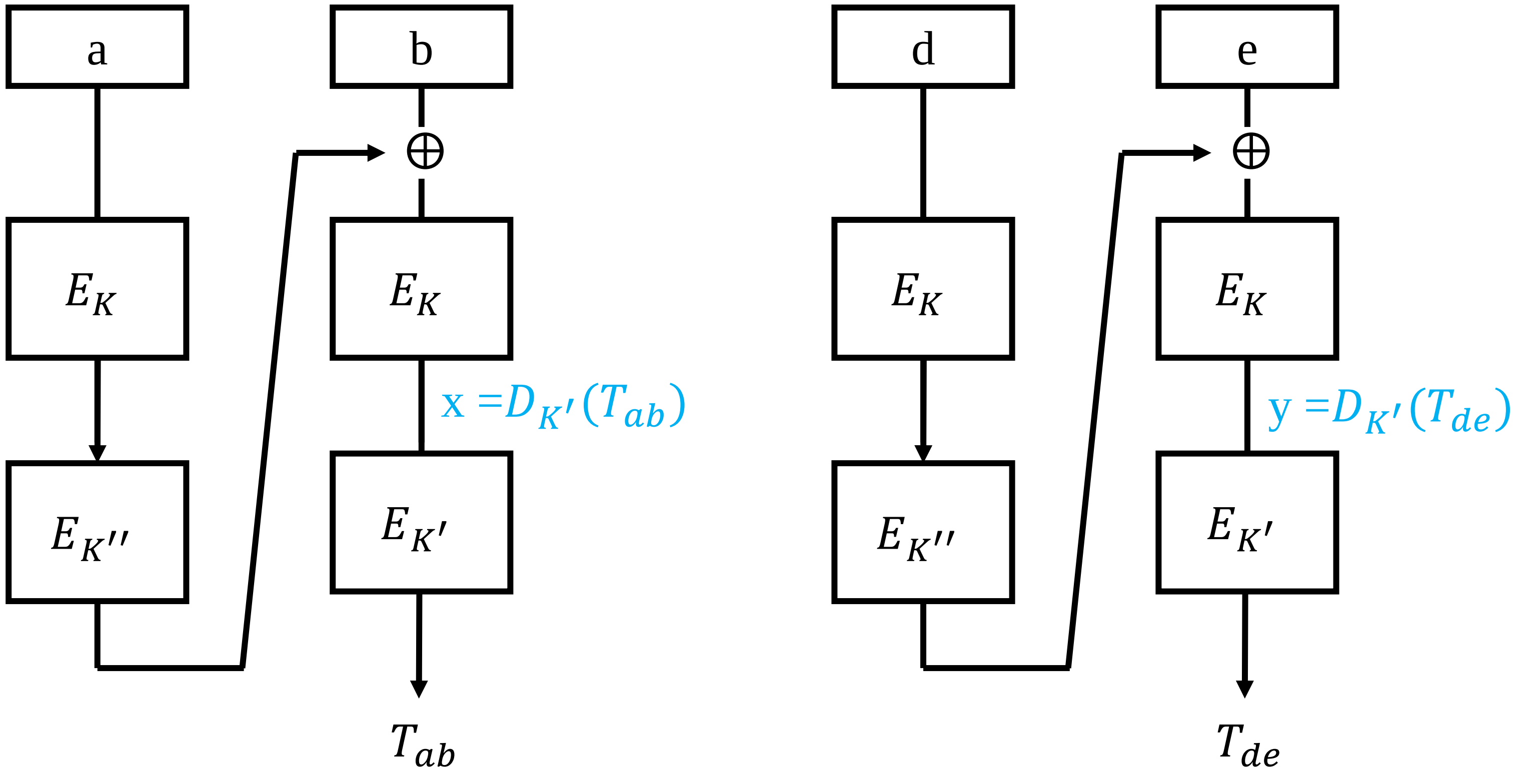


x : output of the blockcipher call that processed b (when MACing a ‖ b ‖ c)

y : output of the blockcipher call that processed e (when MACing d ‖ e ‖ f)

Attack on Algorithm 4 :

From second part we have



Attack on Algorithm 4 :

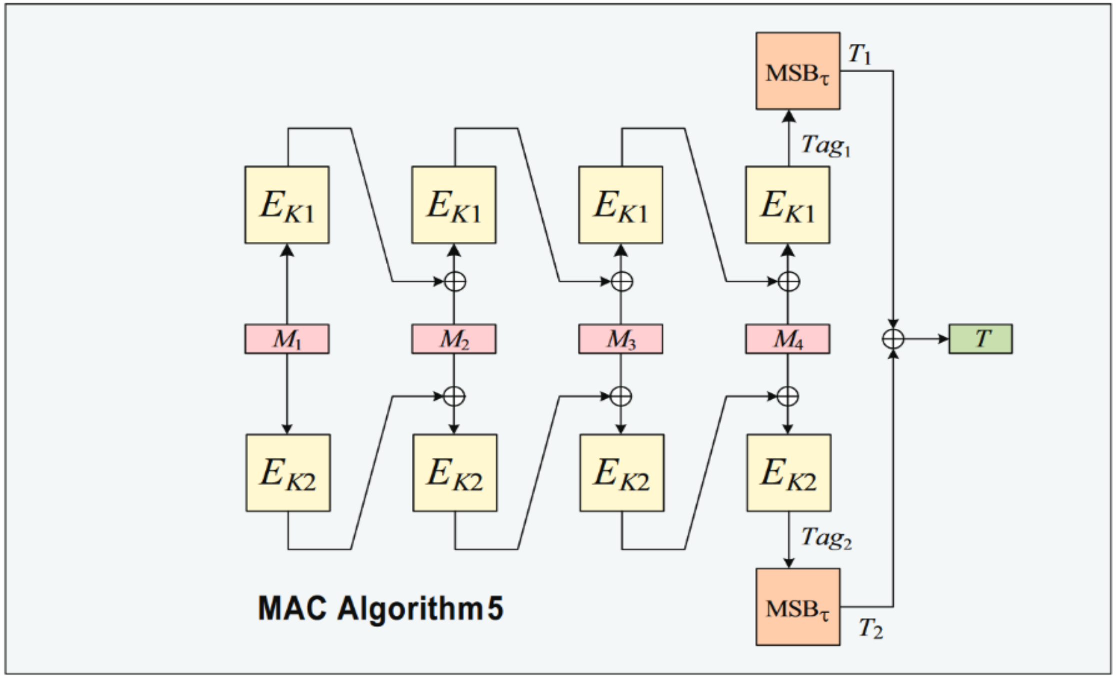
$$E_K(x \oplus c) = E_K(y \oplus f)$$

$x = D_{K'}(T_{ab})$ $y = D_{K'}(T_{de})$

We have simple equation lets one verify the correct guess of K' with trying possible keys.

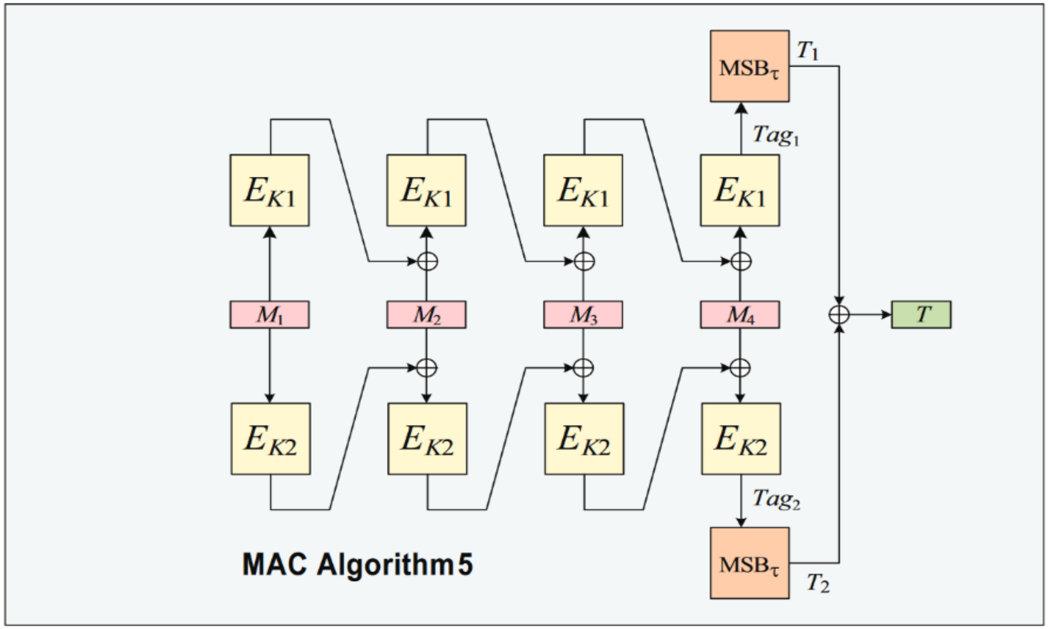
$$E_K(D_{K'}(T_{ab}) \oplus c) = E_K(D_{K'}(T_{cd}) \oplus f).$$

CBC-MAC Algorithms



- Consist of two independent executions of MAC algorithm 1
- Double-piped construction Aim : Improve security beyond the birthday bound.
- The ISO standart says nothing about key seperation.

CBC-MAC Algorithms



Alg	Pad	Sep	Klen	Mlen	#Calls	Goals	Proof	Attack	Ref
5	1	✓	k	$[0..\infty)$	$2\lceil\mu/n\rceil$	C F	—	✓	[105]
5	2	✓	k	$[0..\infty)$	$2\lceil\mu'/n\rceil$	C V	—	✓	[105]
5	3	✓	k	$[0..2^n-1]$	$2\lceil\mu/n\rceil+2$	C V	—	—	—

Attack on Algorithm 5 :

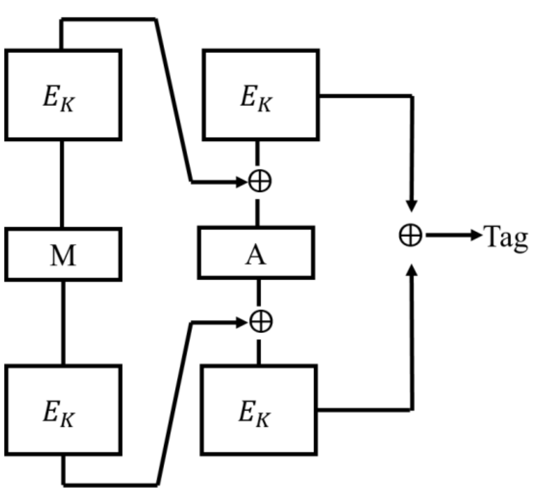
- Find one block messages M and N that yield the same tag (requires $2^{n/2}$ queries)

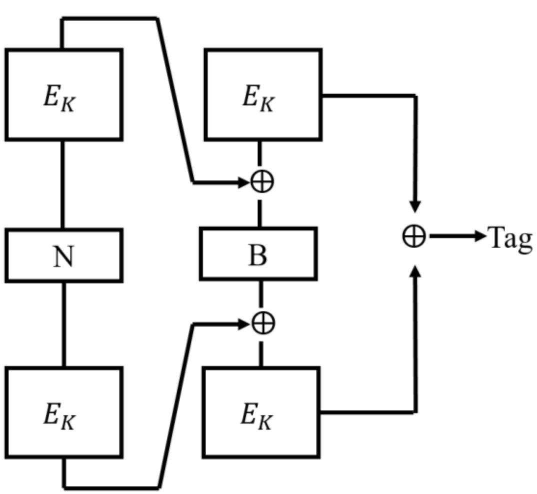
$$T = E_{K1}(M) \oplus E_{K2}(M) = E_{K1}(N) \oplus E_{K2}(N)$$

- Compute MAC of M||A and N||B

$$T_A = E_{K1}(E_{K1}(M) \oplus A) \oplus E_{K2}(E_{K2}(M) \oplus A) \qquad \text{(EQ1)}$$

$$T_B = E_{K1}(E_{K1}(N) \oplus B) \oplus E_{K2}(E_{K2}(N) \oplus B)$$





CBC-MAC Algorithms

- If it so happens that

$$A \oplus B = E_{K1}(M) \oplus E_{K1}(N) = E_{K2}(M) \oplus E_{K2}(N)$$

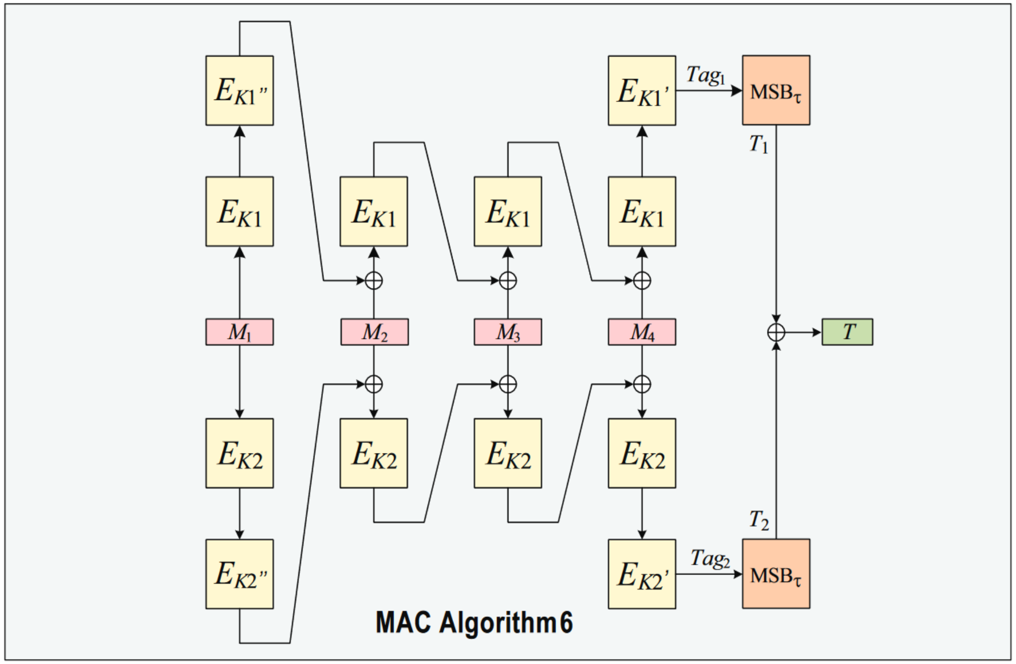
we can write

$$\underbrace{A = E_{K1}(M) \oplus E_{K1}(N) \oplus B}\ ,\ \underbrace{A = E_{K2}(M) \oplus E_{K2}(N) \oplus B}$$

Put in T_A in EQ1

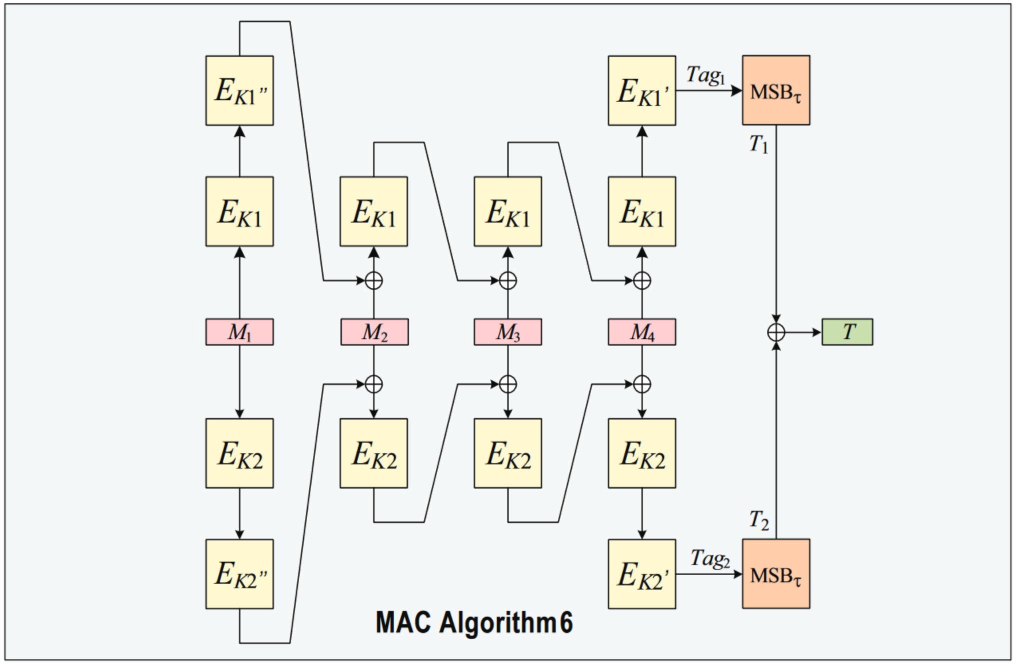
$$T_A = E_{K1}(E_{K1}(M) \oplus A) \oplus E_{K2}(E_{K2}(M) \oplus A)$$

We get $T_A = T_B$



- Same like MAC Algorithm 5 but, instead of xoring two copies of MAC Algorithm 1, two copies of MAC Algorithm 4 are xor'ed together.
- The ISO standart says nothing about key seperation.

CBC-MAC Algorithms



Alg	Pad	Sep	Klen	Mlen	#Calls	Goals	Proof	Attack	Ref
6	1	✓	$2k$	$[n+1..\infty]$	$2\lceil\mu/n\rceil+4$	C F K	✓	—	[206]
6	2	✓	$2k$	$[n..\infty]$	$2\lceil\mu'/n\rceil+4$	C V K	✓	—	[206]
6	3	✓	$2k$	$[0..2^n-1]$	$2\lceil\mu/n\rceil+6$	C V K	✓	—	[206]

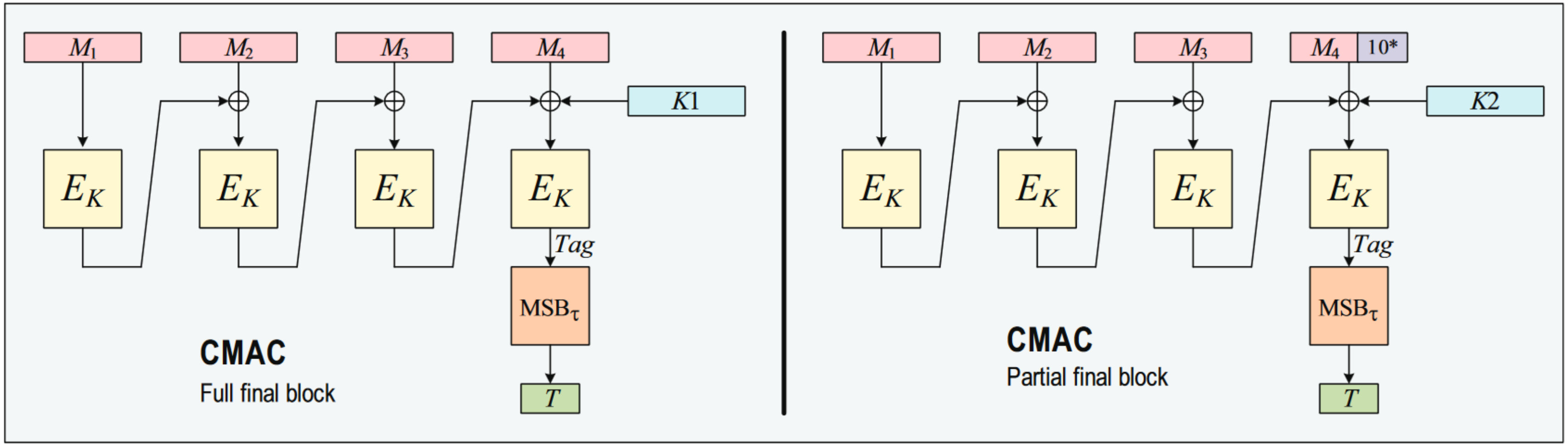
CMAC Mode

NIST SP 800-38B

CMAC Mode

- CMAC is a simple and clean variant of the CBC-MAC.
- No significant identified flaws.
- Nature of CMAC will limit its performance
- According to the spec., the blockcipher underlying CMAC must be NIST approved, which limits the choices to AES, TDEA(triple-DES), and Skipjack.

CMAC Mode



algorithm $\mathrm{CMAC}_K(M)$ *CMAC mode*

$K1 \leftarrow \mathrm{dbl}(E_K(0^n))$

$K2 \leftarrow \mathrm{dbl}(K1)$

if $|M| \in \{n, 2n, 3n, \ldots\}$

 then $K' \leftarrow K1,\ P \leftarrow M$

 else $K' \leftarrow K2,\ P \leftarrow M10^i$ where $i = n - 1 - (|M| \bmod n)$

$M_1 \cdots M_m \leftarrow M$ where $|M_1| = \cdots = |M_m| = n$

for $i \leftarrow 1$ **to** m **do** $C_i \leftarrow E_K(M_i \oplus C_{i-1})$

$T \leftarrow \mathrm{MSB}_\tau(C_m)$

return T

CMAC Mode

algorithm $\mathrm{CMAC}_K(M)$ *CMAC mode*
$K1 \leftarrow \mathrm{dbl}(E_K(0^n))$
$K2 \leftarrow \mathrm{dbl}(K1)$
if $|M| \in \{n, 2n, 3n, \ldots\}$
  then $K' \leftarrow K1,\ P \leftarrow M$
  else $K' \leftarrow K2,\ P \leftarrow M10^i$ where $i = n - 1 - (|M| \bmod n)$
$M_1 \cdots M_m \leftarrow M$ where $|M_1| = \cdots = |M_m| = n$
for $i \leftarrow 1$ **to** m **do** $C_i \leftarrow E_K(M_i \oplus C_{i-1})$
$T \leftarrow \mathrm{MSB}_\tau(C_m)$
return T

$$\mathrm{dbl}(X) = \begin{cases} X \ll 1 & \text{if } \mathrm{MSB}_1(X) = 0, \text{ and} \\ X \ll 1 \oplus 0^{120}10000111 & \text{if } \mathrm{MSB}_1(X) = 1 \end{cases}$$

when $X \in \{0,1\}^{128}$,

$$\mathrm{dbl}(X) = \begin{cases} X \ll 1 & \text{if } \mathrm{MSB}_1(X) = 0, \text{ and} \\ X \ll 1 \oplus 0^{59}11011 & \text{if } \mathrm{MSB}_1(X) = 1 \end{cases}$$

when $X \in \{0,1\}^{64}$.

The best provable-security bound found so far :

Distinguishing probability of CMAC from random permutation

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{CMAC}}(m, q, \sigma) \leq \frac{5\sigma q}{2^n}.$$

Maximal block length

# of oracle queries

Total block length for all queries divided by n.

CMAC Mode

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{CMAC}}(m, q, \sigma) \;\leq\; \frac{5\sigma q}{2^n}.$$

For,

AES-128 and 2^{48} messages of size 1 GB

Maximal adversarial advantage in distinguishing CMAC from a random function would be at most **:** $5 * \left(\frac{8 * 10^9}{128}\right) * 2^{48} * 2^{-128} < 2^{-51}$

HMAC Mode

NIST FISPT 198-1

HMAC Mode

IDEA : Key is hashed together with message.

BUT we should not use

M = $MAC_k(x) = h(k||x)$ directly.

Attack Against Secret Prefix MACs

Alice | **Oscar** | **Bob**

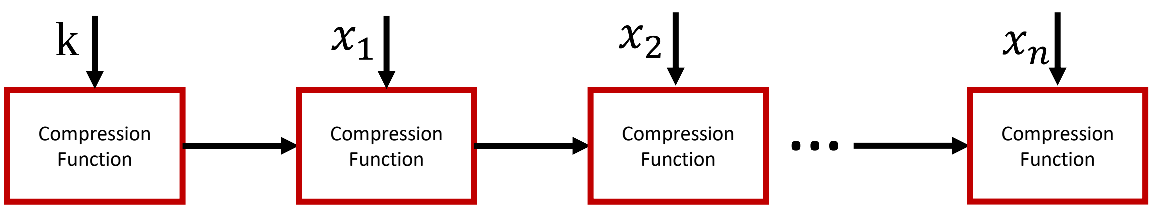
Bob:
$x = (x_1, \ldots, x_n)$
$m = h(k||x_1, \ldots, x_n)$

$\xleftarrow{(x,m)}$

Oscar:
↯ intercept
$x_O = (x_1, \ldots, x_n, x_{n+1})$
$m_O = h(m||x_{n+1})$

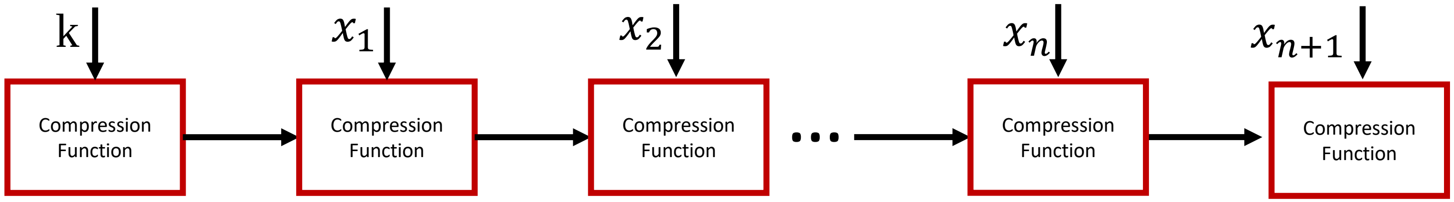
$\xleftarrow{(x_O, m_O)}$

Alice:
$m' = h(k||x_1, \ldots, x_n, x_{n+1})$
since $m' = m_O$
$\Rightarrow$ valid signature!



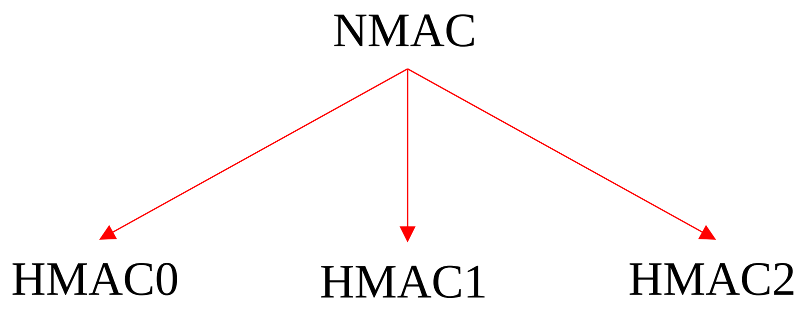
Attack Against Secret Prefix MACs

Alice	**Oscar**	**Bob**
		$x = (x_1, \ldots, x_n)$ $m = h(k \| x_1, \ldots, x_n)$
	↯ intercept $\xleftarrow{(x,m)}$	
	$x_O = (x_1, \ldots, x_n, x_{n+1})$ $m_O = h(m \| x_{n+1})$	
$\xleftarrow{(x_O, m_O)}$		
$m' = h(k \| x_1, \ldots, x_n, x_{n+1})$ since $m' = m_O$ $\Rightarrow$ valid signature!		



HMAC Mode

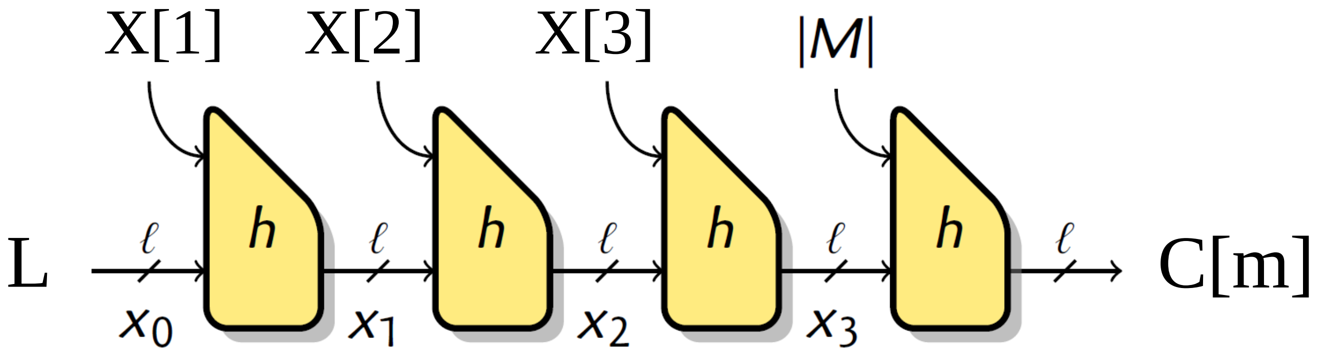
- Standardized in NIST FISPT 198-1 (FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION)
- HMAC is used to refer several algorithms : HMAC0 , HMAC1 , HMAC2
- All algorithms have common core but different key lengths and key derivation function.



HMAC Mode

According to FIPS 198-1, the hash function must be an approved, iterated one.

algorithm $h_L^*(X)$
$C[0] \leftarrow L$
$X[1] \ldots X[m] \leftarrow X$ where $|X[i]| = b$
for $i \leftarrow 1$ **to** m **do** $C[i] \leftarrow h(C[i-1], X[i])$
return $C[m]$



HMAC Mode

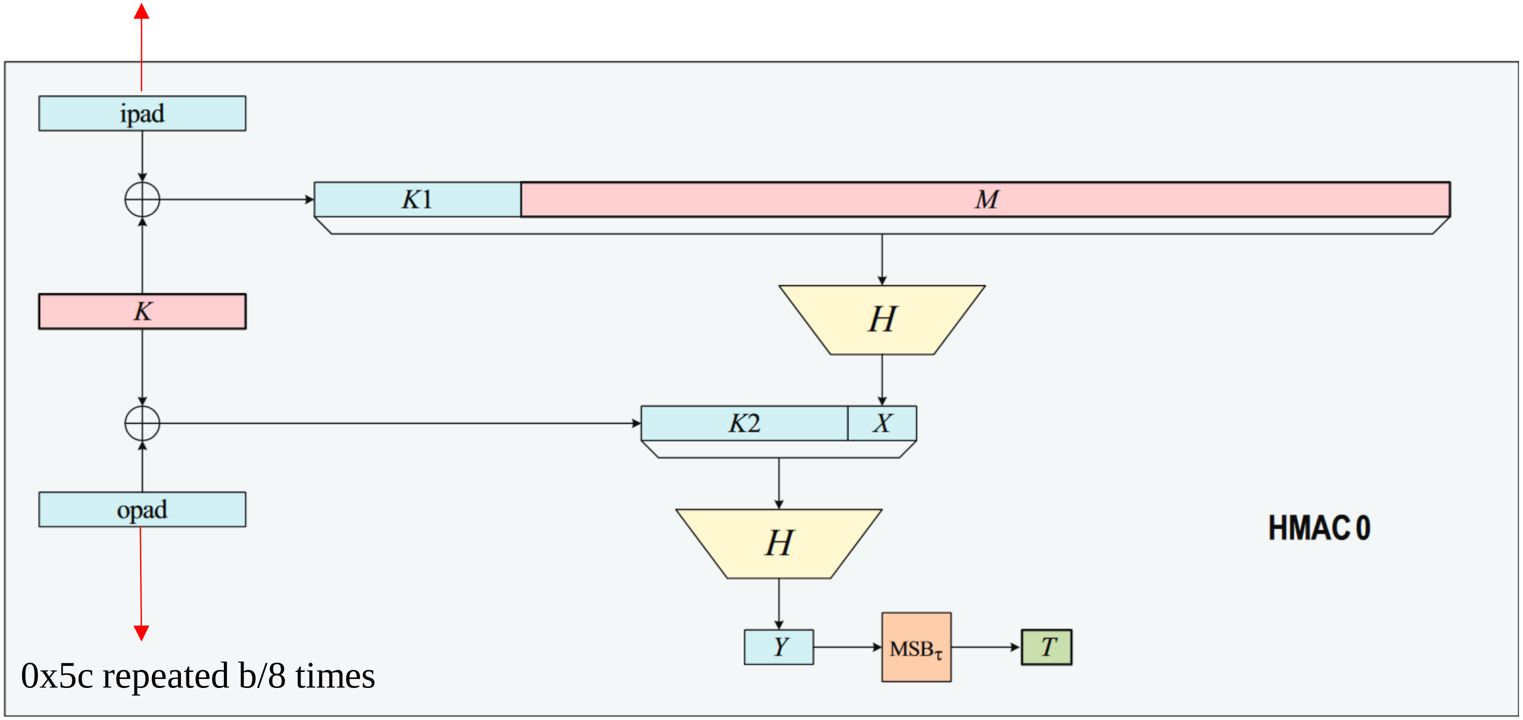
According to FIPS(FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION) 198-1, the hash function must be an approved, iterated one.

At present, there are five NIST-approved hash functions (all of them iterated):

SHA-1, SHA-224, SHA-256, SHA-384, SHA-512

HMAC Mode

0x36 repeated b/8 times



0x5c repeated b/8 times

HMAC Mode

algorithm $\mathrm{NMAC}_{L1 \,\|\, L2}(M)$
$X \leftarrow h^*_{L1}(M \,\|\, \mathsf{pad}(b + |M|))$
$Y \leftarrow h_{L2}(X \,\|\, \mathsf{pad}(b + c))$
return Y

b bit key needed

algorithm $\mathrm{HMAC0}_K(M)$
$K1 \leftarrow K \oplus \mathsf{ipad}$
$K2 \leftarrow K \oplus \mathsf{opad}$
$X \leftarrow H(K1 \,\|\, M)$
$Y \leftarrow H(K2 \,\|\, X)$
$T \leftarrow \mathrm{MSB}_\tau(Y)$
return T

algorithm $\mathrm{KDF0}(K)$
$L1 \leftarrow h_{IV}(K \oplus \mathsf{ipad})$
$L2 \leftarrow h_{IV}(K \oplus \mathsf{opad})$
return $L1 \,\|\, L2$

algorithm $\mathrm{HMAC0}_K(M)$
$L1 \,\|\, L2 \leftarrow \mathrm{KDF0}(K)$
$Y \leftarrow \mathrm{NMAC}_{L1 \,\|\, L2}(M)$
$T \leftarrow \mathrm{MSB}_\tau(Y)$
return T

c bit key needed

algorithm $\mathrm{HMAC1}_K(M)$
$K \leftarrow K \,\|\, 0^{b-c}$
$T \leftarrow \mathrm{HMAC0}_K(M)$
return T

algorithm $\mathrm{KDF1}(K)$
$K \leftarrow K \,\|\, 0^{b-c}$
$L1 \,\|\, L2 \leftarrow \mathrm{KDF0}(K)$
return $L1 \,\|\, L2$

algorithm $\mathrm{HMAC1}_K(M)$
$L1 \,\|\, L2 \leftarrow \mathrm{KDF1}(K)$
$Y \leftarrow \mathrm{NMAC}_{L1 \,\|\, L2}(M)$
$T \leftarrow \mathrm{MSB}_\tau(Y)$
return T

No restriction for key size

algorithm $\mathrm{HMAC2}_K(M)$
if $|K| > b$ **then** $K \leftarrow H(K)$
$K \leftarrow K \,\|\, 0^{b-|K|}$
$T \leftarrow \mathrm{HMAC0}_K(M)$
return T

algorithm $\mathrm{KDF2}(K)$
if $|K| > b$ **then** $K \leftarrow H(K)$
$K \leftarrow K \,\|\, 0^{b-|K|}$
$L1 \,\|\, L_2 \leftarrow \mathrm{KDF0}(K)$
return $L1 \,\|\, L2$

algorithm $\mathrm{HMAC2}_K(M)$
$L1 \,\|\, L2 \leftarrow \mathrm{KDF2}(K)$
$Y \leftarrow \mathrm{NMAC}_{L1 \,\|\, L2}(M)$
$T \leftarrow \mathrm{MSB}_\tau(Y)$
return T

b= output size of the hash function
c= input size of the hash function

HMAC Mode

Compression function under the Hash is good PRF $\Rightarrow$ NMAC is good PRF

KDF is good PRG

$\Rightarrow$ HMAC is good PRF

Right now(2011), we have no attacks refuting this assumption, not only for SHA1 but even for MD5.

Collisions have been found for MD5 but this has not given rise to better-than-birthday PRF-attacks on HMAC-MD5.

HMAC Mode

Mode	Type	q (# of queries)	t (Time complexity)	RKA? (Related key attack)	Who
NMAC-MD4	recover $K1, K2$	2^{77}	2^{77}	No	[196]
HMAC-MD4	DH	2^{58}		No	[53]
HMAC-MD4	recover $K1$	2^{63}		No	[53]
HMAC-MD4	forgery	2^{58}		No	[111]
HMAC-MD4	recover $K1, K2$	2^{72}	2^{77}	No	[196]
NMAC-MD5	recover $L1$	2^{47}		Yes	[53]
NMAC-MD5	recover $L1 \parallel L2$	2^{51}	2^{100}	Yes	[70]
NMAC-MD5	recover $L1 \parallel L2$	2^{76}	2^{77}	Yes	[70]
HMAC-MD5	DH	2^{97}	2^{97}	No	[200]

References

1. Rogaway, Phillip. "Evaluation of some blockcipher modes of operation." (2011).

2. R. C. Merkle and M. E. Hellman, “On the security of multiple encryption,” Comm. of the ACM, vol. 24, no. 7, pp. 465–467, Jul. 1981.

3. Joux A., Poupard G., Stern J. (2003) New Attacks against Standardized MACs. In: Johansson T. (eds) Fast Software Encryption. FSE 2003. Lecture Notes in Computer Science, vol 2887. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-39887-5_13