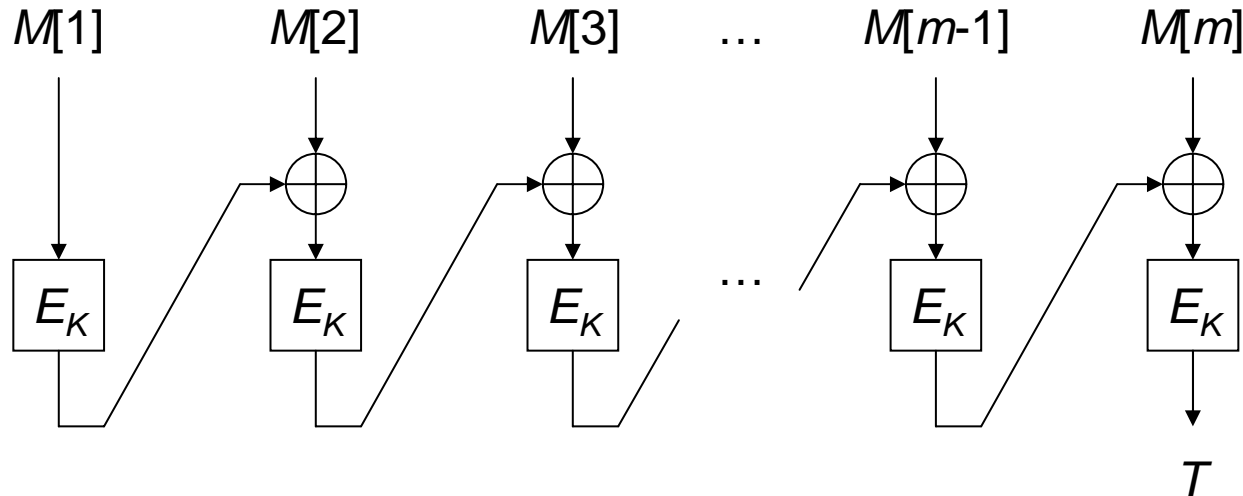


On the Impact of Key Check Value on CBC MACs and Others

Tetsu Iwata

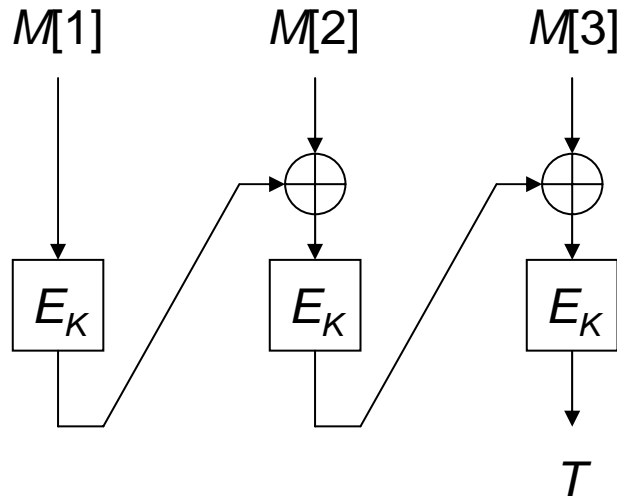
Dagstuhl Seminar, Symmetric Cryptography
13th January 2009

CBC MAC

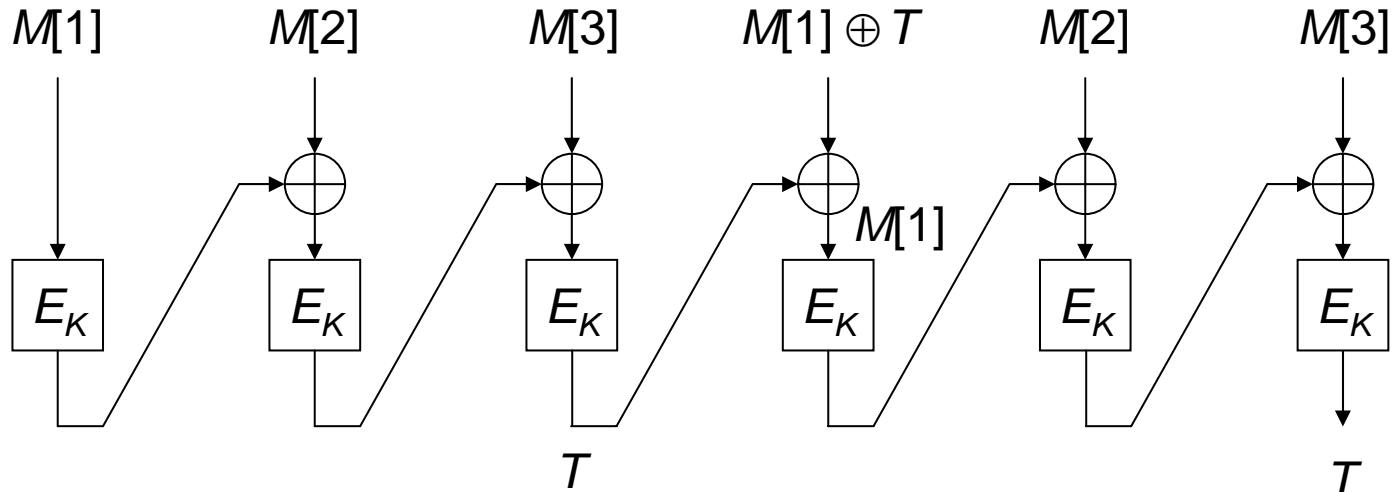


- $M=(M[1],M[2],\dots,M[m])$: input message
- T : Tag
- Fixed-Input-Length PRF if E is a PRP [BKR '94, BPR '05]
- Provably implies that it is a secure MAC (over a fixed-length messages)
- It allows forgery attacks for variable-length messages

Length-Extension Attack on CBC MAC



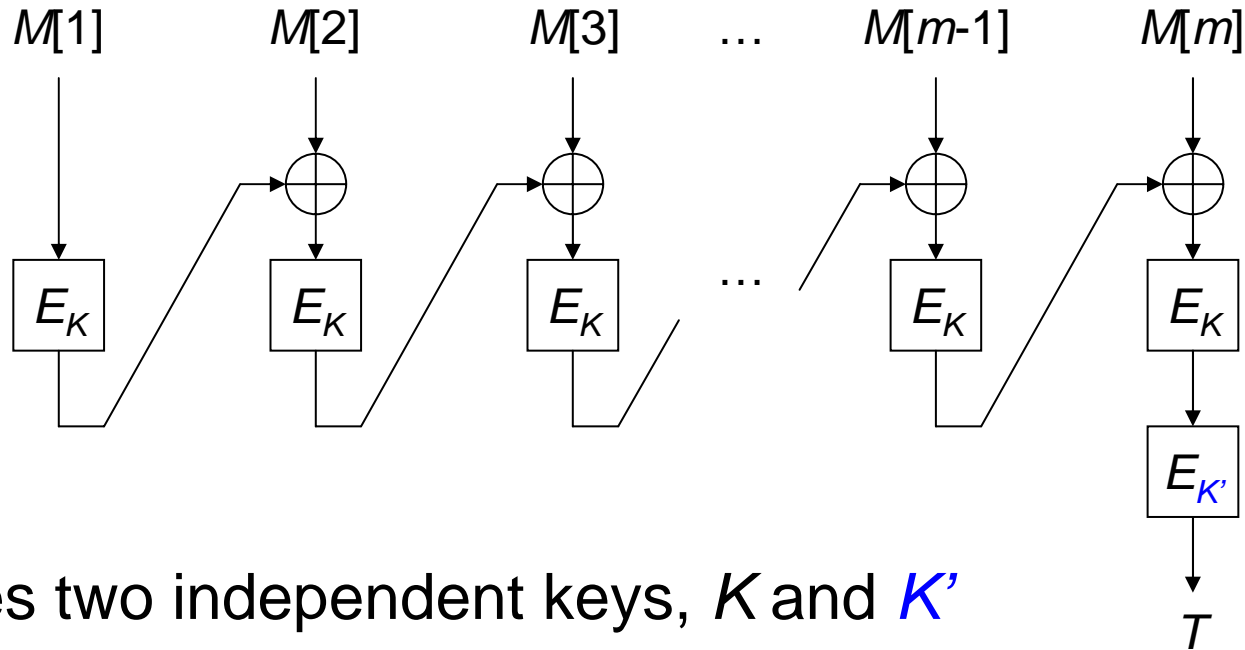
- For given $(M[1], M[2], M[3])$, T ,
- $(M[1], M[2], M[3], M[1] \oplus T, M[2], M[3])$, T is a valid (message, Tag) pair



Variants of CBC MAC

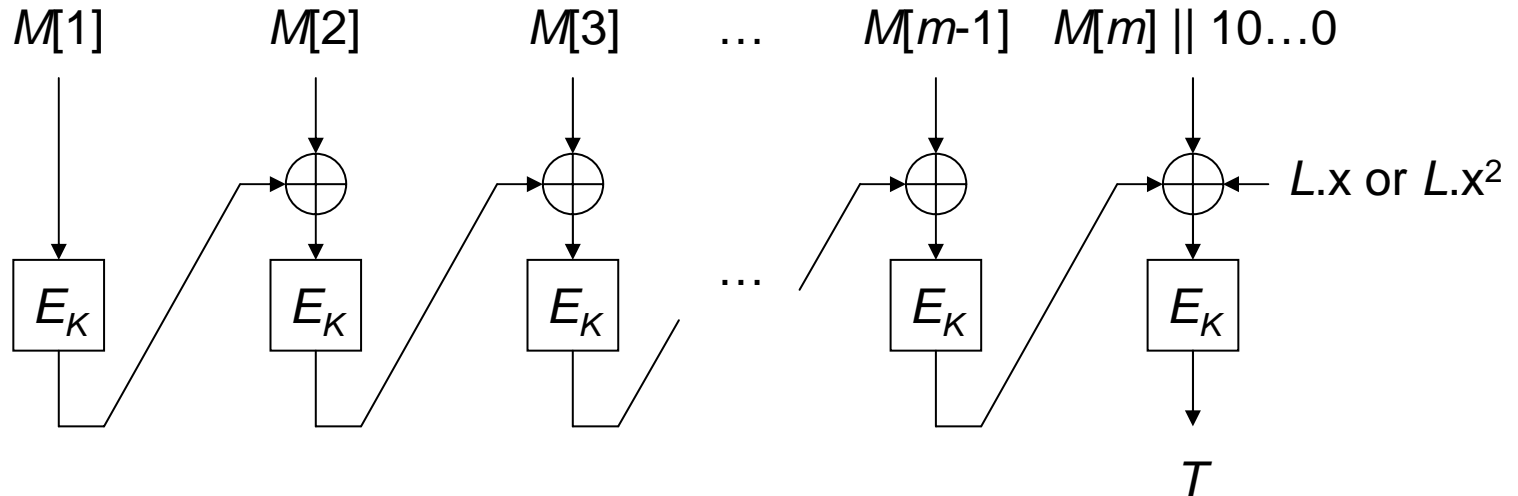
- EMAC [RIPE-RACE '95, PR '00, ISO 9797-1]
- CMAC [NIST '05]
- many others...
 - retail MAC, XCBC, TMAC,...
- Variable-Input-Length PRFs if E is a PRP
- Provably implies the security as a MAC (over a variable-length messages)

EMAC



- Uses two independent keys, K and K'
- $\text{EMAC}_{K,K'} : (\{0,1\}^n)^* \rightarrow \{0,1\}^n$

CMAC (Cipher-based MAC)



- Uses single key K , $\text{CMAC}_K : \{0,1\}^* \rightarrow \{0,1\}^n$
- $L = E_K(0^n)$
- $L.x = L \times (00\dots 010)$ in $\text{GF}(2^n)$
- $L.x^2 = (L.x).x$
- use $L.x$ if $M[m]$ is n bits, otherwise pad $M[m]$ and use $L.x^2$

Key Check Value

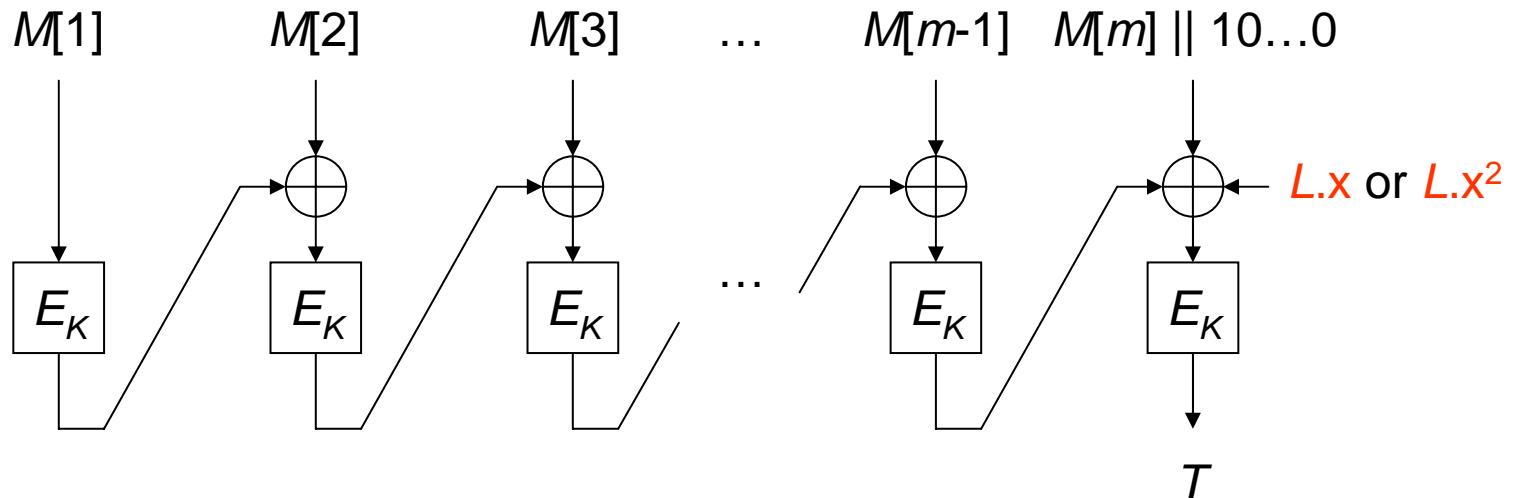
- ANSI X9.24, Annex C, “Retail Financial Services, Symmetric Key Management”
- “The key check value is the left-most six hexadecimal digits from the ciphertext produced by using the DES in ECB mode to encrypt a 64-bit binary zero value with the subject key or key component.”
- “The check value process may be simplified ... by limiting the check value to the left-most four or six hexadecimal digits of the ciphertext.”

Key Check Value

- $KCV = \text{msb}(s, E_K(0^n))$
- $s = 16$ or 24 (for $n=64$)
- (only defined for DES and Triple-DES)
- used as an ID for the key K in a banking system
- the adversary may learn this value
- Proofs of security for EMAC and CMAC do not take this into account
- What is the impact of the use of KCV?

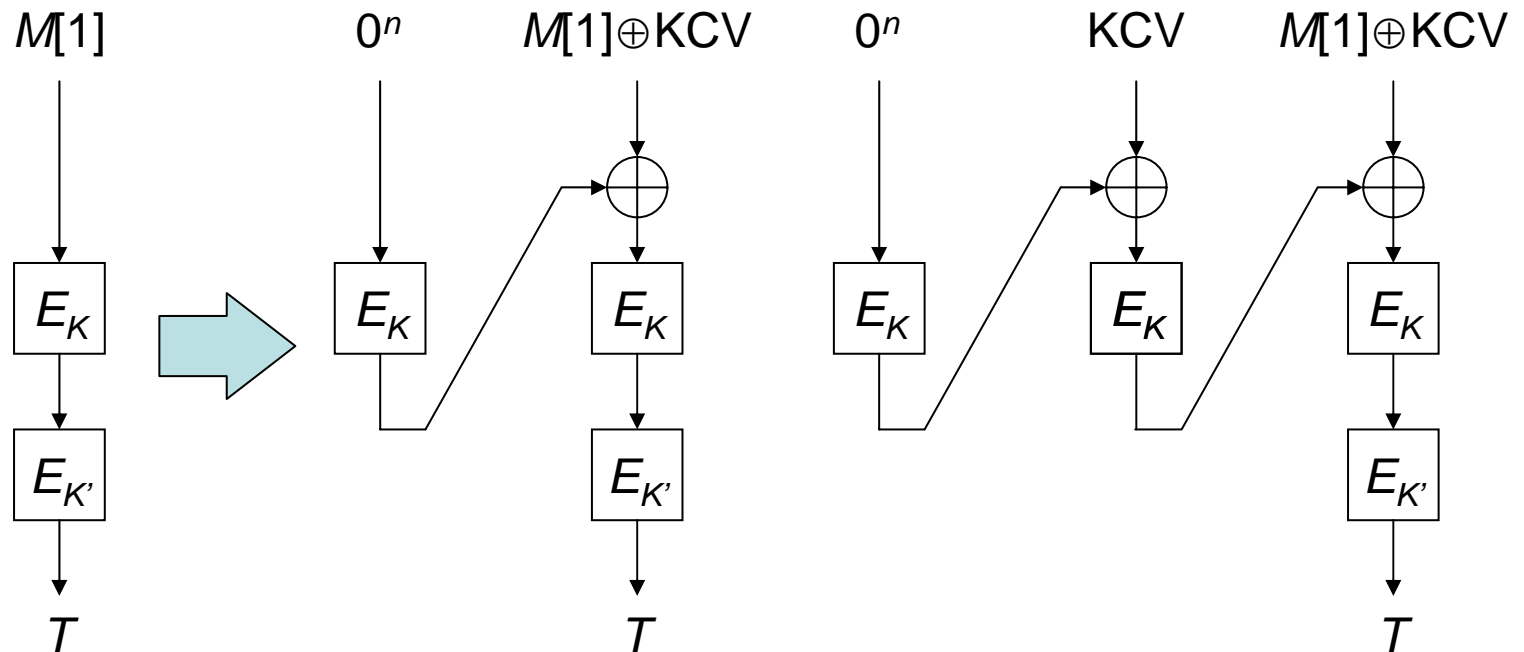
Case $s = n$

- CMAC
 - $KCV = \text{msb}(s, E_K(0^n))$
 - $L = E_K(0^n)$ is known, then $L \cdot x$ and $L \cdot x^2$ are known
 - reduces to CBC MAC
 - length-extension attack



Case $s = n$

- $\text{EMAC}_{K,K'}$
 - $\text{KCV} = E_K(0^n)$ and $\text{KCV}' = E_{K'}(0^n)$



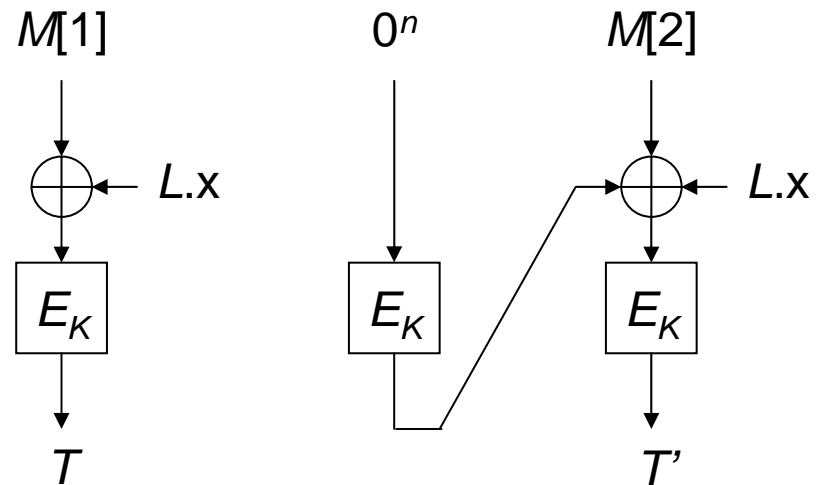
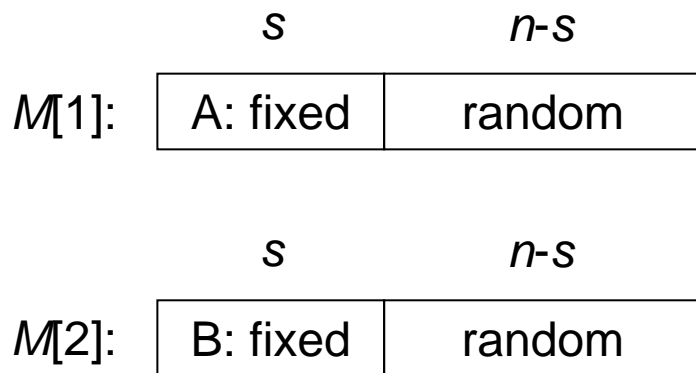
CMAC with $s < n$

- Trivial attack:
 - guess the missing $n-s$ bits of L and try length-extension attack
 - $\Pr(\text{success}) = 1/2^{n-s}$

CMAC with $s < n$

- Better attack --- birthday attack

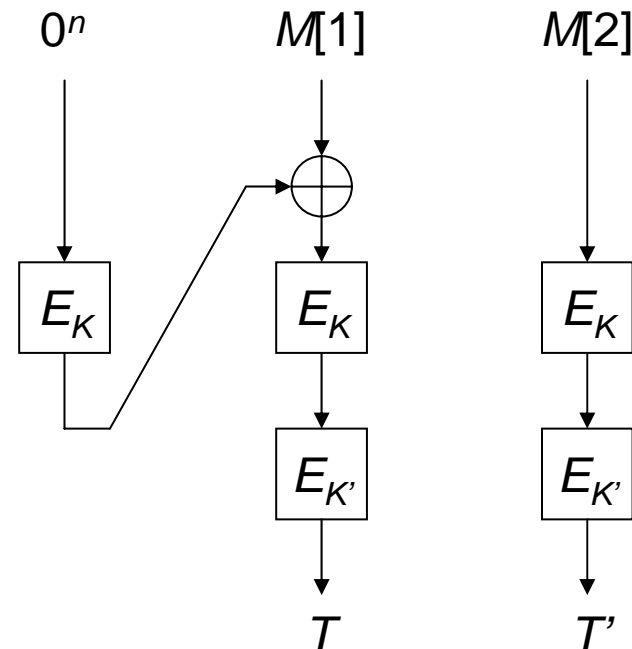
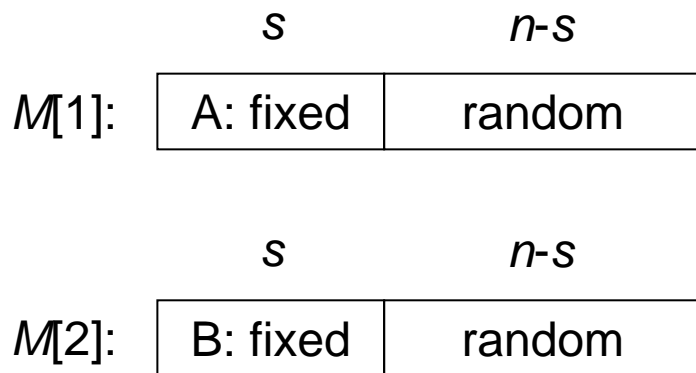
- A: arbitrarily fixed constant
- B: A xor msb(s , L)



- ask $2^{(n-s)/2}$ different $M[1]$'s and $2^{(n-s)/2}$ different $(0^n, M[2])$'s
 - with high probability, $T = T'$
- distinguishing attack with $O(2^{(n-s)/2})$ queries
 - $L (=M[1] \text{ xor } M[2])$ is known
 - length-extension attack

EMAC with $s < n$

- The **same** attack can be applied
 - A: arbitrarily fixed constant
 - B: $A \text{ xor KCV}$, $\text{KCV} = \text{msb}(s, E_K(0^n))$



- ask $2^{(n-s)/2}$ different $(0^n, M[1])$'s and $2^{(n-s)/2}$ different $M[2]$'s
 - with high probability, $T = T'$
- distinguishing attack with $O(2^{(n-s)/2})$ queries
 - $E_K(0^n)$ is known
 - forgery attack

Can We Do Better?

- for EMAC and CMAC, $O(2^{(n-s)/2})$ attacks exist
- can we do better? --- No
- Theorem:

$$\text{Adv}_{\text{EMAC}}^{\text{prf}} \leq \frac{4\sigma^2}{2^{n-s}} \quad \text{and} \quad \text{Adv}_{\text{CMAC}}^{\text{prf}} \leq \frac{5\sigma^2}{2^{n-s}}$$

σ is the total number of message blocks

Implication

- does not result in a total security loss
- security is lost by $s/2$ bits
 - for $n=64$,
 - if $s=0$, then the best attack needs $O(2^{32})$
 - if $s=16$ $O(2^{24})$
 - if $s=24$ $O(2^{20})$
 - for $n=128$ (not defined in ANSI X9.24, Annex C),
 - if $s=0$, then the best attack needs $O(2^{64})$
 - if $s=16$ $O(2^{58})$
 - if $s=24$ $O(2^{52})$
 - if $s=32$ $O(2^{46})$
 - if $s=48$ $O(2^{40})$
- can still be used in practice (depending on applications)

Fixes?

- Option 1
 - change the definition of KCV
- Option 2
 - change the definition of MAC

Option 1

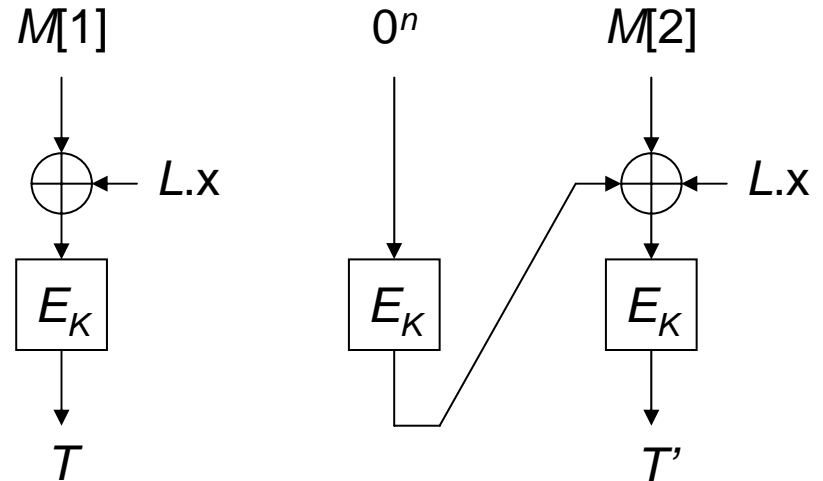
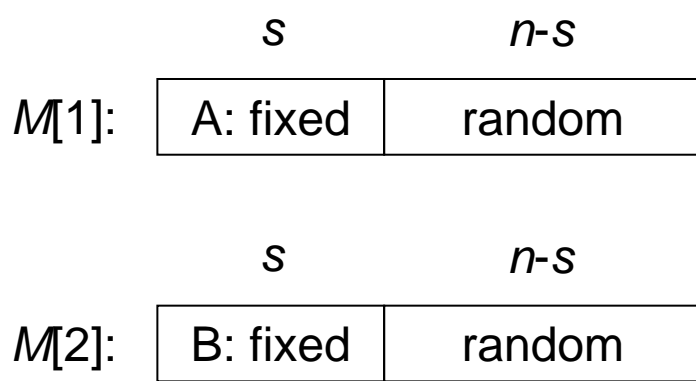
- Use a hash function to derive KCV
- it works (in theory)
- ... but maybe impractical
 - the standard is widely deployed

Option 2

- Solution 1
 - a solution by tweaking CMAC
- Solution 2
 - a solution that can be used for general blockcipher mode

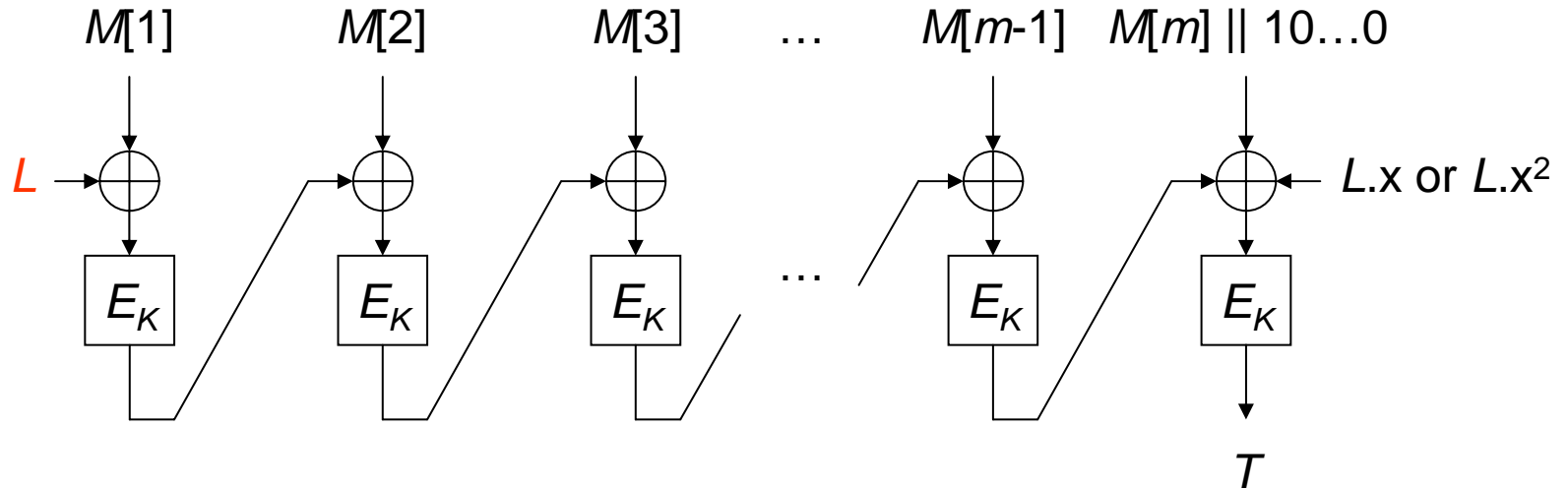
Tweaking CMAC --- First Try

- First try --- $L = E_K(1^n)$ instead of $L = E_K(0^n)$
 - $KCV = \text{msb}(s, E_K(0^n))$
 - A: arbitrarily fixed constant
 - B: A xor KCV



- the same attack as before still works

Tweaking CMAC --- Second Try



- $L = E_K(1^n)$
- $KCV = \text{msb}(s, E_K(0^n))$

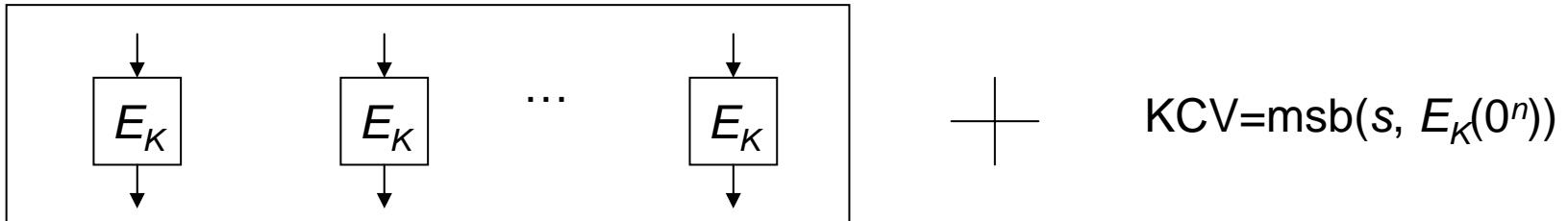
• Theorem:

$$\text{Adv}_{\text{CMAC-second-try}}^{\text{prf}} \leq \frac{6\sigma^2}{2^n}$$

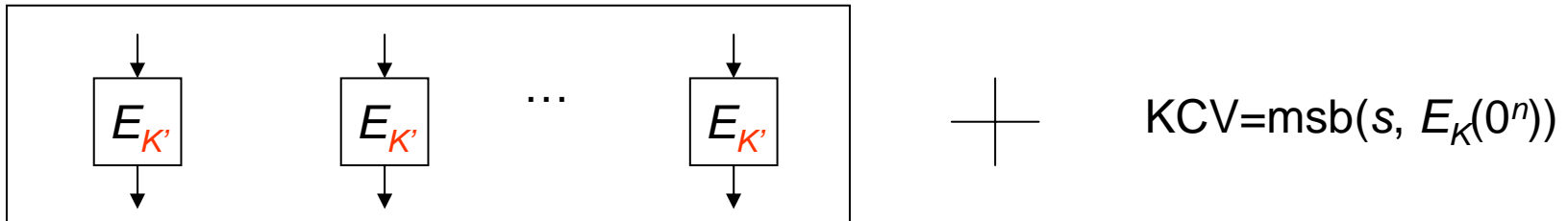
σ is the total number of message blocks

For General Use

Blockcipher mode + proof of security with a PRP assumption

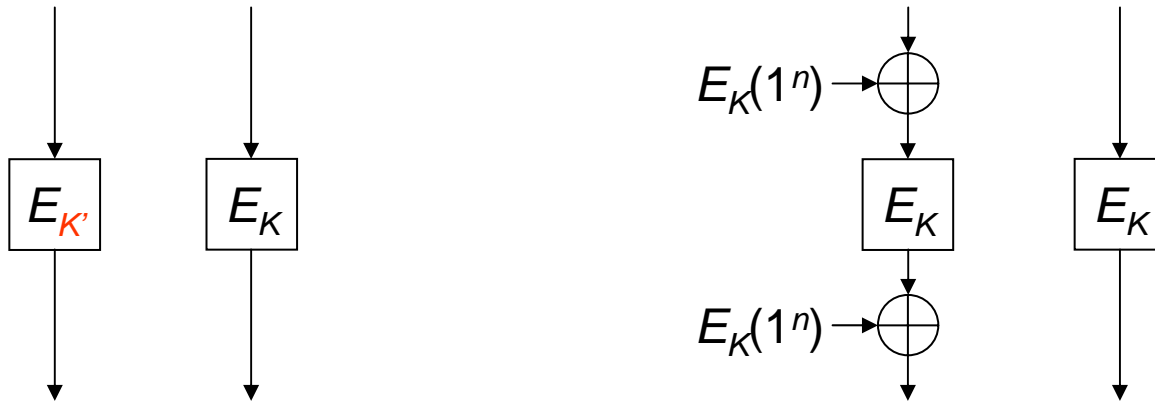


- proof of security is lost if KCV is introduced



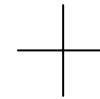
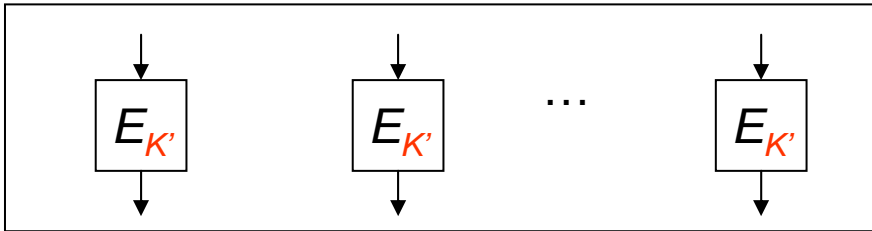
- no effect on the proof of security,
- but KCV does not check K' ...

For General Use

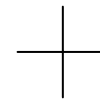
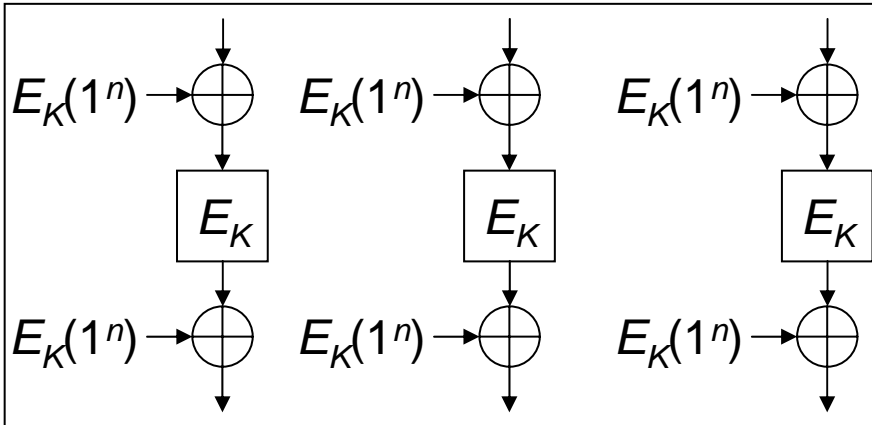


- These two pairs of oracles are indistinguishable as long as the adversary is not allowed to ask 1^n to the right oracle

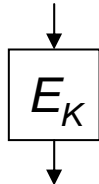
For General Use

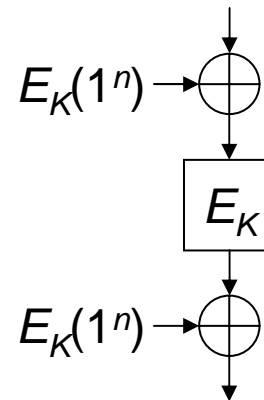


$$\text{KCV} = \text{msb}(s, E_{K'}(0^n))$$



$$\text{KCV} = \text{msb}(s, E_K(0^n))$$

Replace  by



Summary

- KCV affects on the security of blockcipher modes
- EMAC and CMAC have a similar security loss
- the impact is limited in practice (unless s is not very big)
- suggested possible fixes