Message Integrity

This slide is made based the online course of Cryptography by Dan Boneh

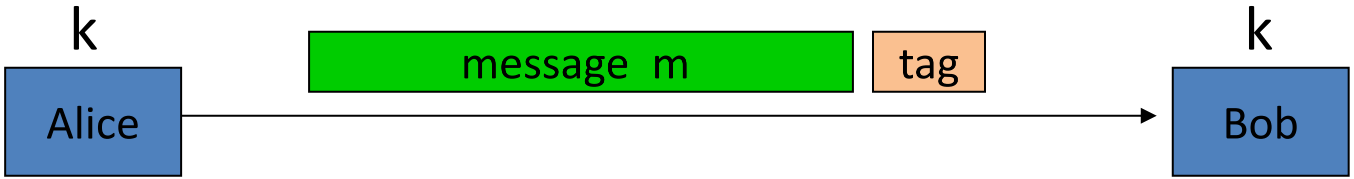
Message Integrity

Goal: **integrity**, no confidentiality.

Examples:

- Protecting public binaries on disk.
- Protecting banner ads on web pages.

Message integrity: MACs



Generate tag:

$tag \leftarrow S(k, m)$

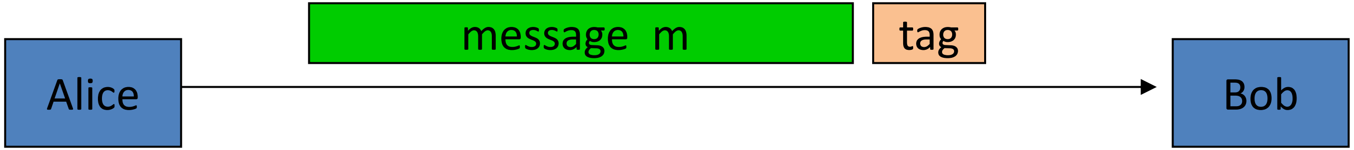
Verify tag:

$V(k, m, tag) \stackrel{?}{=}$ `yes'

Def: **MAC** $I = (S,V)$ defined over (K,M,T) is a pair of algs:

- $S(k,m)$ outputs t in T
- $V(k,m,t)$ outputs `yes' or `no'

Integrity requires a secret key



Generate tag:

tag $\leftarrow$ CRC(m)

Verify tag:

V(m, tag) $\stackrel{?}{=}$ `yes'

- Attacker can easily modify message m and re-compute CRC.
- CRC designed to detect **random**, not malicious errors.

Secure MACs

Attacker's power: **chosen message attack**

- for $m_1, m_2, \ldots, m_q$ attacker is given $t_i \leftarrow S(k, m_i)$

Attacker's goal: **existential forgery**

- produce some **<u>new</u>** valid message/tag pair (m,t).

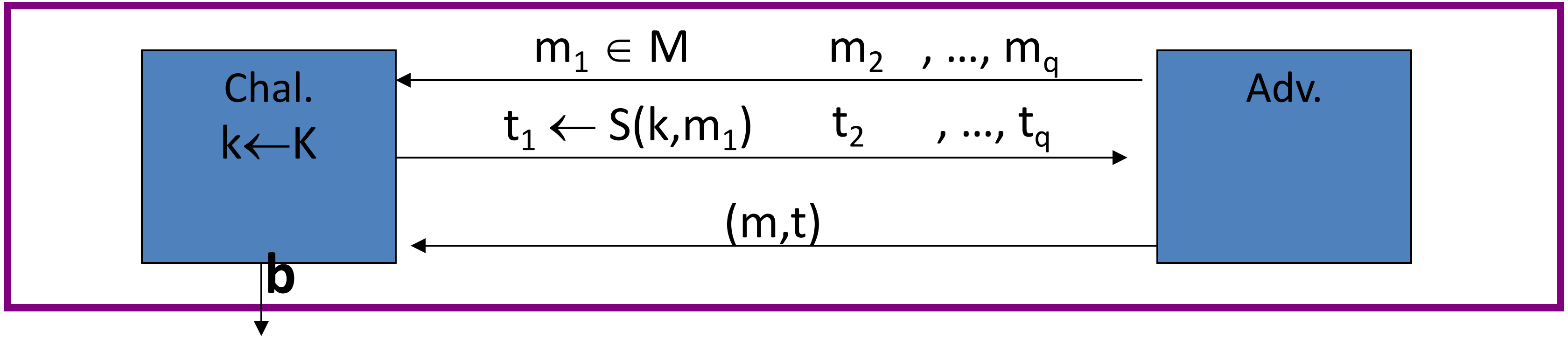
$$(m,t) \notin \{ (m_1,t_1), \ldots, (m_q,t_q) \}$$

$\Rightarrow$ attacker cannot produce a valid tag for a new message

$\Rightarrow$ given (m,t) attacker cannot even produce (m,t') for $t' \neq t$

Secure MACs

- For a MAC I=(S,V) and adv. A define a MAC game as:



b=1 if $V(k,m,t) =$ `yes' and $(m,t) \notin \{ (m_1,t_1), \dots, (m_q,t_q) \}$

b=0 otherwise

Def: I=(S,V) is a **secure MAC** if for all "efficient" A:

$\mathrm{Adv}_{\mathrm{MAC}}[A,I] = \Pr[\text{Chal. outputs } 1]$ is "negligible."

Let I = (S,V) be a MAC.

Suppose an attacker is able to find $m_0 \neq m_1$ such that

$$S(k, m_0) = S(k, m_1) \quad \text{for } ½ \text{ of the keys } k \text{ in } K$$

Can this MAC be secure?

- ○ Yes, the attacker cannot generate a valid tag for m_0 or m_1
- ○ No, this MAC can be broken using a chosen msg attack
- ○ It depends on the details of the MAC
- ○

Let I = (S,V) be a MAC.

Suppose S(k,m) is always 5 bits long

Can this MAC be secure?

- ◯ No, an attacker can simply guess the tag for messages
- ◯ It depends on the details of the MAC
- ◯ Yes, the attacker cannot generate a valid tag for any message
- ◯

Example: protecting system files

Suppose at install time the system computes:

filename F_1	filename F_2	$\cdots$	filename F_n
$t_1 = S(k,F_1)$	$t_2 = S(k,F_2)$		$t_n = S(k,F_n)$

k derived from user's password

Later a virus infects system and modifies system files

User reboots into clean OS and supplies his password

– Then: secure MAC $\Rightarrow$ all modified files will be detected

End of Segment

Message Integrity

MACs based on PRFs

Review: Secure MACs

MAC: signing alg. **$S(k,m) \longrightarrow t$** and verification alg. **$V(k,m,t) \longrightarrow 0,1$**

Attacker's power: **chosen message attack**

- for $m_1, m_2, \ldots, m_q$ attacker is given $t_i \leftarrow S(k, m_i)$

Attacker's goal: **existential forgery**

- produce some **new** valid message/tag pair (m,t).

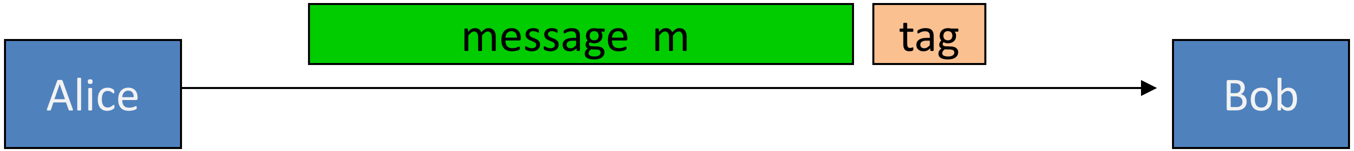
$$(m,t) \notin \{ (m_1,t_1), \ldots, (m_q,t_q) \}$$

$\Rightarrow$ attacker cannot produce a valid tag for a new message

Secure PRF $\Rightarrow$ Secure MAC

For a PRF **$F: K \times X \longrightarrow Y$** define a MAC $I_F = (S,V)$ as:

- $S(k,m) := F(k,m)$
- $V(k,m,t)$: output `yes' if $t = F(k,m)$ and `no' otherwise.



tag ← F(k,m)

accept msg if

tag = F(k,m)

A bad example

Suppose $\mathbf{F: K \times X \longrightarrow Y}$ is a secure PRF with $\mathbf{Y = \{0,1\}^{10}}$

Is the derived MAC I_F a secure MAC system?

- ○ Yes, the MAC is secure because the PRF is secure
- √ ○ No tags are too short: anyone can guess the tag for any msg
- ○ It depends on the function F
- ○

$\mathrm{Adv}[A, I_F] = 1/1024$

Security

<u>Thm</u>: If $\mathbf{F: K \times X \longrightarrow Y}$ is a secure PRF and $1/|Y|$ is negligible (i.e. $|Y|$ is large) then I_F is a secure MAC.

In particular, for every eff. MAC adversary A attacking I_F there exists an eff. PRF adversary B attacking F s.t.:

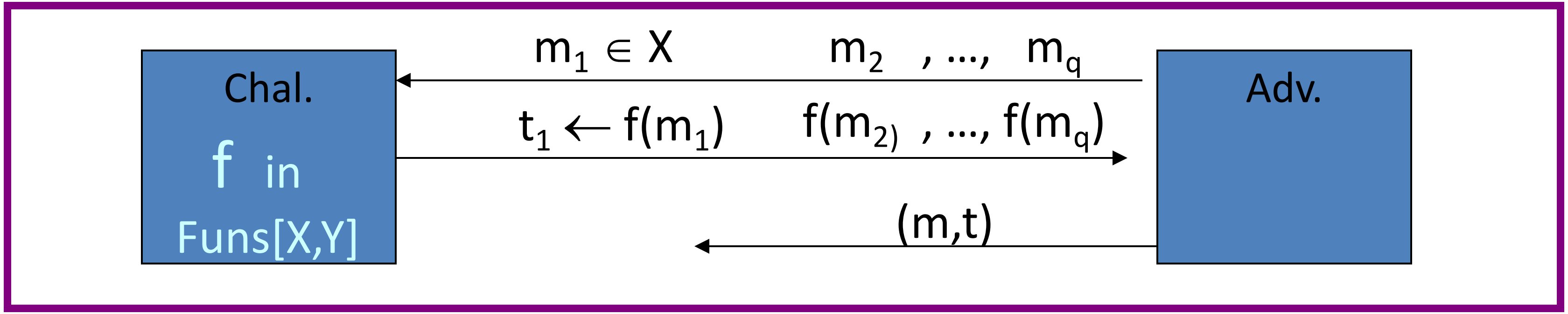
$$\mathrm{Adv}_{\mathrm{MAC}}[A, I_F] \leq \mathrm{Adv}_{\mathrm{PRF}}[B, F] + 1/|Y|$$

$\Rightarrow$ I_F is secure as long as $|Y|$ is large, say $|Y| = 2^{80}$.

Proof Sketch

Suppose $f: \mathbf{X} \longrightarrow \mathbf{Y}$ is a truly random function

Then MAC adversary A must win the following game:



A wins if $\mathbf{t = f(m)}$ and $\mathbf{m \notin \{ m_1 , \ldots , m_q \}}$

$\Rightarrow$ $\Pr[\text{A wins}] = 1/|Y|$ same must hold for $F(k,x)$

Examples

- AES: a MAC for 16-byte messages.

- Main question: how to convert Small-MAC into a Big-MAC ?

- Two main constructions used in practice:
  - **CBC-MAC**   (banking – ANSI X9.9, X9.19,   FIPS 186-3)
  - **HMAC** (Internet protocols: SSL, IPsec, SSH, …)

- Both convert a small-PRF into a big-PRF.

Truncating MACs based on PRFs

Easy lemma: suppose $F: K \times X \longrightarrow \{0,1\}^n$ is a secure PRF.

Then so is $F_t(k,m) = \underbrace{F(k,m)[1...t]}_{\text{First t-bit of output}}$ for all $1 \le t \le n$

$\Rightarrow$ if (S,V) is a MAC is based on a secure PRF outputting n-bit tags

the truncated MAC outputting w bits is secure

... as long as $1/2^w$ is still negligible (say $w \ge 64$)

End of Segment

Message Integrity

CBC-MAC and NMAC

MACs and PRFs

Recall: secure PRF **F** $\Rightarrow$ secure MAC, as long as $|Y|$ is large

$$S(k, m) = F(k, m)$$

Our goal:

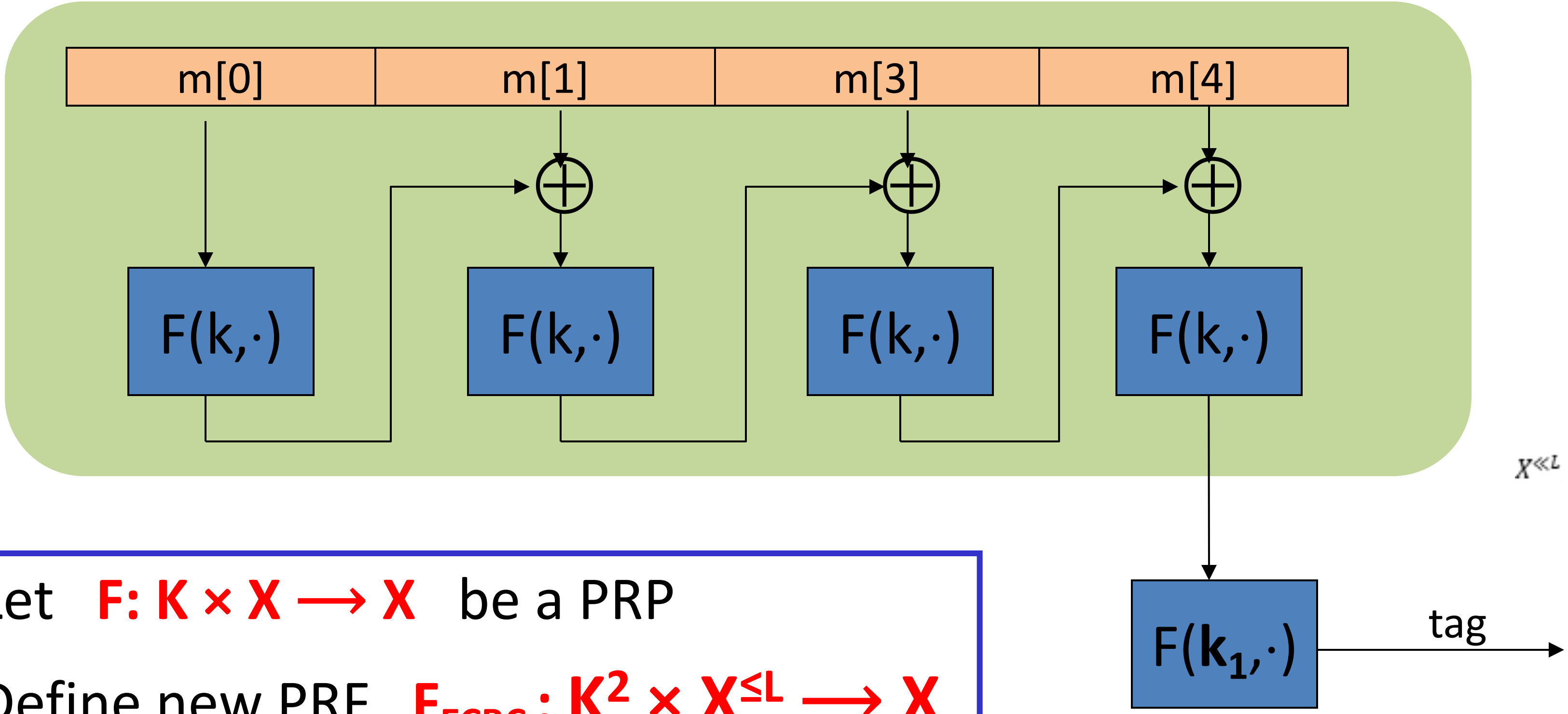
given a PRF for short messages (AES)

construct a PRF for long messages

From here on let $X = \{0,1\}^n$ (e.g. $n=128$)

Construction 1: encrypted CBC-MAC

raw CBC



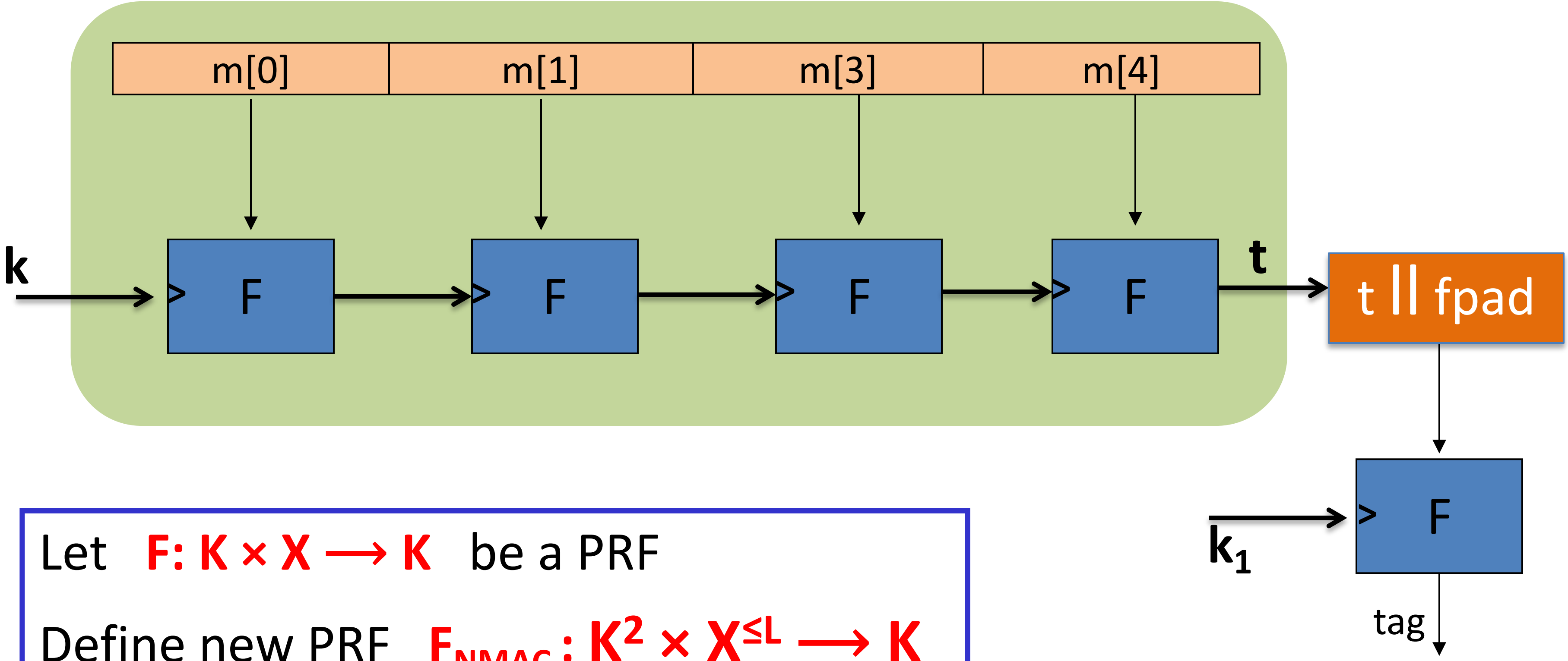
$$X^{\leq L} = \bigcup_{i=1}^{L} X^i$$

Let $\mathbf{F: K \times X \longrightarrow X}$ be a PRP

Define new PRF $\mathbf{F_{ECBC} : K^2 \times X^{\leq L} \longrightarrow X}$

Construction 2: NMAC (nested MAC)

cascade



Let $F: K \times X \longrightarrow K$ be a PRF

Define new PRF $F_{NMAC} : K^2 \times X^{\leq L} \longrightarrow K$

Why the last encryption step in ECBC-MAC and NMAC?

NMAC: suppose we define a MAC $I = (S,V)$ where

$$S(k,m) = \text{cascade}(k, m)$$

- ○ This MAC is secure
- ○ This MAC can be forged without any chosen msg queries
- ○ This MAC can be forged with one chosen msg query
- ○ This MAC can be forged, but only with two msg queries

$\text{cascade}(k,m) \rightarrow \text{cascade}(k,m\|w)$ for any w

Why the last encryption step in ECBC-MAC?

Suppose we define a MAC $I_{RAW} = (S,V)$ where

$$S(k,m) = \text{rawCBC}(k,m)$$

Then I_{RAW} is easily broken using a 1-chosen msg attack.

Adversary works as follows:

- Choose an arbitrary one-block message $m \in X$
- Request tag for m. Get $t = F(k,m)$
- Output t as MAC forgery for the 2-block message $(m, t \oplus m)$

Indeed: $\text{rawCBC}\big(k, (m, t\oplus m)\big) = F\big(k, F(k,m)\oplus(t\oplus m)\big) = F\big(k, t\oplus(t\oplus m)\big) = t$

ECBC-MAC and NMAC analysis

Theorem: For any L>0,

For every eff. q-query PRF adv. A attacking F_{ECBC} or F_{NMAC}

there exists an eff. adversary B s.t.:

$$\mathrm{Adv}_{PRF}[A, F_{ECBC}] \leq \mathrm{Adv}_{PRP}[B, F] + 2\,q^2 / |X|$$

$$\mathrm{Adv}_{PRF}[A, F_{NMAC}] \leq q \cdot L \cdot \mathrm{Adv}_{PRF}[B, F] + q^2 / 2|K|$$

CBC-MAC is secure as long as $q \ll |X|^{1/2}$

NMAC is secure as long as $q \ll |K|^{1/2}$ (2^{64} for AES-128)

An example

$$\mathrm{Adv_{PRF}}[A, F_{ECBC}] \le \mathrm{Adv_{PRP}}[B, F] + \mathbf{2\, q^2 / |X|}$$

q = # messages MAC-ed with k

Suppose we want $\mathrm{Adv_{PRF}}[A, F_{ECBC}] \le 1/2^{32}$ $\Leftarrow$ $q^2 / |X| < 1/ 2^{32}$

- AES: $|X| = 2^{128}$ $\Rightarrow$ $q < 2^{48}$

  So, after 2^{48} messages must, must change key

- 3DES: $|X| = 2^{64}$ $\Rightarrow$ $q < 2^{16}$

Comparison

ECBC-MAC is commonly used as an AES-based MAC

- CCM encryption mode (used in 802.11i)
- NIST standard called CMAC

NMAC not usually used with AES or 3DES

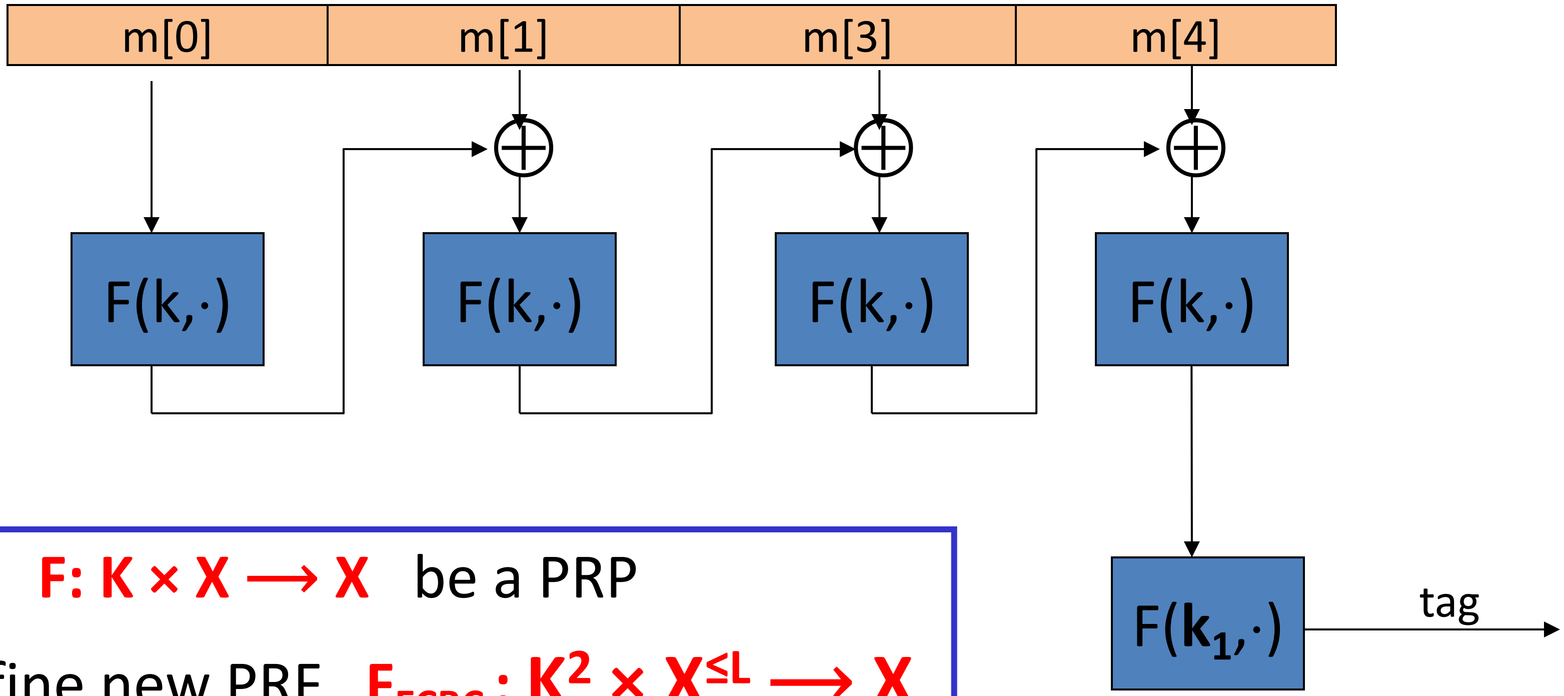
- Main reason: need to change AES key on every block
  requires re-computing AES key expansion
- But NMAC is the basis for a popular MAC called HMAC (next)

End of Segment

Message Integrity

MAC padding

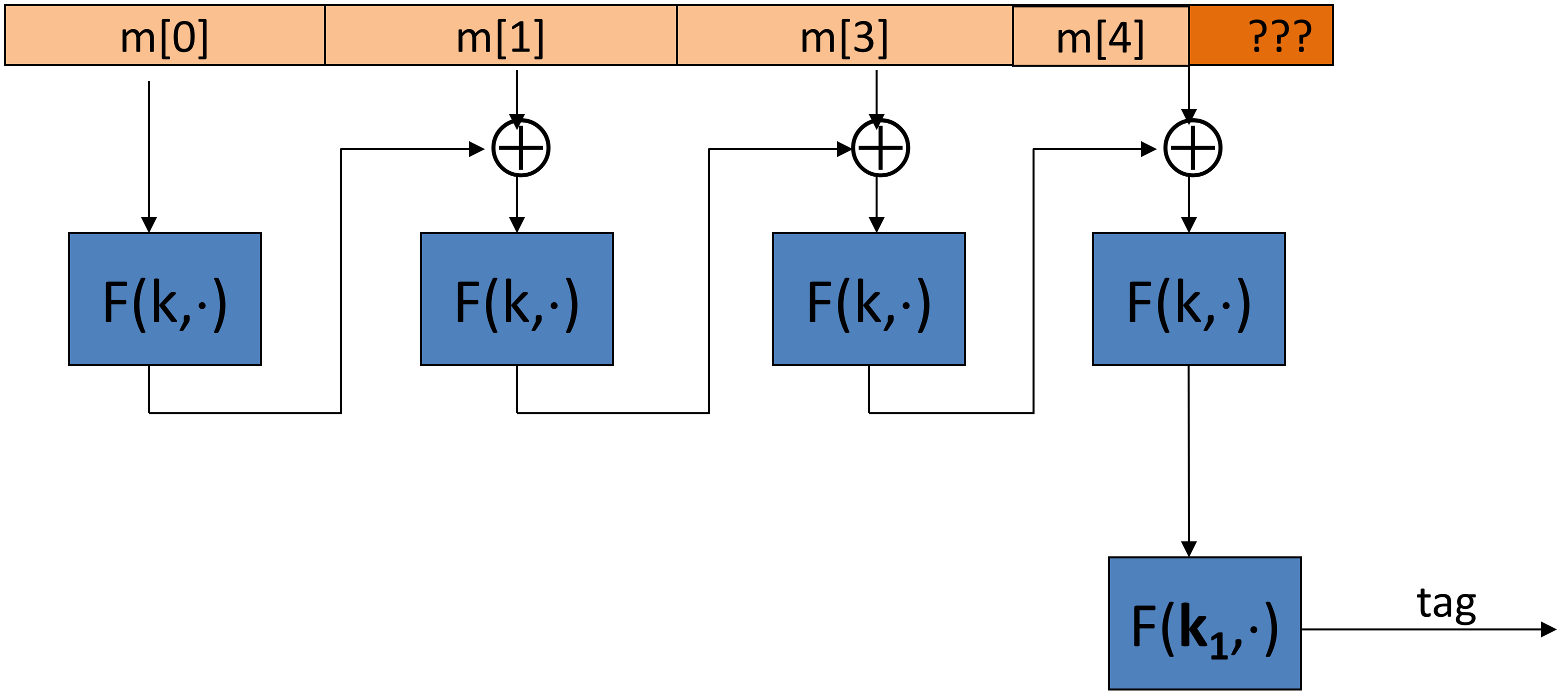
Recall: ECBC-MAC



Let $\mathbf{F: K \times X \longrightarrow X}$ be a PRP

Define new PRF $\mathbf{F_{ECBC} : K^2 \times X^{\leq L} \longrightarrow X}$

What if msg. len. is not multiple of block-size?



CBC MAC padding

Bad idea: pad m with 0's

Is the resulting MAC secure?

- ○ Yes, the MAC is secure
- ○ It depends on the underlying MAC
- √ ○ No, given tag on msg **m** attacker obtains tag on **m||0**
- ○

Problem: pad(m) = pad(m||0)

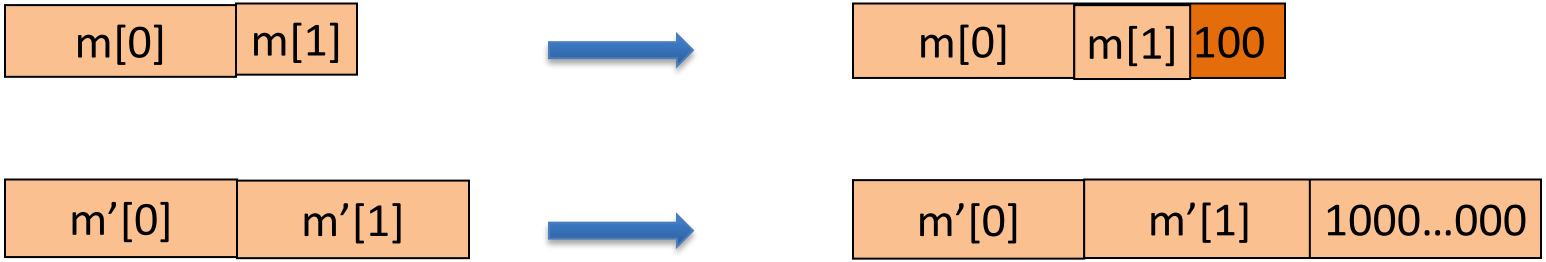
CBC MAC padding

For security, padding must be invertible !

$$m_0 \neq m_1 \quad \Rightarrow \quad pad(m_0) \neq pad(m_1)$$

<u>ISO</u>: pad with "1000…00". Add new dummy block if needed.

– The "1" indicates beginning of pad.

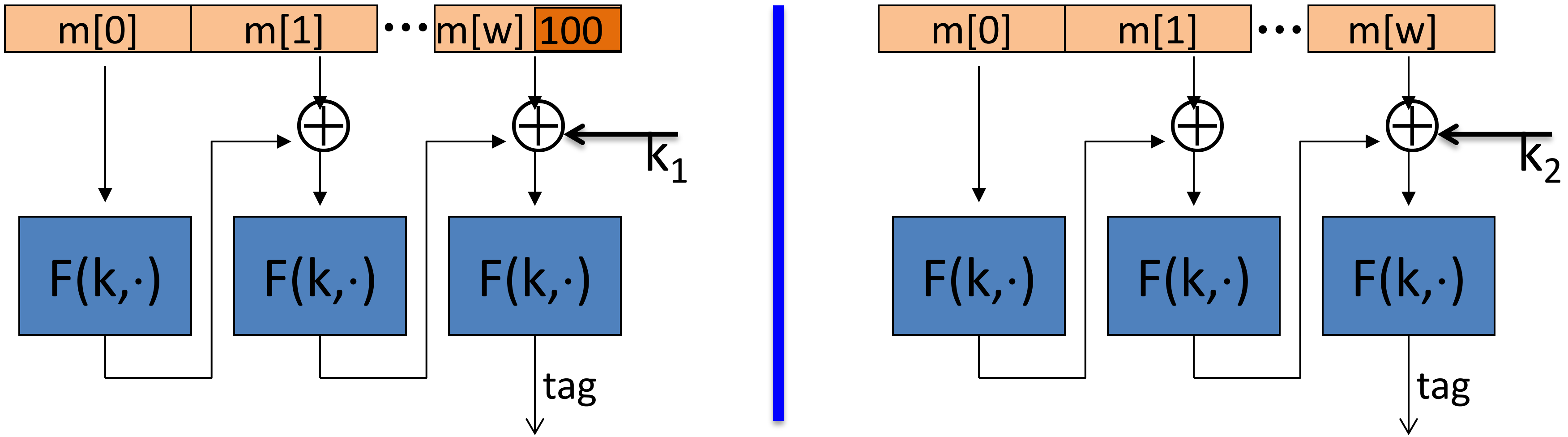


CMAC (NIST standard)

(k1,k2)devided from k

Variant of CBC-MAC where key = (k, k_1, k_2)

- No final encryption step (extension attack thwarted by last keyed xor)
- No dummy block (ambiguity resolved by use of k_1 or k_2)



End of Segment