CIS 5560

Cryptography
Lecture 9

Course website:

pratyushmishra.com/classes/cis-5560-s24/

Slides adapted from Dan Boneh and Vinod Vaikuntanathan

Announcements

- **HW 4 out after lecture**
  - Due **Tuesday**, Feb 20 at 1PM on Gradescope
  - Covers PRFs, IND-CPA

Recap of last lecture

Pseudorandom Functions

Collection of functions $\mathscr{F}_\ell = \{F_k : \{0,1\}^\ell \to \{0,1\}^m\}_{k \in \{0,1\}^n}$

- indexed by a key k
- n: key length, ℓ: input length, m: output length.
- Independent parameters, all poly(sec-param) = poly(n)
- #functions in $\mathscr{F}_\ell \leq 2^n$ (singly exponential in n)

$\mathbf{Gen}\left(1^n\right)$: Generate a random n-bit key k.

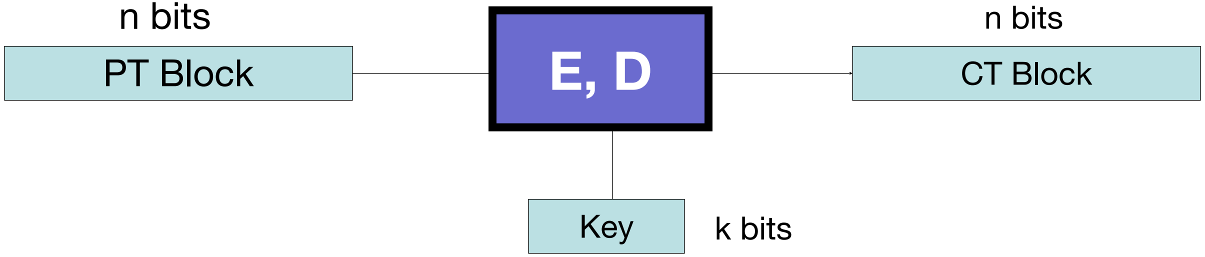
$\mathbf{Eval}(k, x)$ is a poly-time algorithm that outputs $F_k(x)$

Security: Cannot distinguish from random function

$$\left|\Pr\left[A^{f_k}(1^n) = 1 \mid k \leftarrow \{0,1\}^{\ell}\right] - \Pr\left[A^{F}(1^n) = 1 \mid F \leftarrow \mathsf{Fns}\right]\right| \leq \mathsf{negl}(n)\,.$$

PRP/Block Cipher

A **block cipher** is a pair of efficient algs. (E, D):

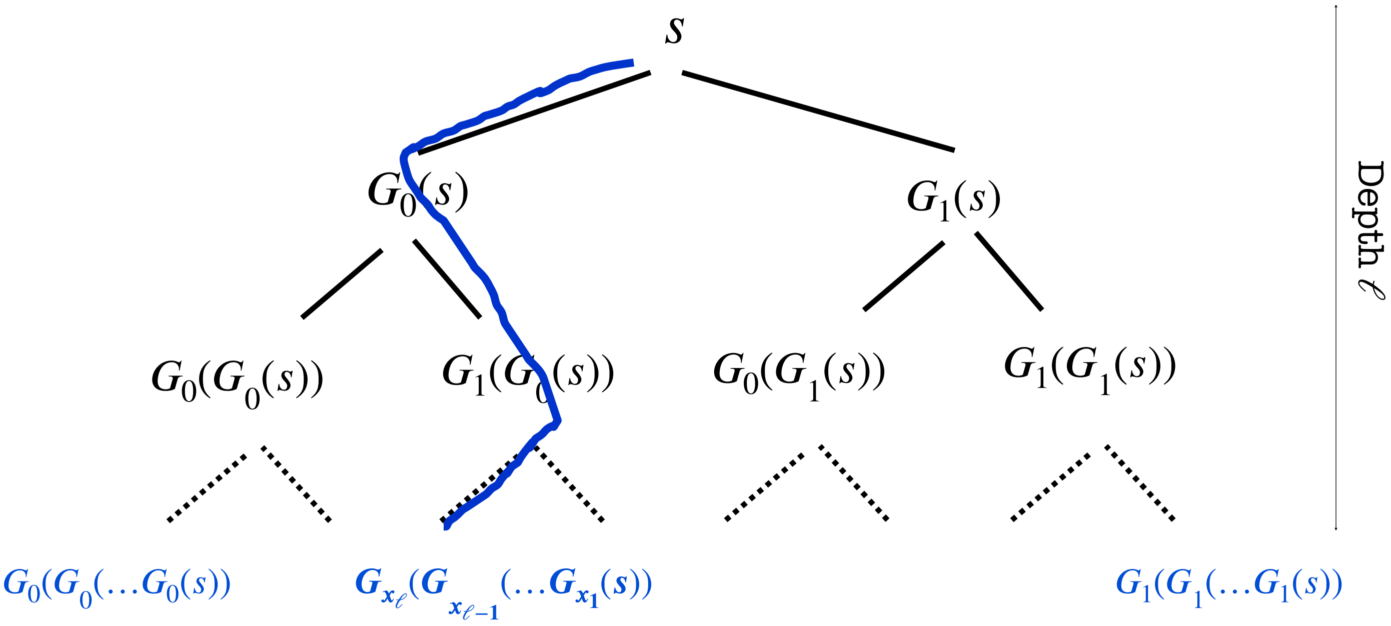


Canonical examples:

1. **AES**: n=128 bits, k = 128, 192, 256 bits
2. **3DES**: n= 64 bits, k = 168 bits (historical)

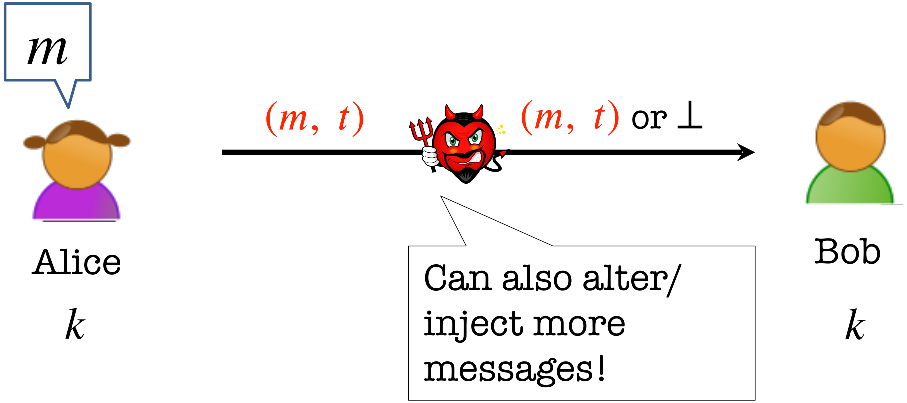
Goldreich-Goldwasser-Micali PRF

Construction: Let G(s) = $G_0(s) || G_1(s)$ where $G_0(s)$ and $G_1(s)$ are both n bits each.



Each path/leaf labeled by $x \in \{0,1\}^\ell$ corresponds to $f_s(x)$.

The authentication problem



We want Alice to generate a tag for the message *m* which is **hard to generate** without the secret key *k*.

Message Authentication Codes (MACs)

A triple of algorithms (Gen, MAC, Ver):

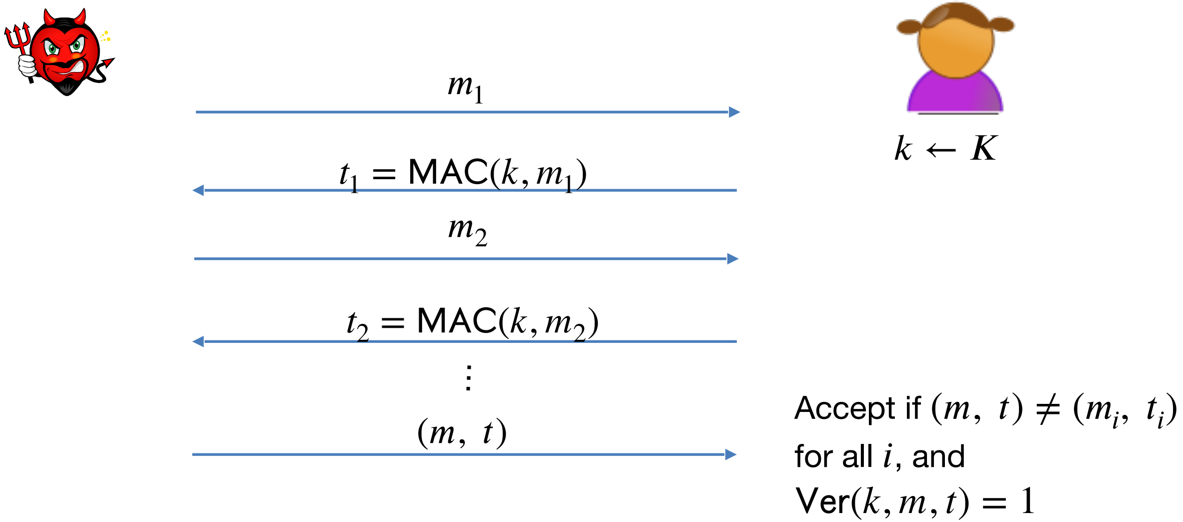
- Gen(1^n): Produces a key $k \leftarrow \mathscr{K}$.
- MAC$(k,\ m)$: Outputs a tag t (may be deterministic).
- Ver$(k,\ m,\ t)$: Outputs Accept or Reject.

Correctness: $\Pr[\text{Ver}(k, m, \text{MAC}(k, m)) = 1] = 1$

Security: *Hard to forge.* Intuitively, it should be hard to come up with a new pair *(m', t')* such that Ver accepts.

EUF-CMA Security

Existentially Unforgeable against Chosen Message Attacks

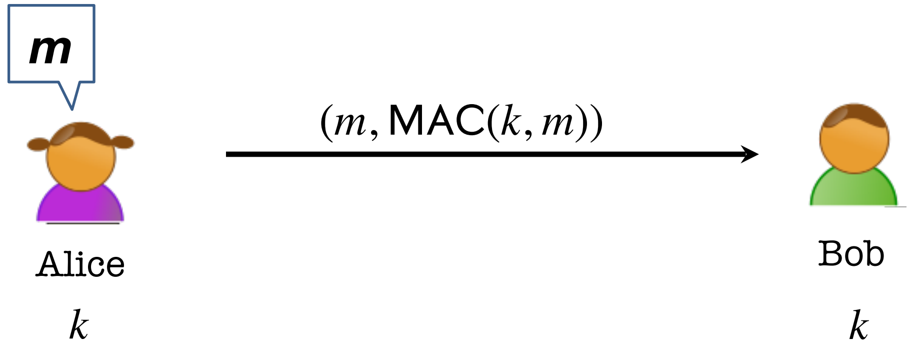


Accept if $(m,\ t) \neq (m_i,\ t_i)$
for all i, and
$\text{Ver}(k, m, t) = 1$

Want: $\Pr((m,\ t) \leftarrow A^{MAC(k,\ \bullet)}\left(1^n\right),\ Ver(k,\ m,\ t) = 1,\ (m,\ t) \notin Q)) = negl(n)$.

where Q is the set of queries $\left\{ \left(m_i,\ t_i\right) \right\}_i$ that A makes.

Constructing a MAC



Gen(1^n): Produces a PRF key $k \leftarrow K$.
MAC$(k,\ m)$: Output $f_k(m)$.
Ver$(k,\ m,\ t)$: Accept if $f_k(m) = t$, reject otherwise.

Security: Our earlier unpredictability lemma about PRFs essentially proves that this is secure!

Today's Lecture

- Proof of security for MAC
- Short MAC → Long MACs

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

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

$$\text{MAC}(k, m_0) = \text{MAC}(k, m_1) \quad \text{for } \tfrac{1}{2} \text{ of the keys k in K}$$

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

$\text{Adv}[A, I] = 1/2$

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

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

Can this MAC be secure?

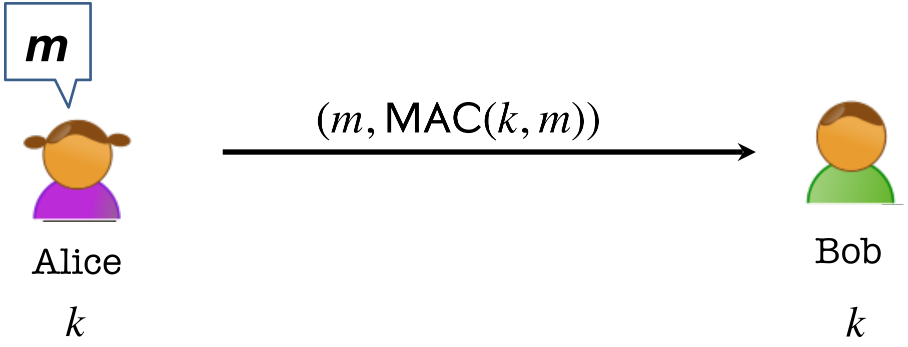
[✓ No, an attacker can simply guess the tag for messages]

It depends on the details of the MAC

Yes, the attacker cannot generate a valid tag for any message

$$\mathrm{Adv}[A, I] = 1/32$$

Constructing a MAC



Gen(1^n): Produces a PRF key $k \leftarrow K$.

MAC$(k,\ m)$: Output $F_k(m)$.

Ver$(k,\ m,\ t)$: Accept if $F_k(m) = t$, reject otherwise.

Security: ??

A bad example

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

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
- ○

$Adv[A, I_F] = 1/1024$

Security

Thm: 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.:

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

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

A Simple Lemma about Unpredictability

Let **F: K×X⟶Y** be a pseudorandom function.

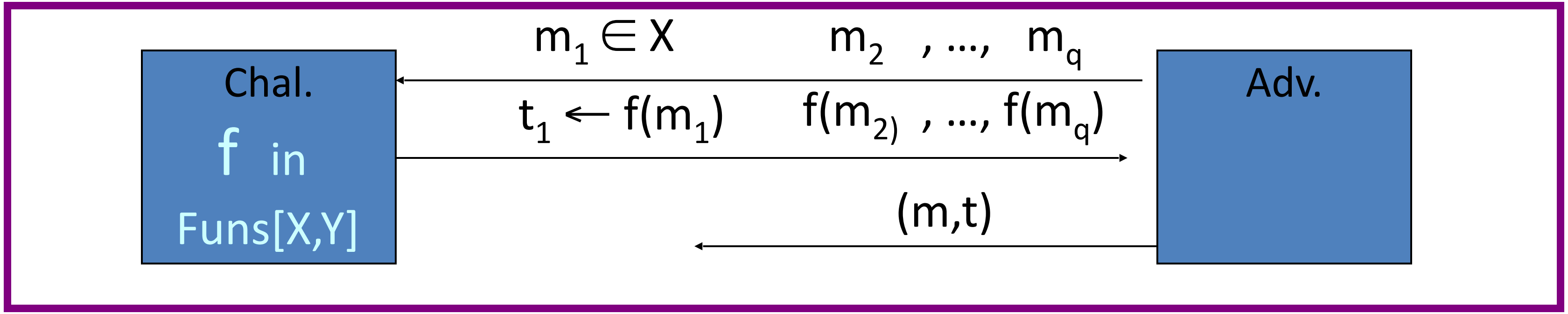
- Consider an adversary who requests and obtains $\mathsf{F}_k(x_1), \ldots, \mathsf{F}_k(x_q)$ for a polynomial $q = q(n)$.
- Can she predict $\mathsf{F}_k(x^\star)$ for some x^* of her choosing where $x^* \notin \{x_1,\ldots, x_q\}$? How well can she do it?

Lemma: If she succeeds with probability $\frac{1}{2^m} + 1/\mathrm{poly}(n)$, then she broke PRF security.

Proof Sketch

Suppose **$f: X \longrightarrow Y$** is a truly random function

Then MAC adversary A must win the following game:



Chal.

f in Funs[X,Y]

$m_1 \in X$ m_2 , …, m_q

$t_1 \leftarrow f(m_1)$ $f(m_{2)}$, …, $f(m_q)$

(m,t)

Adv.

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

$\Rightarrow$ Pr[A wins] = 1/|Y|

By PRF security,
same must hold for F(k,x)

Dealing with Replay Attacks

- The adversary could send an old valid *(m, tag)* at a later time.
  - In fact, our definition of security does not rule this out.

- **In practice:**
  - Append a time-stamp to the message. Eg. (m, T, MAC(m, T)) where T = 21 Sep 2022, 1:47pm.
  - Sequence numbers appended to the message (this requires the MAC algorithm to be *stateful*).

MACs and PRFs

So far: secure PRF **F** ⇒ secure MAC, as long as $|Y|$ is large

$$\text{MAC}(k, m) = F(k, m)$$

Our goal:

given a PRF for short messages (AES)

construct a PRF for long messages

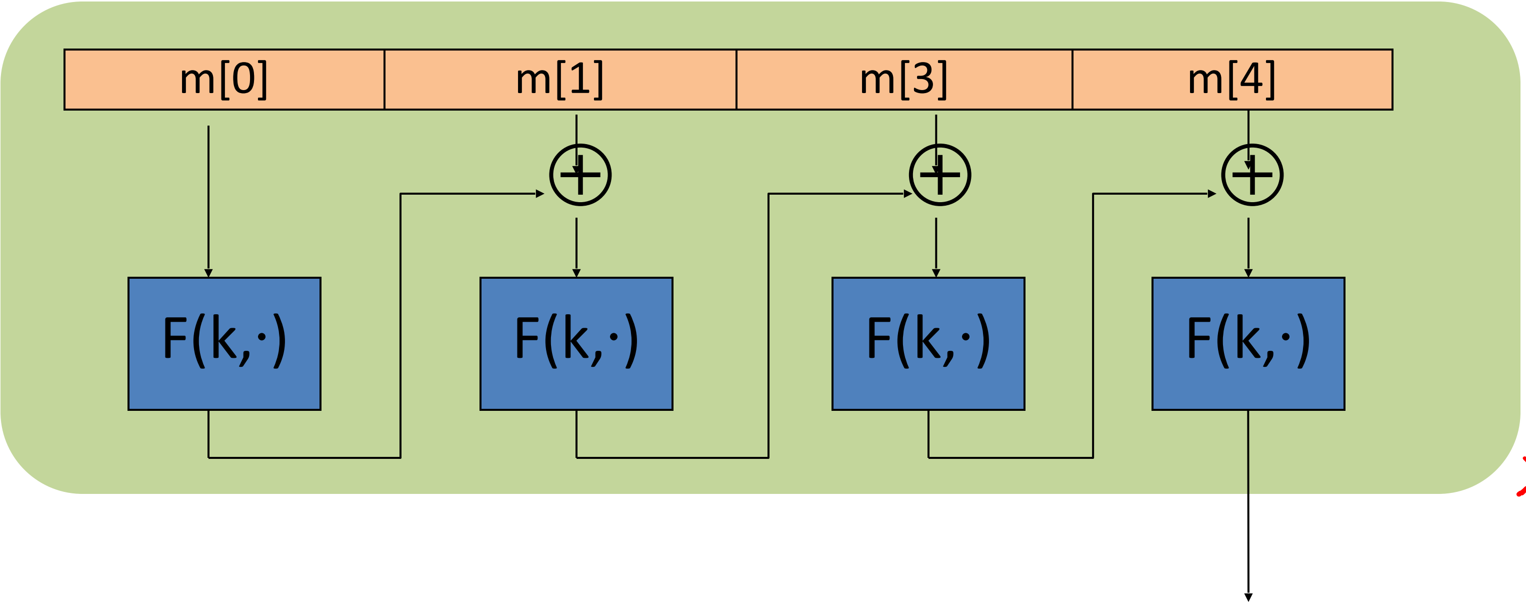
From here on let $X = \{0,1\}^n$ (e.g. $n=128$)

Ideas?

On board: randomized construction

Construction Attempt: just CBC-MAC

raw CBC



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

Why is this broken?

rawCBC is easily broken using a 1-chosen msg attack.

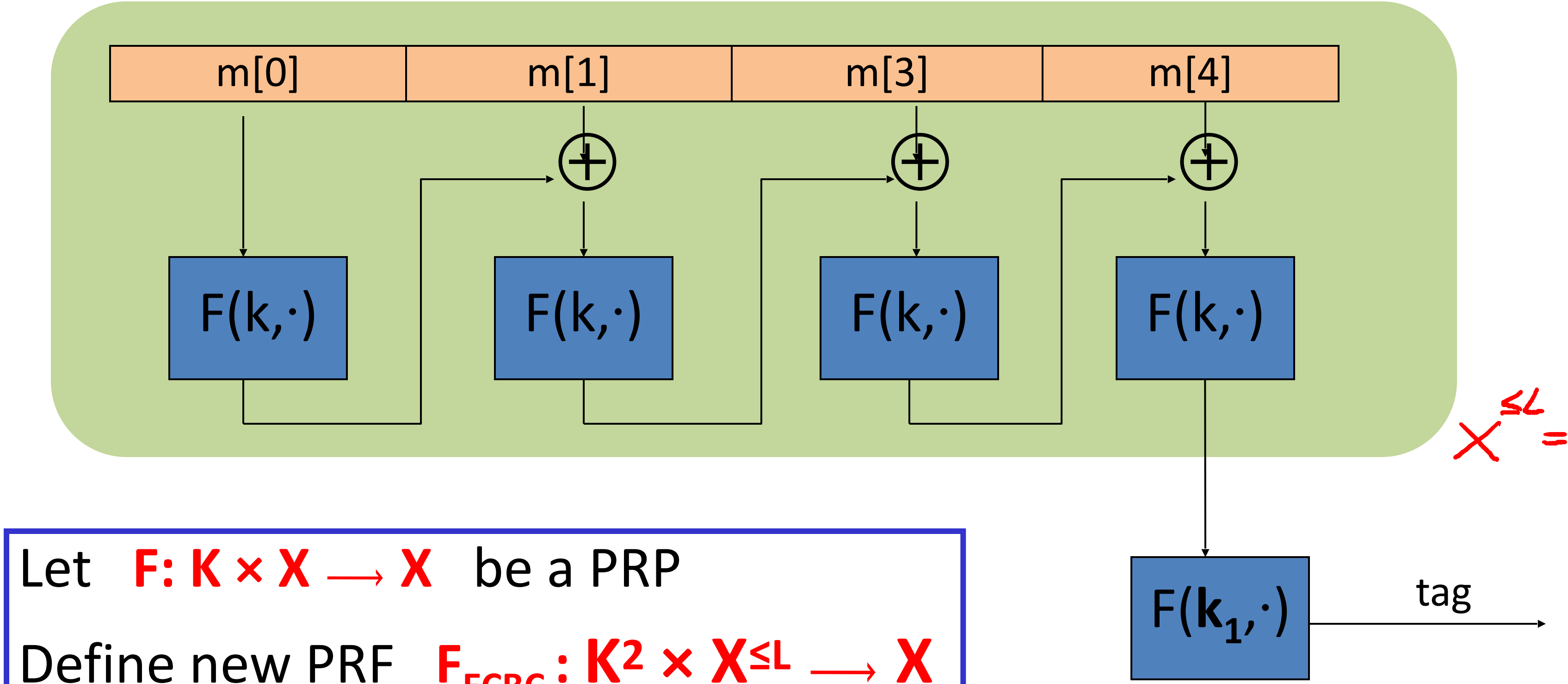
Adversary works as follows:

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

Construction: encrypted CBC-MAC

raw CBC



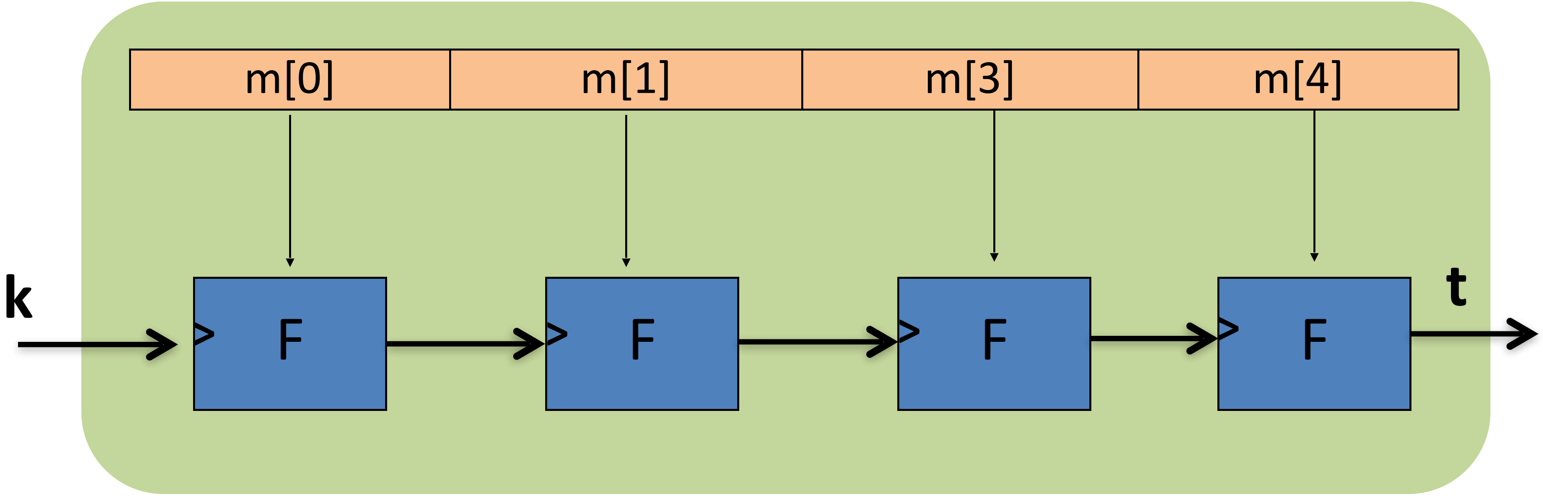
$X^{\le L} = \bigcup_{i=1}^{L} X^i$

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

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

Construction Attempt: Just Cascade

cascade



Does this work?

This MAC is secure

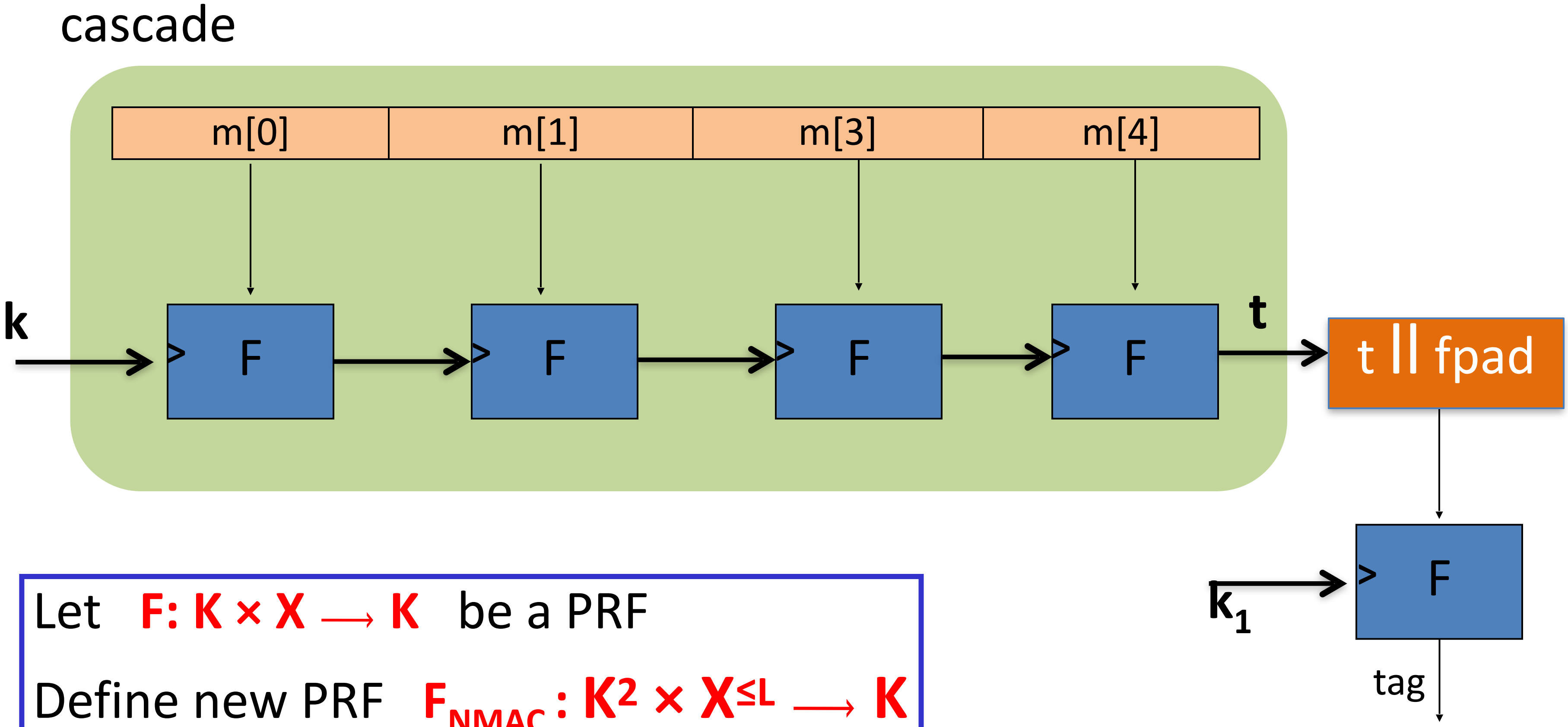
This MAC can be forged without any chosen msg queries

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

Construction: NMAC (nested MAC)



Let **F: K × X ⟶ K** be a PRF

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

Dan Boneh

Comparison

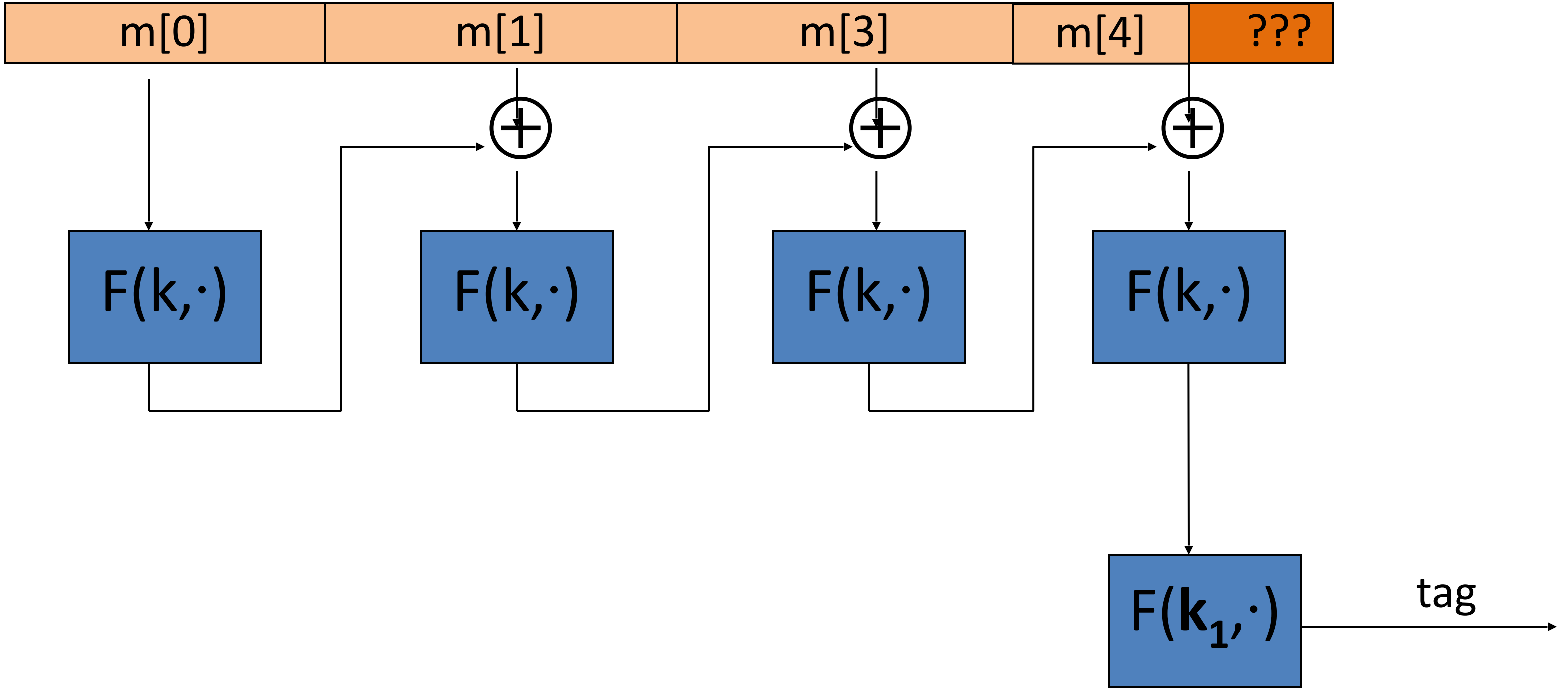
ECBC-MAC is commonly used as an AES-based MAC

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

NMAC not usually used with AES or 3DES

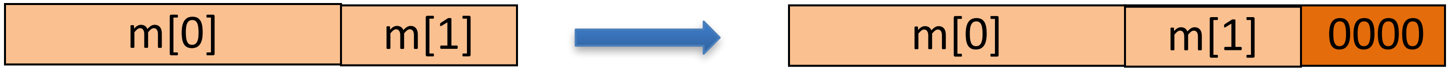
- Main reason: need to change AES key on every block
  requires re-computing AES key expansion
- But NMAC is the basis for a popular MAC called HMAC (next)

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)

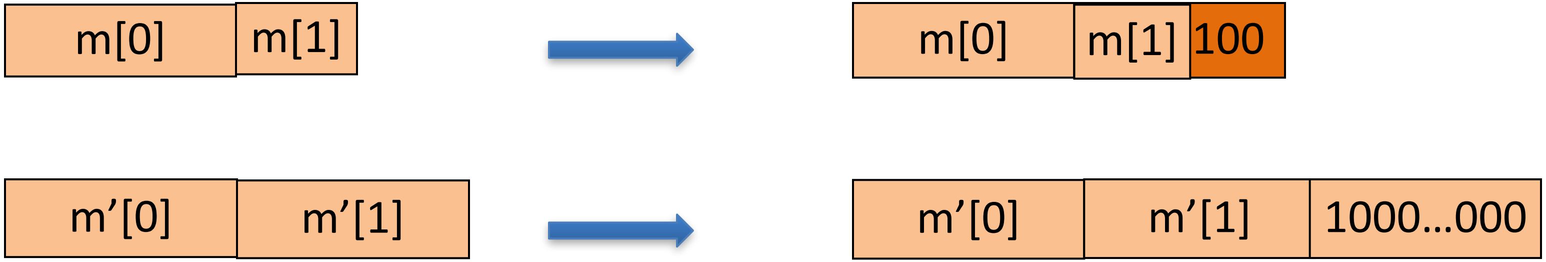
CBC MAC padding

For security, padding must be invertible !

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

ISO: pad with "1000…00". Add new dummy block if needed.

- The "1" indicates beginning of pad.

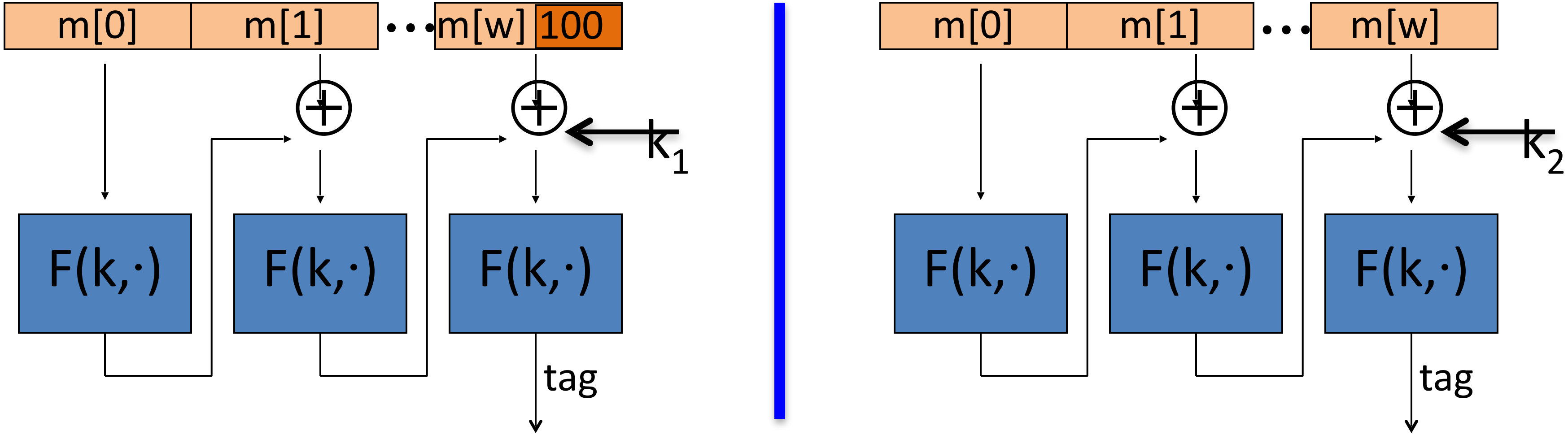


CMAC (NIST standard)

(k_1, k_2) derived 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