CIS 5560

Cryptography
Lecture 10

Announcements

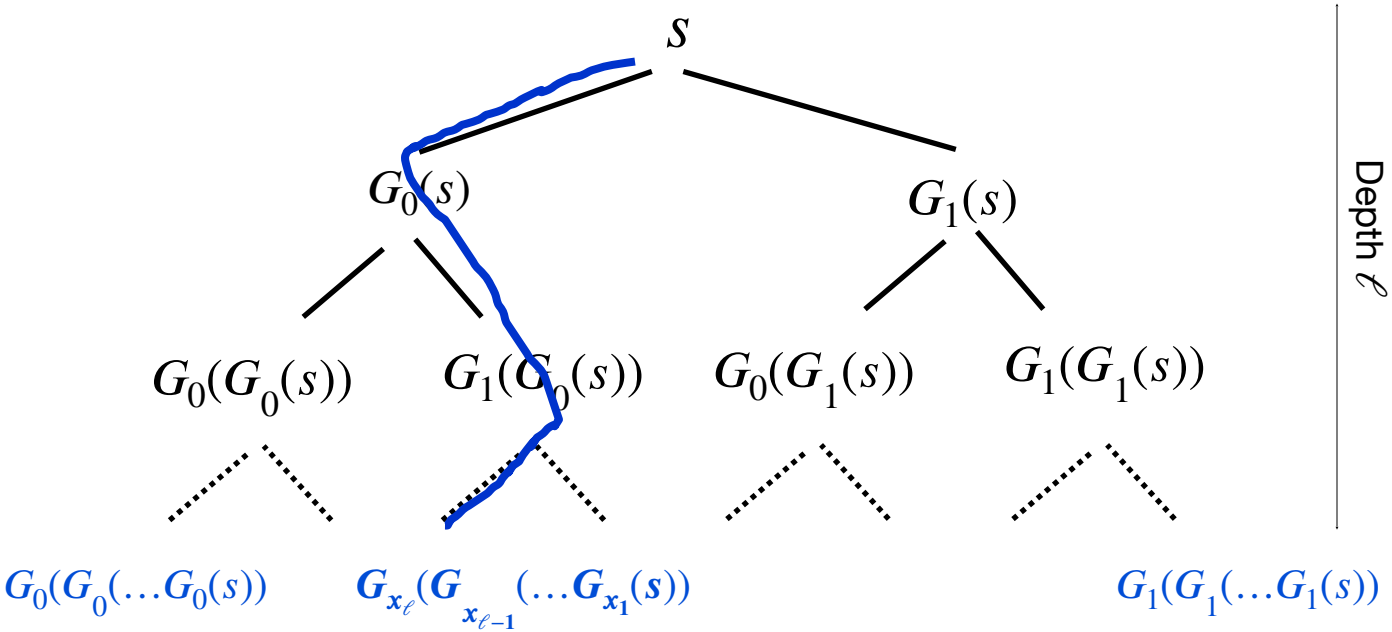
- **HW 3 due on Friday 2/20** via HW-writing sessions
- **HW 4 out on Wednesday 2/18**
  - Due **Friday**, 2/27
  - Covers MACs, and CRHFs
- If you can't make it to a HW writing session, let me know by email (See Ed post)

Recap of last lecture

- Proof of security for GGM construction
- New Security Goal: message authentication
- New primitive: Message Authentication Codes

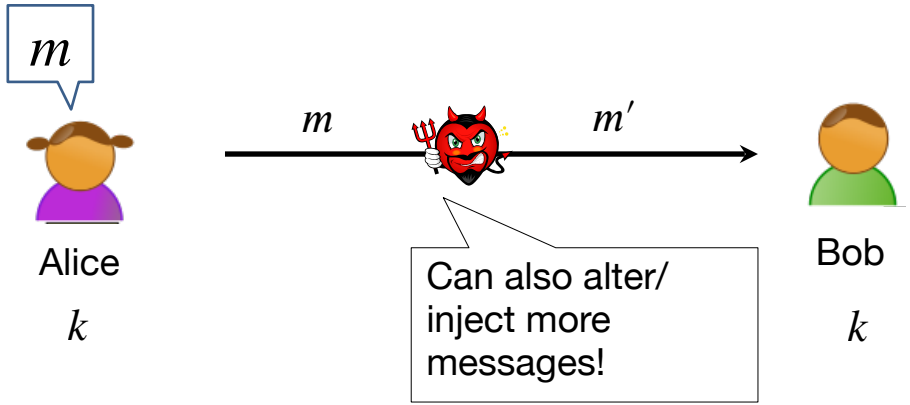
Goldreich-Goldwasser-Micali PRF

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



This is known as a **man-in-the-middle attack.**

How can Bob check if the message is indeed from Alice?

Today’s Lecture

- MACs
- Birthday bound
- CRH → MACs
  - HMAC

Message Authentication Codes (MACs)

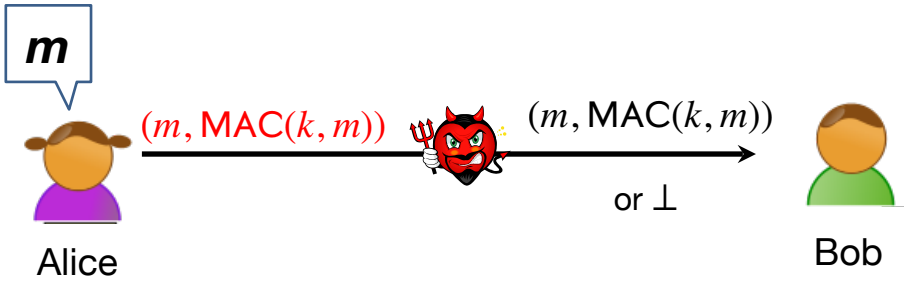
A triple of algorithms (Gen, MAC, Ver):

- Gen(1^n): Produces a key $k \leftarrow \mathcal{K}$.
- 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.

What is the power of the adversary?



- Can see many pairs $(m, \text{MAC}(k, m))$
- Modeled as a MAC oracle $\text{MAC}(k, \cdot)$
  - Obtain tags for message of choice.

This is called a *chosen message attack (CMA).*

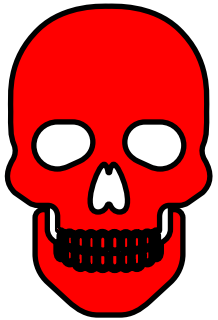
Defining MAC Security

- **Total break:** The adversary should not be able to recover the key k.
- **Universal break:** The adversary can generate a valid tag for every message.
- **Existential break:** The adversary can generate a new valid tag t for some message m.

We will require MACs to be secure against the existential break!!

EUF-CMA Security

Existentially **U**n**f**orgeable against **C**hosen **M**essage **A**ttacks



m_1

$t_1 = \mathsf{MAC}(k, m_1)$

m_2

$t_2 = \mathsf{MAC}(k, m_2)$

$\vdots$

$(m,\ t)$

$k \leftarrow K$

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

Want: $\Pr\left[\begin{array}{r|l} \mathsf{Ver}(k, m, t) = 1 & (m, t) \leftarrow A^{\mathsf{MAC}(k,\cdot)}() \\ (m, t) \notin Q & \end{array}\right] = \mathsf{negl}(n)$

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

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

$$\mathsf{MAC}(k, m_0) = \mathsf{MAC}(k, m_1) \text{ for } ½ \text{ of the keys}$$

Can this MAC be secure?

(1) Yes, the attacker cannot generate a valid tag for m_0 or m_1

(2) No, this MAC can be broken using a chosen msg attack

(3) It depends on the details of the MAC

Suppose $\mathsf{MAC}(k, m)$ is always 5 bits long

Can this MAC be secure?

(1) No, an attacker can simply guess the tag for messages

(2) It depends on the details of the MAC

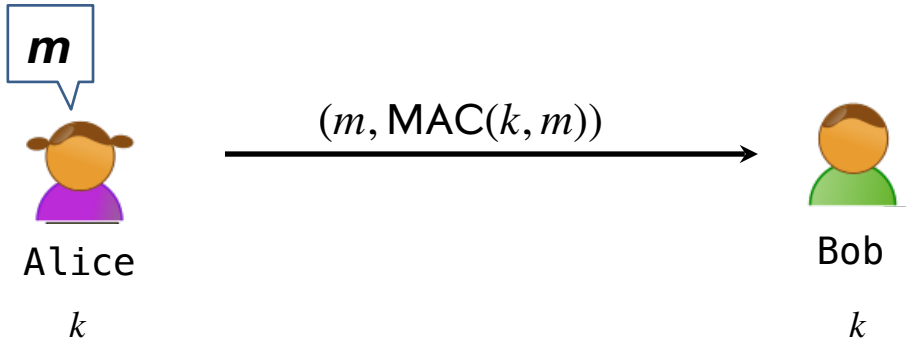
(3) Yes, the attacker cannot generate a valid tag for any message

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*).

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 $F : K \times X \rightarrow \{0,1\}^{10}$ is a secure PRF.

Does plugging F into the previous construction give a secure MAC?

- ○ 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

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

A Simple Lemma about Unpredictability

Let $f : X \to Y$ be a random function.

Consider an adversary who requests and obtains $f(x_1), \ldots, f(x_q)$ for a polynomial $q = q(n)$.

Can she predict $f(x^\star)$ for some x^* of her choosing where $x^* \notin \{x_1, \ldots, x_q\}$? How well can she do it?

She succeeds with probability $1/|Y|$.

Since oracle access to RF is indistinguishable from oracle access to PRF, she guesses output of PRF w/ prob $1/|Y| + \text{negl}(|K|)$

Security

Thm: If $F : K \times X \to Y$ is a secure PRF and $1/|Y|$ is negligible (i.e. $|Y|$ is large) then the previous scheme is a secure MAC.

In particular, for every PPT MAC adversary A,

there exists a PPT PRF adversary B attacking F s.t.:

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

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

MACs and PRFs

So far: secure PRF $F \Rightarrow$ secure MAC, as long as $|Y|$ is large

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

Our goal:

given a PRF for short messages (e.g., AES)

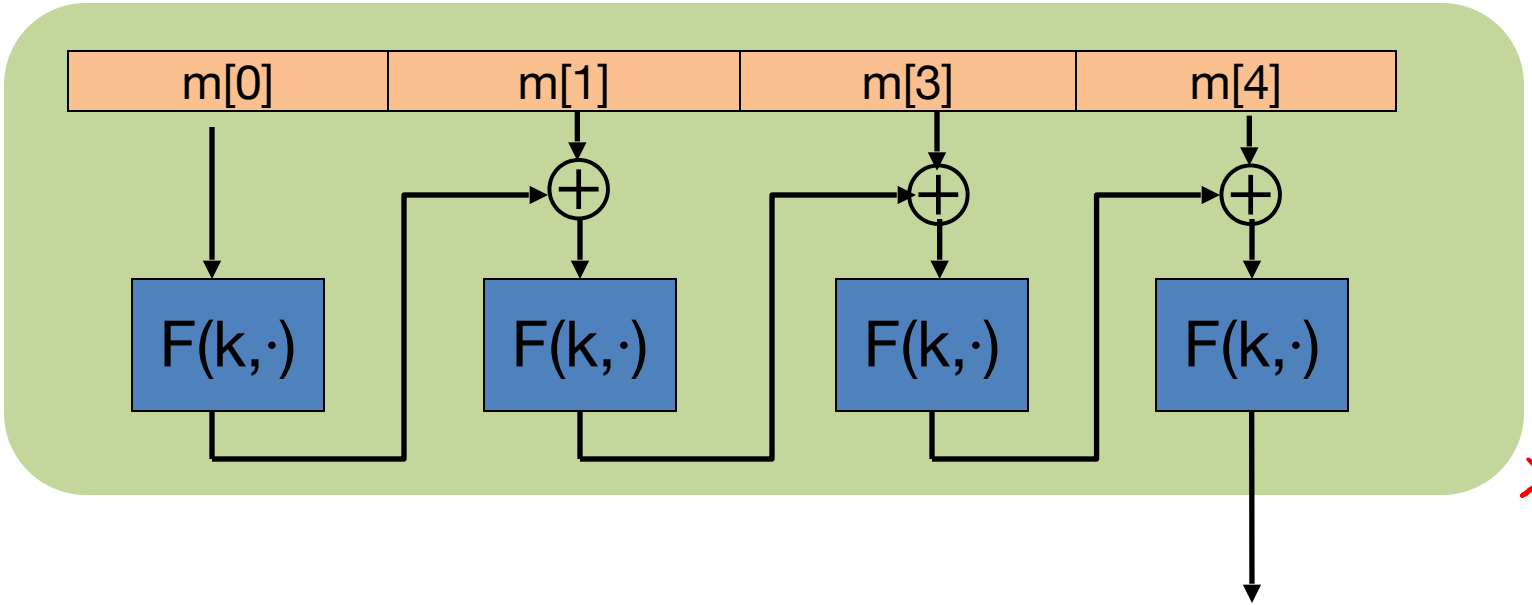
construct a PRF for long messages

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

Ideas?

Construction Attempt: just CBC-MAC

raw CBC



$$x^{\leq \ell} = \bigcup_{i=1}^{\ell} x^i$$

Why is this broken?

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

Adversary works as follows:

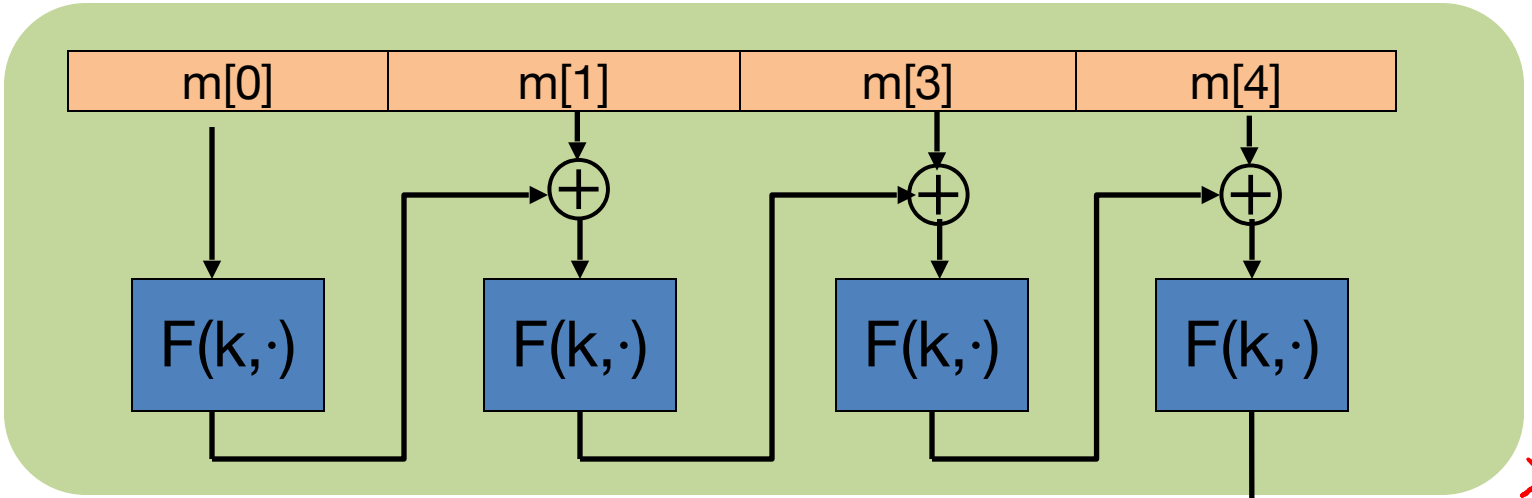
- Choose an arbitrary one-block message $m \in X$
- Request tag for m. Get $t = F(k, m)$
- Output t as MAC forgery for the 2-block message $(m, t \oplus m)$

Indeed:

$$\begin{aligned} \text{rawCBC}(k, (m, t \oplus m)) &= F(k, F(k, m) \oplus (t \oplus m)) \\ &= F(k, t \oplus (t \oplus m)) \\ &= t \end{aligned}$$

Construction Attempt: "encrypted" CBC-MAC

raw CBC

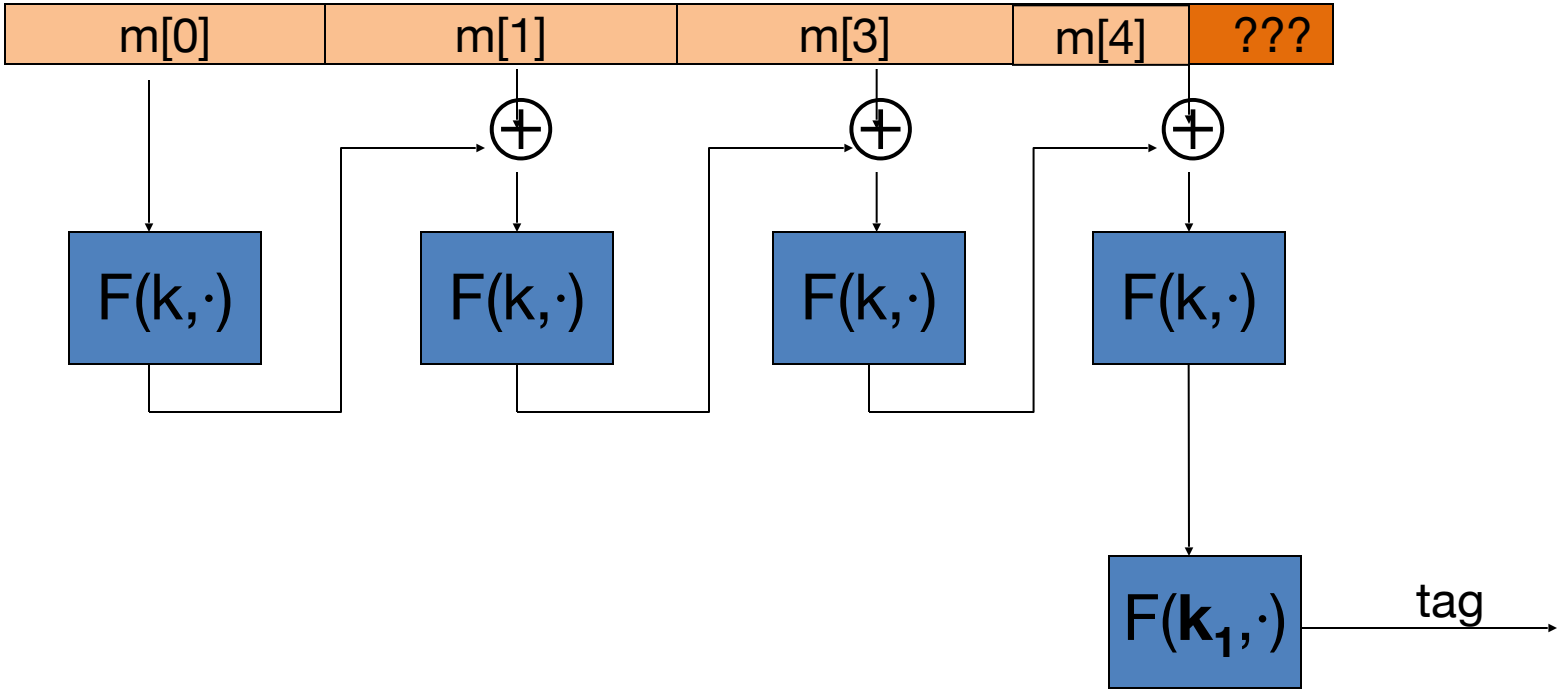


$X^{\leq L} = \bigcup_{i=1}^{L} X^i$

Let $F : K \times X \rightarrow X$ be a PRP

Define new PRF $F_{\text{ECBC}} : K^2 \times X_{\leq L} \rightarrow X$

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)

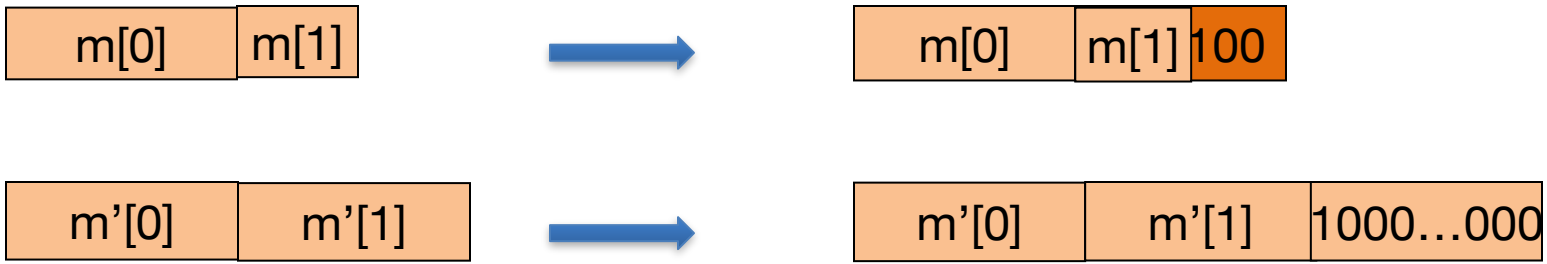
CBC MAC padding

For security, padding must be invertible !

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

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)