COS433/Math 473: Cryptography

Mark Zhandry

Princeton University

Fall 2020

Announcements/Reminders

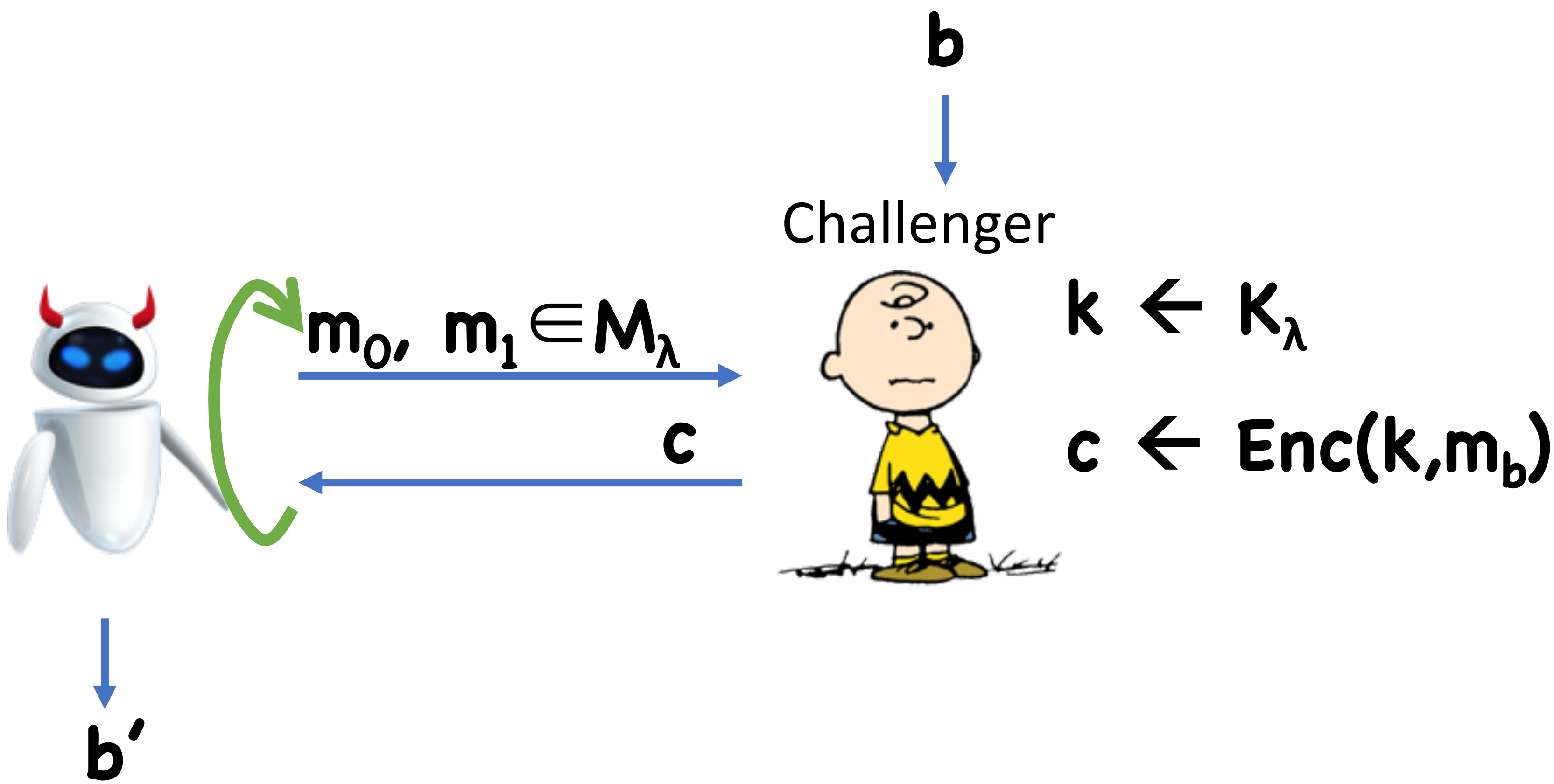
PR1 Due TODAY

HW3 due on Oct 20

Previously on COS 433…

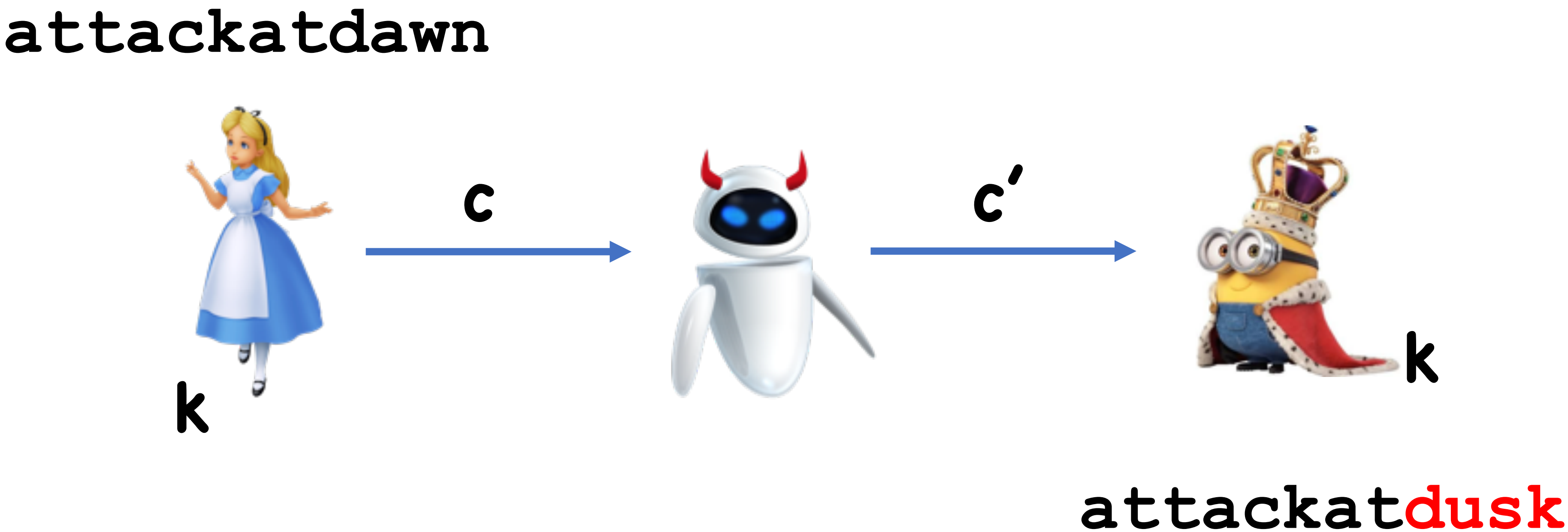
Message Integrity

Recall: CPA Security



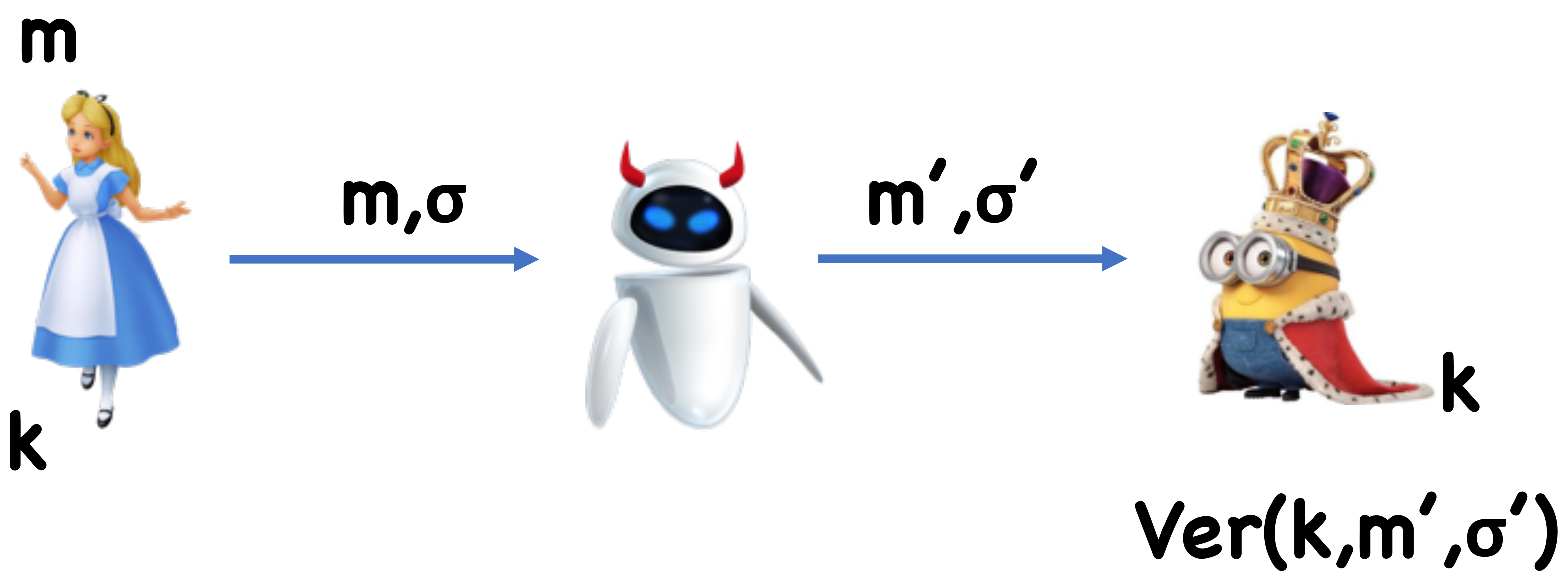
$\text{LoR-Exp}_b($  $, \lambda)$

Limitations of CPA security



How?

Message Authentication



Goal: If Eve changed **m**, Bob should reject

Message Authentication Codes

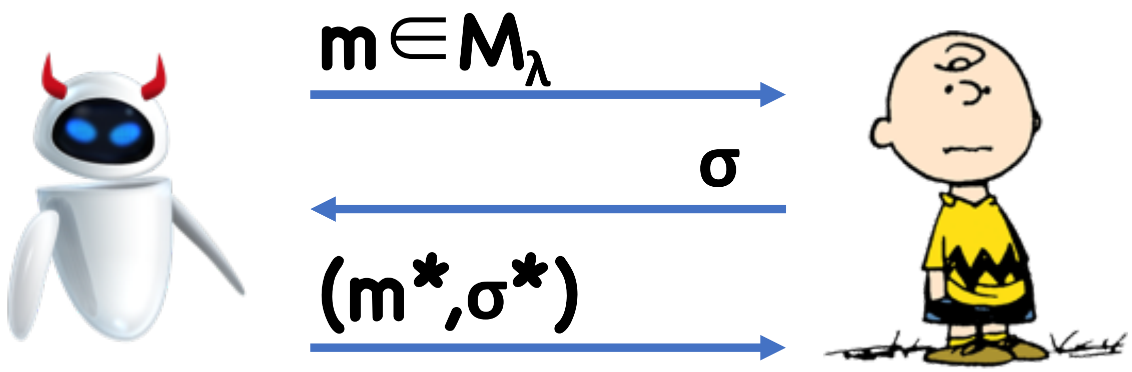
Syntax:

- Key space $\mathbf{K_\lambda}$
- Message space $\mathbf{M_\lambda}$
- Tag space $\mathbf{T_\lambda}$
- **MAC(k,m) → σ**
- **Ver(k,m,σ) → 0/1**

Correctness:

- $\forall$**m,k, Ver(k,m, MAC(k,m)) = 1**

1-time Security For MACs



$k \leftarrow K_\lambda$

$\sigma \leftarrow MAC(k,m)$

Output 1 iff:

- $m^* \neq m$
- $Ver(k, m^*, \sigma^*) = 1$

1CMA-Adv( , λ) = Pr[ outputs 1]

Definition: (MAC,Ver) is 1-time statistically secure under a chosen message attack (**statistically 1CMA-secure)** if, for all , ∃ negligible **ε** such that:

1CMA-Adv(, λ) ≤ ε(λ)

Today

Message Integrity, continued
Authenticated encryption

Question

Is perfect security (**ε=0**) possible?

A Simple 1-time MAC

Suppose $\mathbf{H}_\lambda$ is a family of pairwise independent functions from $\mathbf{M}_\lambda$ to $\mathbf{T}_\lambda$

For any $\mathbf{m_0 \neq m_1} \in \mathbf{M}_\lambda$, $\sigma_0, \sigma_1 \in \mathbf{T}_\lambda$

$$\Pr_{h \leftarrow H_\lambda}[\ h(m_0)=\sigma_0 \wedge h(m_1)=\sigma_1] = 1/|T_\lambda|^2$$

$\mathbf{K = H_\lambda}$

$\mathbf{MAC(h, m) = h(m)}$

$\mathbf{Ver(h,m,\sigma) = (h(m) == \sigma)}$

Theorem: If $|T_\lambda|$ is super-polynomial, then **(MAC,Ver)** is 1-time secure

Intuition: after seeing one message/tag pair, adversary learns nothing about tag on any other message

So to have security, just need $|T_\lambda|$ to be large

Ex: $T_\lambda = \{0,1\}^{128}$

Constructing Pairwise Independent Functions

$\mathbf{T_\lambda} = \mathbb{F}$ (finite field of size $\approx 2^\lambda$)

- Example: $\mathbb{Z}_p$ **for some prime p**

Easy case: let $\mathbf{M_\lambda} = \mathbb{F}$

- $\mathbf{H_\lambda = \{h(x) = a\,x + b:\ a,b \in \mathbb{F}\}}$

Slightly harder case: Embed $\mathbf{M_\lambda} \subseteq \mathbb{F}^n$

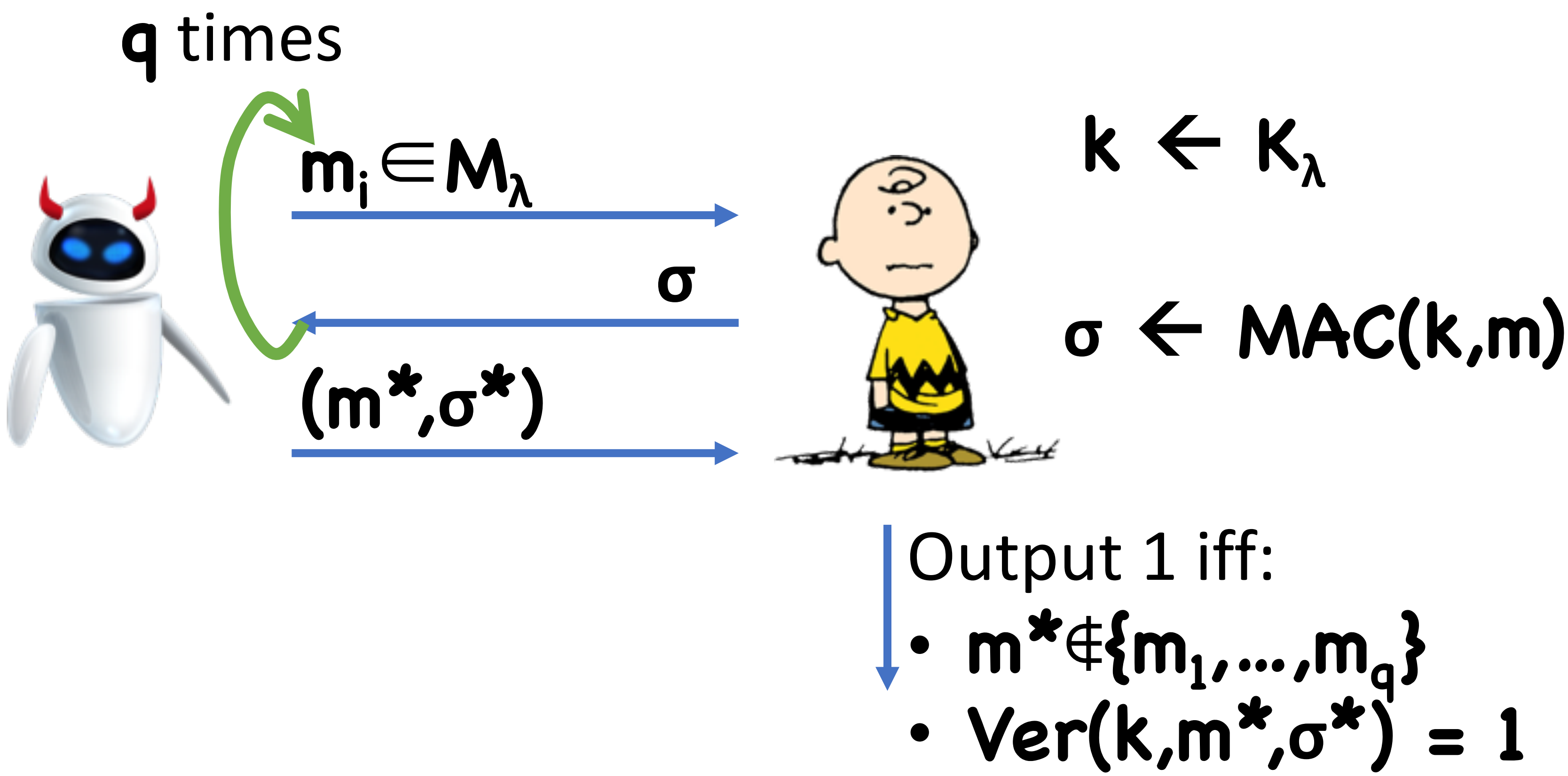
- $\mathbf{H_\lambda = \{h(x) = \langle a,x\rangle + b:\ a \in \mathbb{F}^n,\ b \in \mathbb{F}\}}$

Multiple Use MACs?

Just like with OTP, if use 1-time MAC twice, security no longer guaranteed

Why?

q-Time MACs



qCMA-Adv(, λ) = Pr[ outputs 1]

Definition: (MAC,Ver) is **q**-time statistically secure under a chosen message attack (**statistically qCMA-secure)** if, for all  making at most **q** queries, ∃ negligible **ε** such that:

$$\textbf{CMA-Adv}(\text{<adversary>}, \lambda) \leq \varepsilon(\lambda)$$

CMA-Adv(**, λ) ≤ ε(λ)**

Constructing **q**-time MACs

Ideas?

Limitations?

Impossibility of Large **q**

Theorem: Any **qCMA-secure** MAC must have

$$q \leq \log |K_\lambda|$$

Proof

Idea:

- By making $\mathbf{q \gg \log |K_\lambda|}$ queries, you *should* be able to uniquely determine key
- Once key is determined, can forge any message

Problem:

- What if certain bits of the key are ignored
- Intuition: ignoring bits of key shouldn't help

Proof

Define $\mathbf{r_q}$ as follows:

- Challenger chooses random key $\mathbf{k}$
- Adversary repeatedly choose random (distinct) messages $\mathbf{m_i}$ in $\mathbf{M_\lambda}$
- Query the CMA challenger on each $\mathbf{m_i}$, obtaining $\boldsymbol{\sigma_i}$
- Let $\mathbf{K'_q}$ be set of keys $\mathbf{k'}$ such that $\mathbf{MAC(k',m_i)=\sigma_i}$ for $\mathbf{i=1,\ldots,q}$
- Let $\mathbf{r_q}$ be the expected size of $\mathbf{K'_q}$

Claim: If **(MAC,Ver)** is qCMA-secure, then
$r_q \leq r_{q-1}/2$

If not, then with probability at least ¼,

$|K'_q| > |K'_{q-1}|/4$

Attack:

- Make **q-1** queries on random messages m_i
- Choose key **k** from K'_{q-1}
- Choose random m_q, compute $\sigma_q = MAC(k, m_q)$
- Output (m_q, σ_q)

Probability of forgery?

Claim: If **(MAC,Ver)** is qCMA-secure, then

$r_q \leq r_{q-1}/2$

Finishing the impossibility proof:

- r_q is always at least 1 (since there is a consistent key)
- $r_0 = |K_\lambda|$
- $1 \leq r_q \leq r_0/2^q \leq |K_\lambda|/2^q$
- Setting $q > \log |K_\lambda|$ gives a contradiction

Computational Security

Definition: (MAC,Ver) is computationally secure under a chosen message attack (**CMA-secure)** if, for all  running in polynomial time (and making a polynomial number of queries), $\exists$ negligible ε such that

$$\textbf{CMA-Adv}(\text{[adversary]}, \lambda) \leq \varepsilon(\lambda)$$

CMA-Adv(**, λ) ≤ ε(λ)**

Constructing MACs

Use a PRF

$F: K_\lambda \times M_\lambda \rightarrow T_\lambda$

$MAC(k,m) = F(k,m)$
$Ver(k,m,\sigma) = (F(k,m) == \sigma)$

Theorem: If $\mathbf{F}$ is a secure PRF and $|\mathbf{T}_\lambda|$ is super-polynomial, then **(MAC,Ver)** is CMA secure

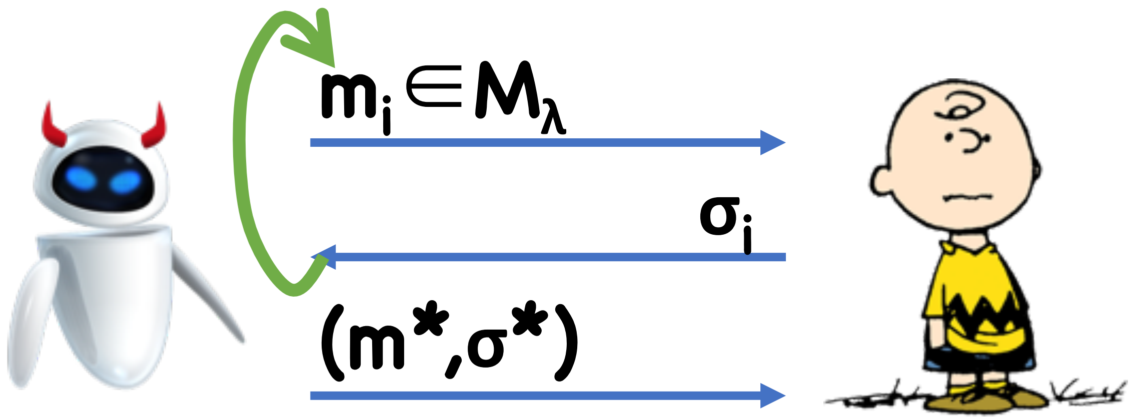
Security Proof

Assume toward contradiction polynomial time 

Hybrids!

Security Proof

Hybrid 0



$k \leftarrow K_\lambda$

$\sigma \leftarrow F(k, m_i)$

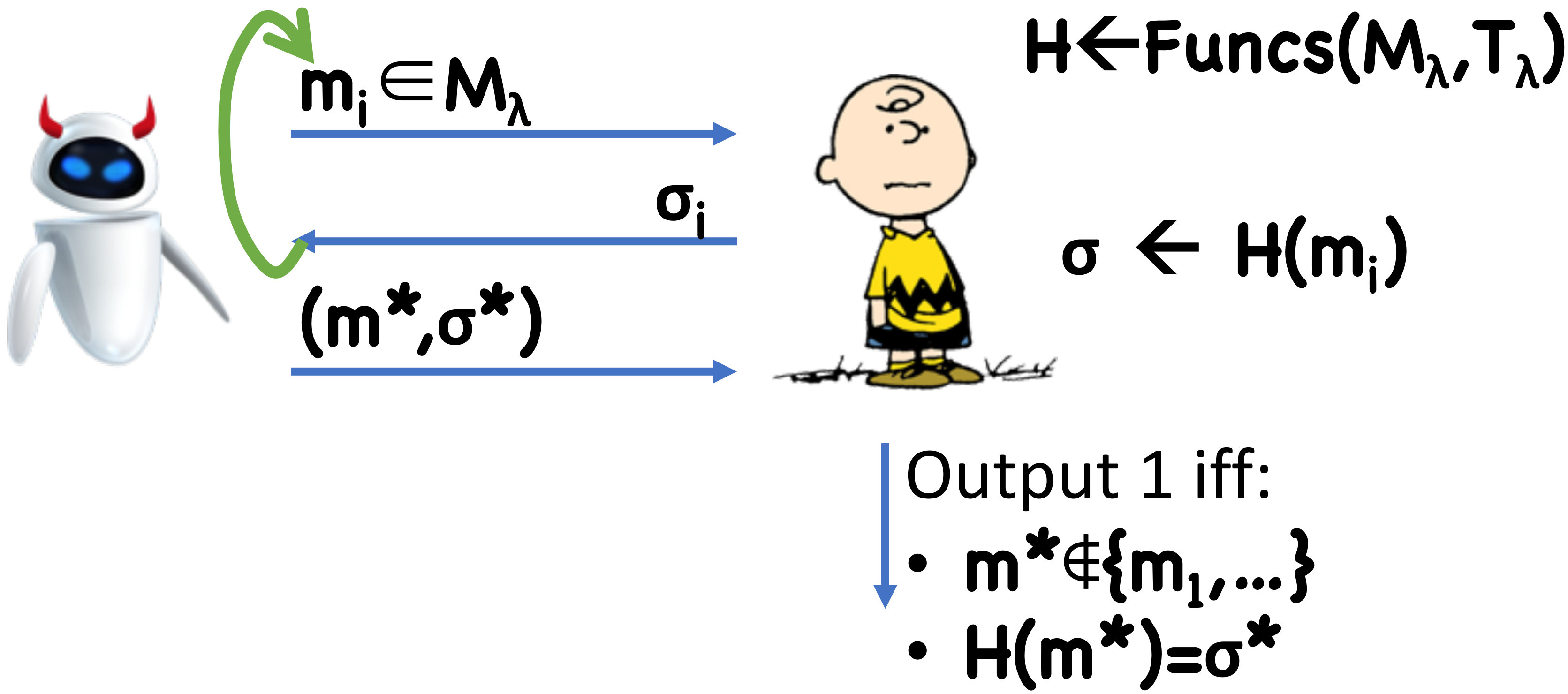
Output 1 iff:

- $m^* \notin \{m_1, \ldots\}$
- $F(k, m^*) = \sigma^*$

CMA Experiment

Security Proof

<u>Hybrid 1</u>



Security Proof

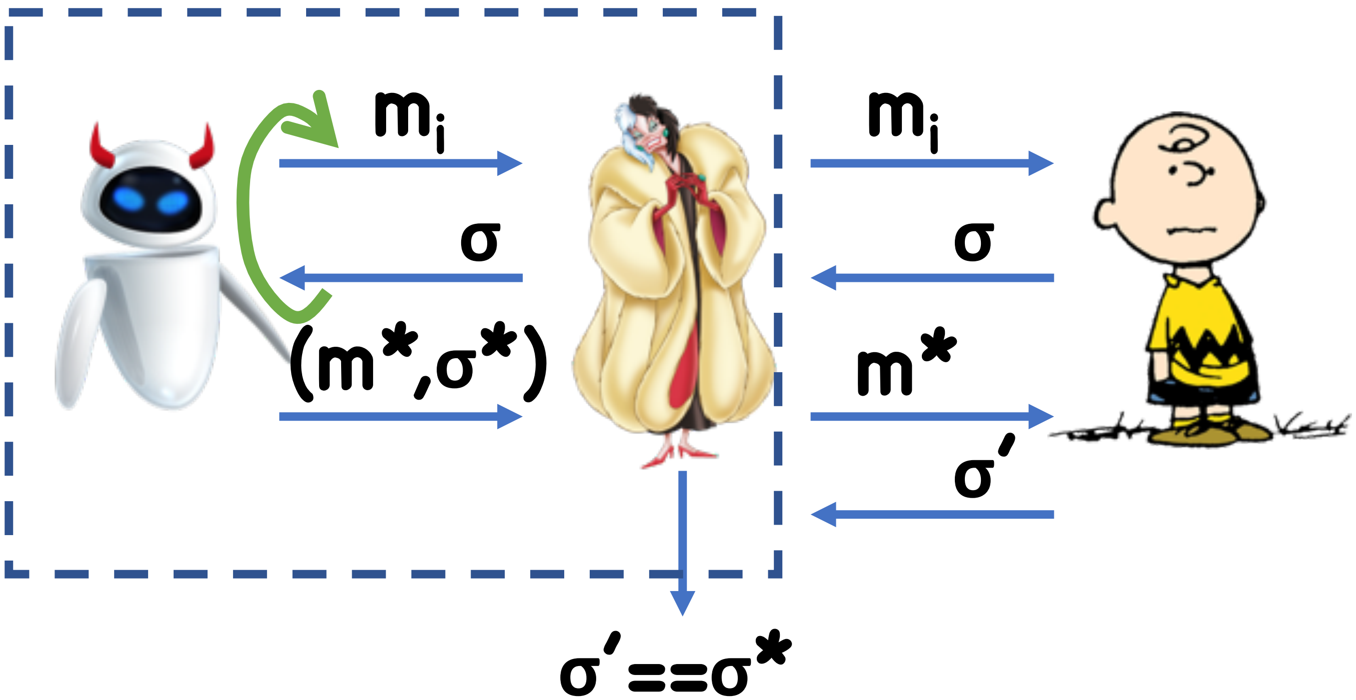
Claim: in Hybrid 1, output 1 with probability $\mathbf{1/|T_\lambda|}$

-  sees values of $\mathbf{H}$ on points $\mathbf{m_i}$
- Value on $\mathbf{m^*}$ independent of  ‘s view
- Therefore, probability $\mathbf{\sigma^*=H(m^*) = 1/|T_\lambda|}$

Security Proof

Claim: **$|\Pr[1 \leftarrow \text{Hyb1}] - \Pr[1 \leftarrow \text{Hyb2}]| \leq \varepsilon(\lambda)$**

Suppose not, construct PRF adversary 



MACs/PRFs for Larger Domains

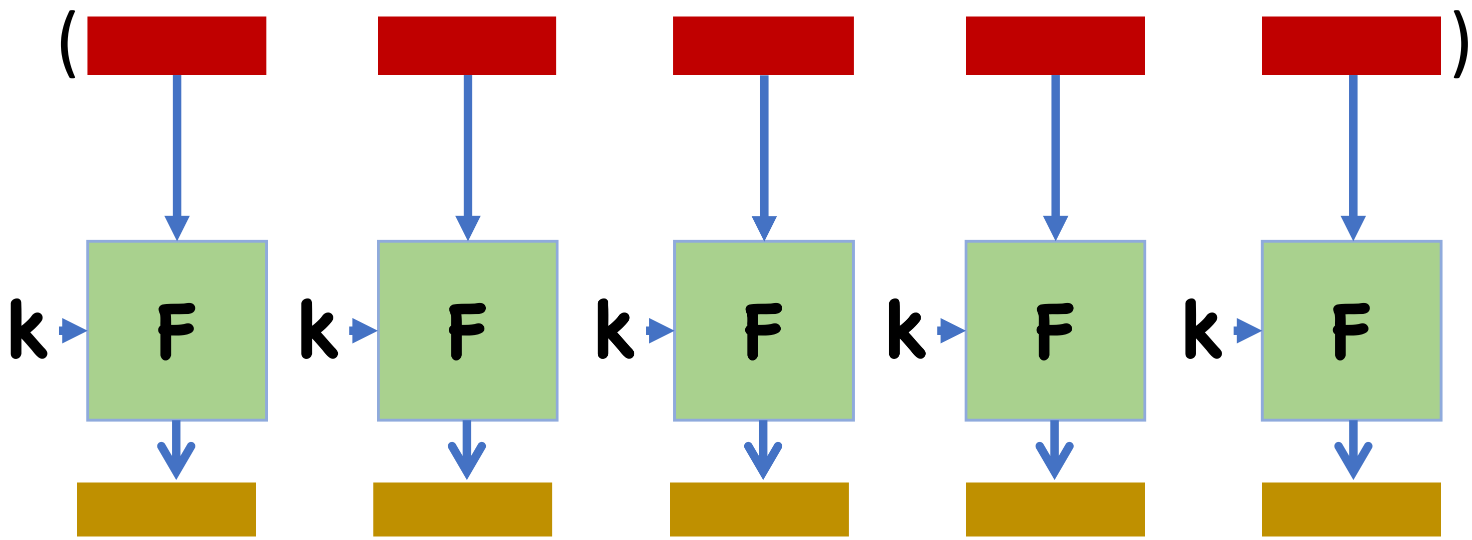
We saw that block ciphers are good PRFs

However, the input length is generally fixed

- For example, AES maximum block length is 128 bits

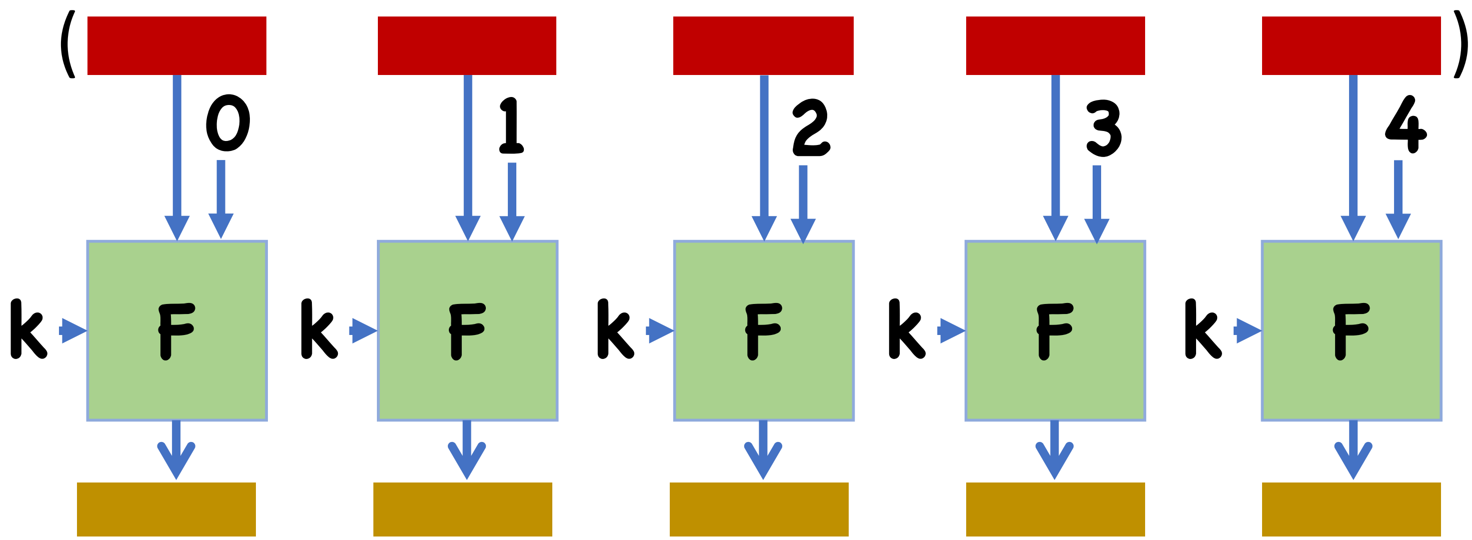
How do we handle larger messages?

Block-wise Authentication?



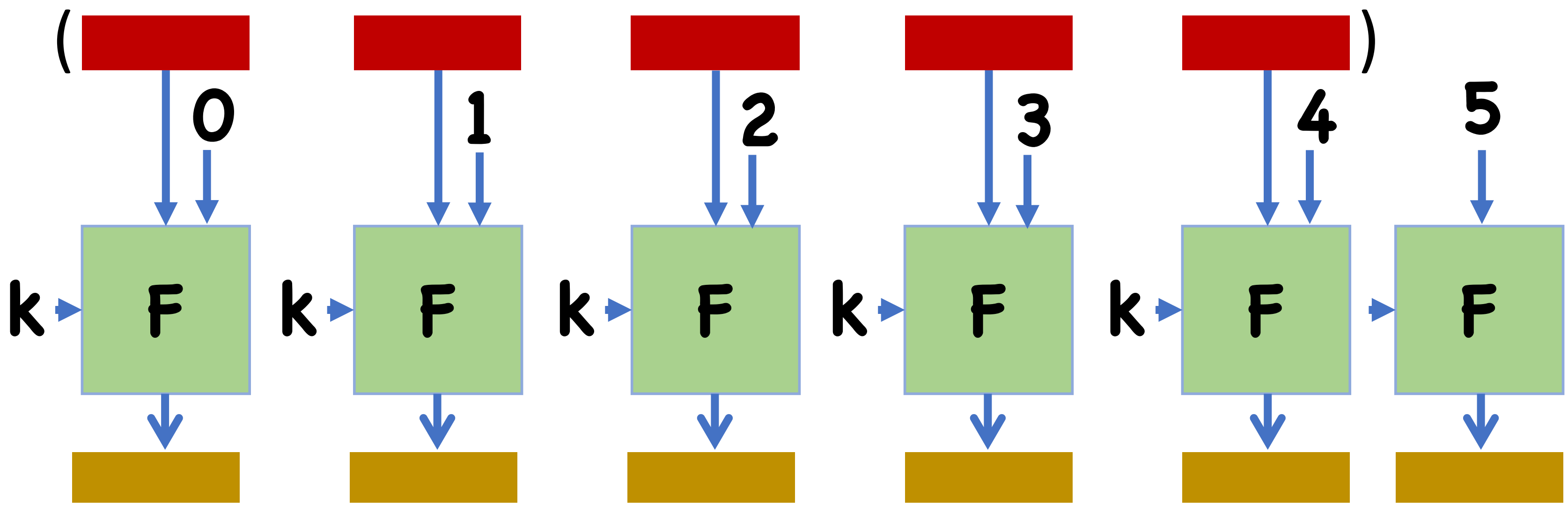
Why is this insecure?

Block-wise Authentication?



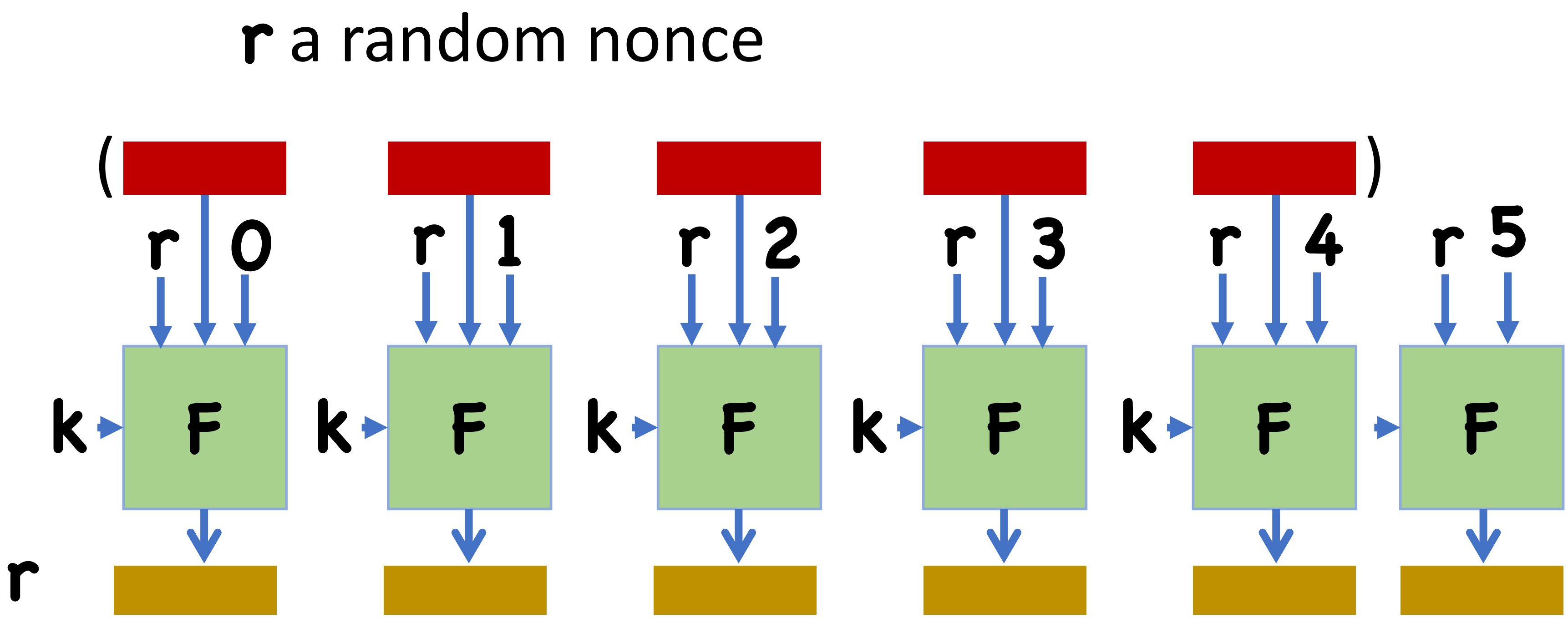
Why is this insecure?

Block-wise Authentication?



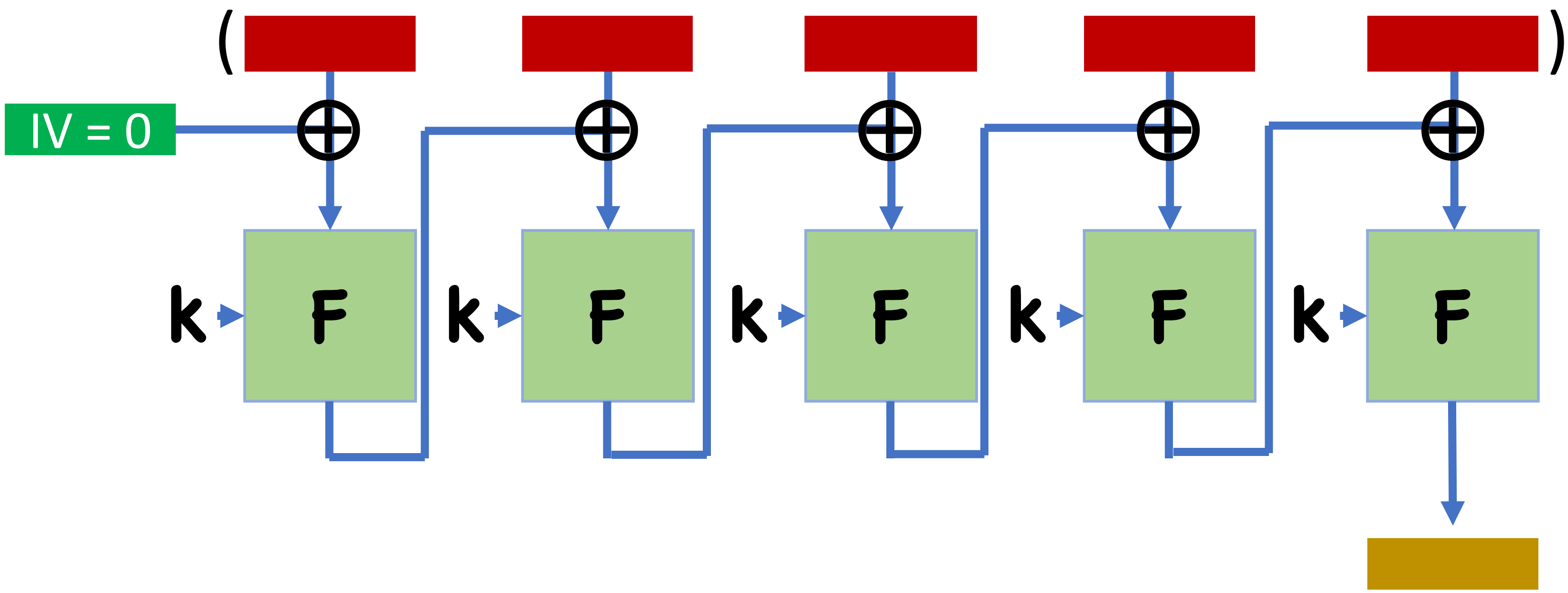
Why is this insecure?

Block-wise Authentication?



Secure, but not very useful in practice

CBC-MAC



Theorem: CBC-MAC is a secure PRF for **fixed-length** messages

Timing Attacks on MACs

How do you implement check **F(k,m)==σ**?

String comparison often optimized for performance

Compare(A,B):

- **For i = 1,…,A.length**
  - **If A[i] != B[i], abort and return False;**
- **Return True;**

Time depends on number of initial bytes that match

Timing Attacks on MACs

To forge a message **m**:

For each candidate first byte $\boldsymbol{\sigma_0}$:

- Query server on **(m, σ)** where first byte of **σ** is $\boldsymbol{\sigma_0}$
- See how long it takes to reject

First byte is $\boldsymbol{\sigma_0}$ that causes the longest response

- If wrong, server rejects when comparing first byte
- If right, server rejects when comparing second

Timing Attacks on MACs

To forge a message **m**:

Now we have first byte $\boldsymbol{\sigma_0}$

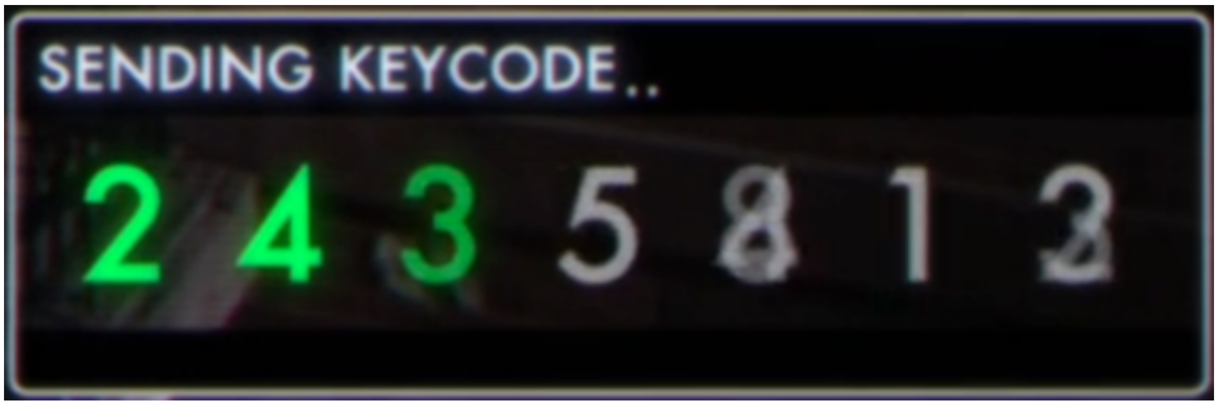
For each candidate second byte $\boldsymbol{\sigma_1}$:

- Query server on **(m, σ)** where first two bytes of **σ** are $\boldsymbol{\sigma_0, \sigma_1}$
- See how long it takes to reject

Second byte is $\boldsymbol{\sigma_1}$ that causes the longest response

…

Holiwudd Criptoe!



Most likely not what was meant by Hollywood, but conceivable

Thwarting Timing Attacks

Possibility:

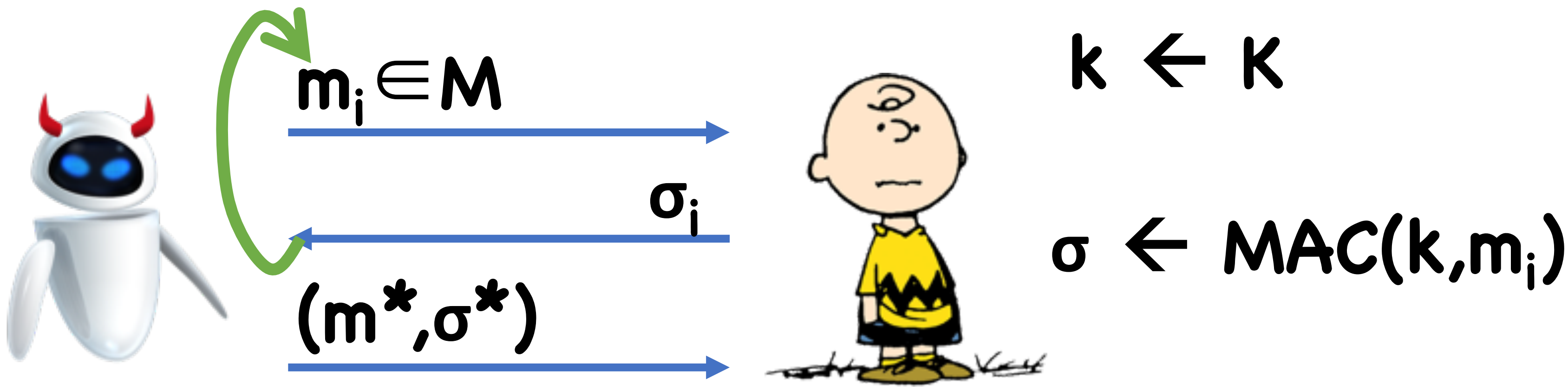
- Use a string comparison that is guaranteed to take constant time
- Unfortunately, this is hard in practice, as optimized compilers could still try to shortcut the comparison

Possibility:

- Choose random block cipher key **k'**
- Compare by testing **F(k',A) == F(k', B)**
- Timing of "**==**" independent of how many bytes **A** and **B** share

Alternate security notions

Strongly Secure MACs



Output 1 iff:

- $(m^*,\sigma^*) \notin \{(m_1,\sigma_1),\ldots\}$
- $Ver(k,m^*,\sigma^*) = 1$

SCMA-Adv() = Pr[ outputs 1]

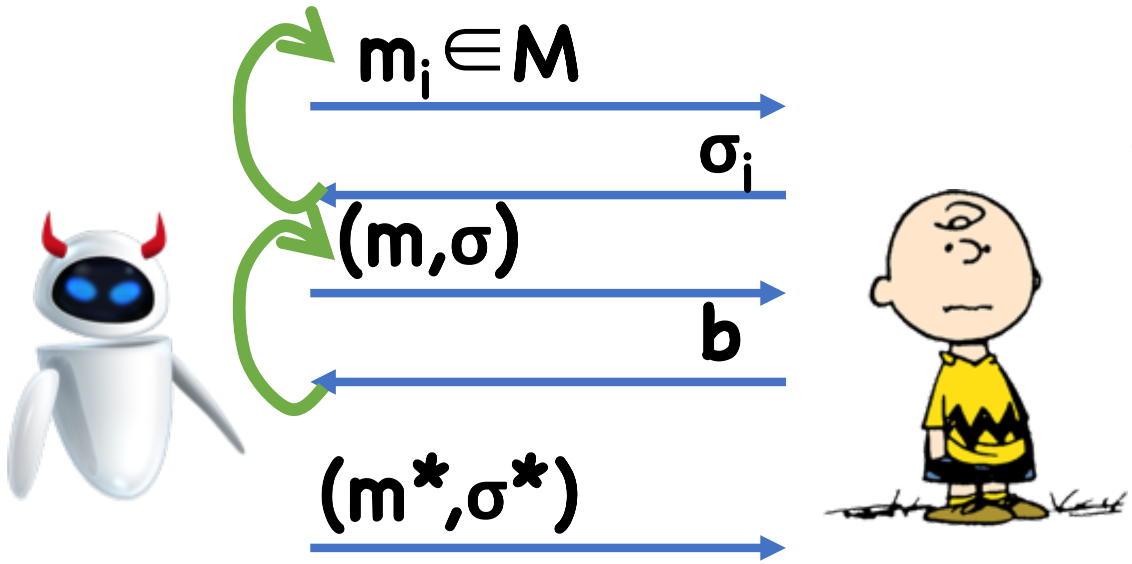
Strongly Secure MACs

Useful when you don’t want to allow the adversary to change *any* part of the communication

If there is only a single valid tag for each message (such as in the PRF-based MAC), then (weak) security also implies strong security

In general, though, strong security is stronger than weak security

Adding Verification Queries



$k \leftarrow K$

$\sigma_i \leftarrow MAC(k,m_i)$

$b \leftarrow Ver(k,m,\sigma)$

Output 1 iff:
- $m^* \notin \{m_1,\ldots\}$
- $Ver(k,m^*,\sigma^*) = 1$

CMA′-Adv() = Pr[ outputs 1]

Theorem: (MAC,Ver) is strongly CMA secure if and only if it is strongly CMA' secure

Improving efficiency

Limitations of CBC-MAC

Many block cipher evaluations

Sequential

Carter Wegman MAC

k' = (k,h) where:

- **k** is a PRF key for $\mathbf{F: K \times R \rightarrow Y}$
- **h** is sampled from a pairwise independent function family

MAC(k',m):

- Choose a random $\mathbf{r \leftarrow R}$
- Set $\boldsymbol{\sigma} = \mathbf{(r,\ F(k,r) \oplus h(m))}$

Theorem: If $\mathbf{F}$ is secure and $|\mathbf{T}|,|\mathbf{R}|$ are super-polynomial, then the Carter Wegman MAC is strongly CMA secure

Efficiency of CW MAC

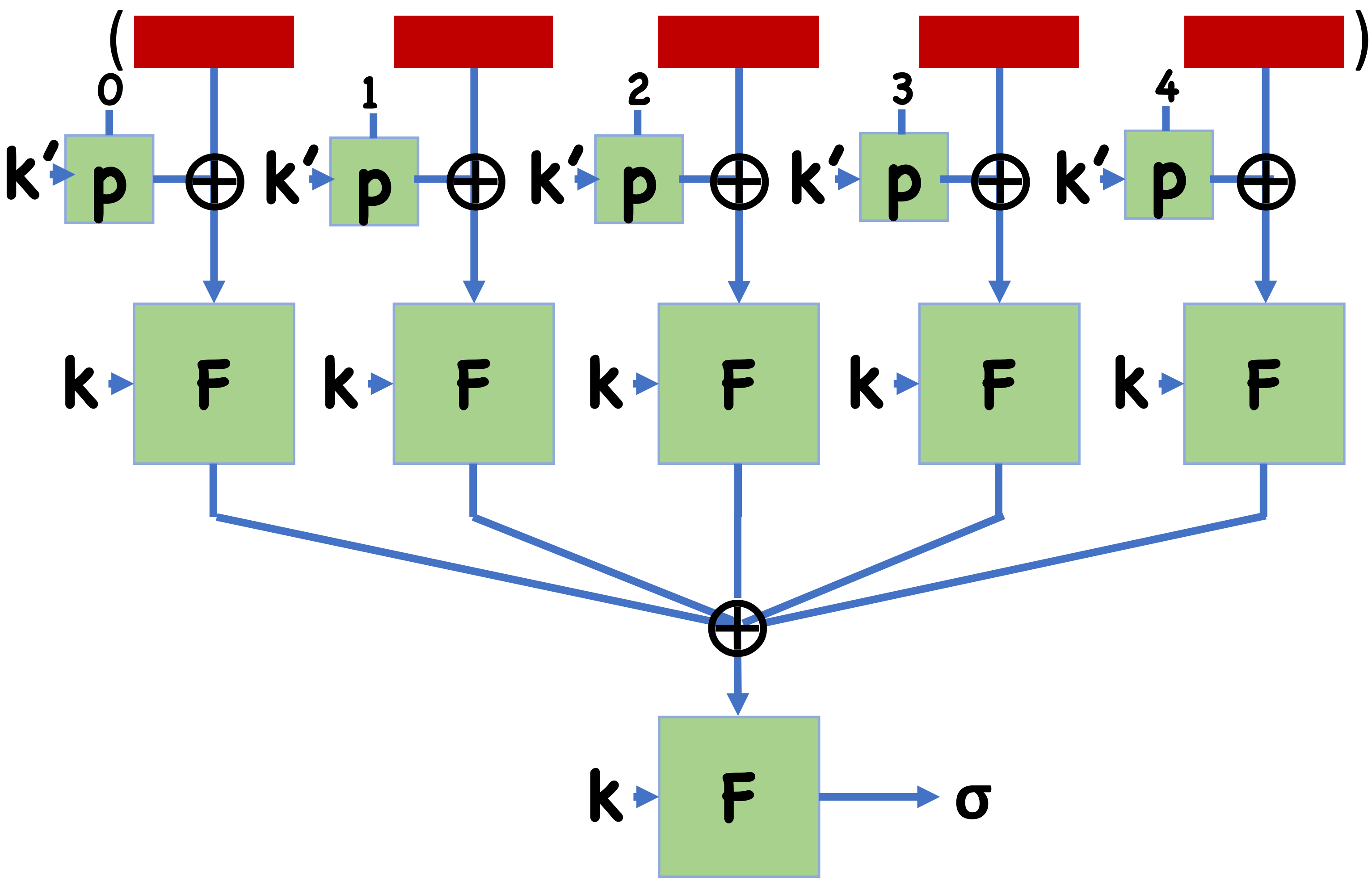
MAC(k',m):

- Choose a random **r←R**
- Set **σ = (r, F(k,r)⊕h(m))**

h much more efficient that PRFs

PRF applied only to small nonce **r**
h applied to large message **m**

PMAC: A Parallel MAC



Announcements/Reminders

PR1 Due TODAY

HW3 due on Oct 20