COS433/Math 473: Cryptography

Mark Zhandry

Princeton University

Spring 2020

Reminders

PR1 Due **Thursday**

- No late days

Previously on COS 433…

Brute Force Attacks

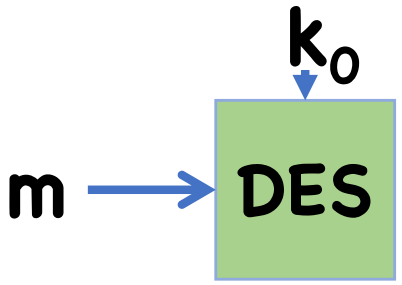
Suppose attacker is given a few input/output pairs

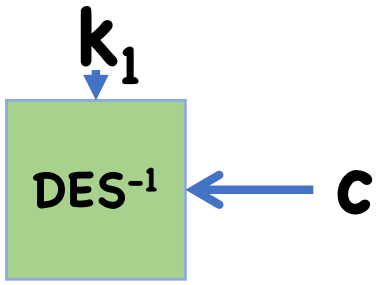
Likely only one key could be consistent with this input/output

Brute force search: try every key in the key space, and check for consistency

Attack time: $2^{\text{key length}}$

Meet In The Middle Attacks

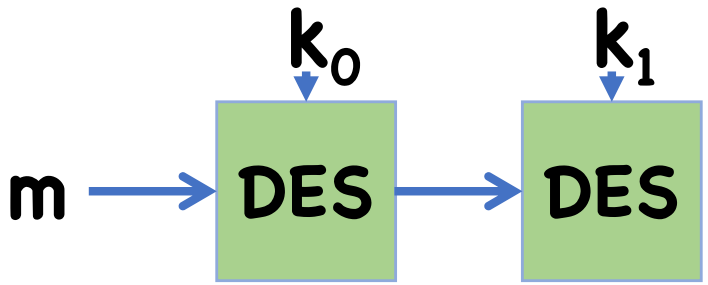


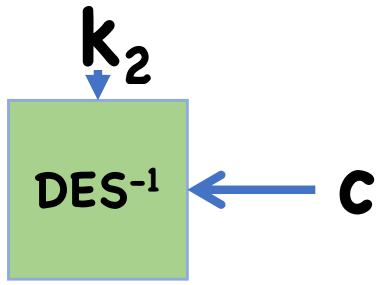


k_0	$d = DES(k_0,m)$
0	52
1	93
2	03
3	96
4	20
5	49
…	…

k_1	$d = DES^{-1}(k_1,m)$
0	69
1	10
2	86
3	49
4	99
5	08
…	…

MITM for 3DES



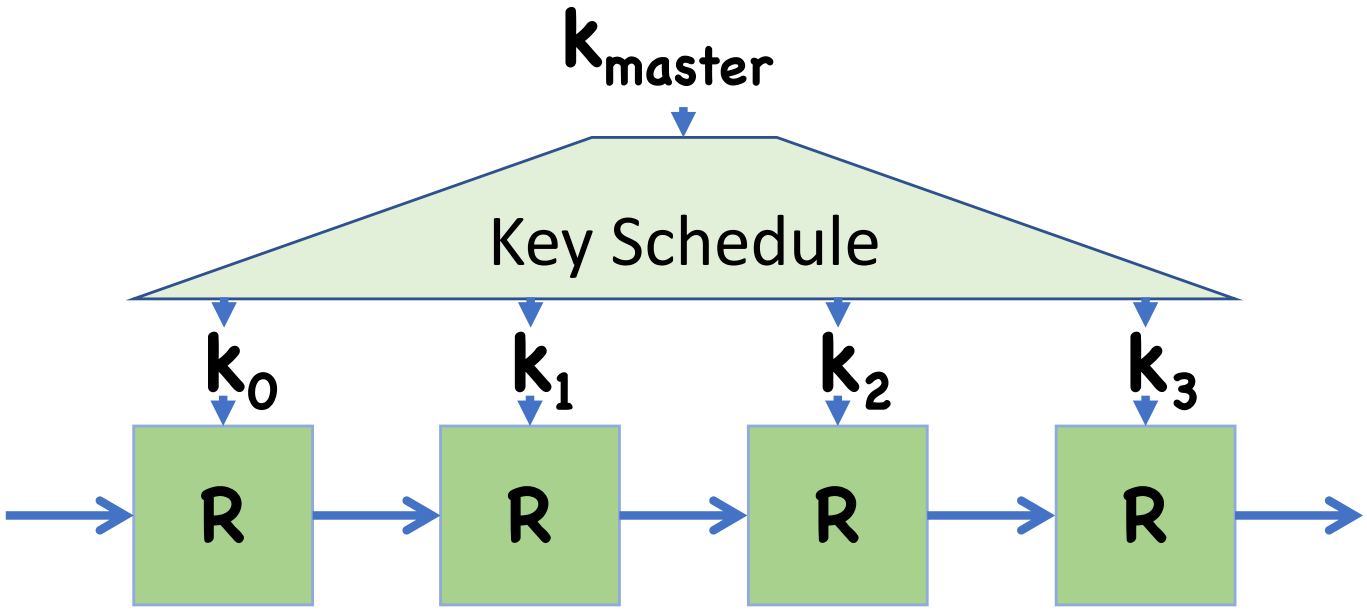


k_0	k_1	$d = DES(k_0,m)$
0	0	52
0	1	93
…	…	03
5	6	96
5	7	20
5	8	49
…		…

k_2	$d = DES^{-1}(k_2,m)$
0	69
1	10
2	86
3	49
4	99
5	08
…	…

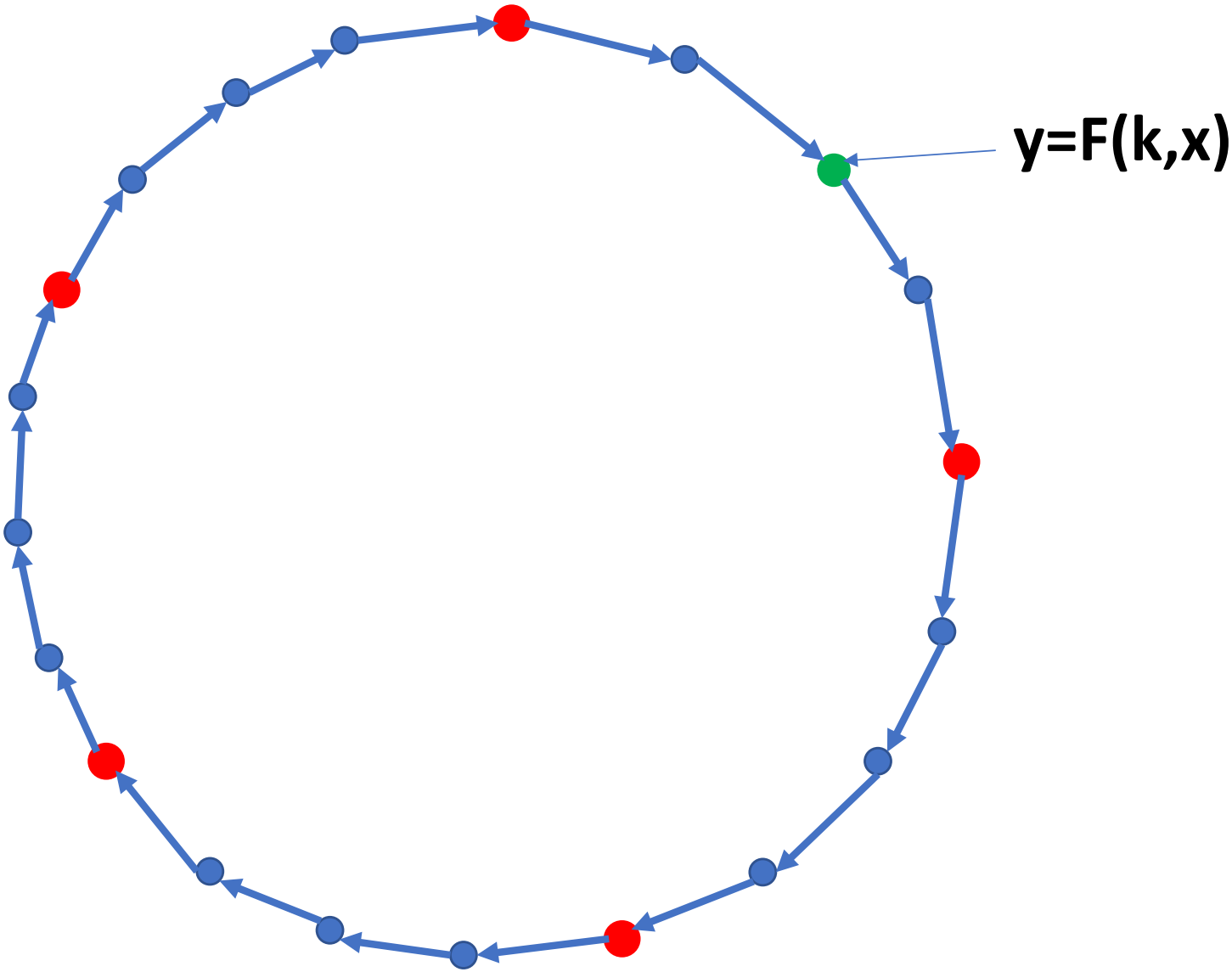
MITM Attacks

MITM attacks can also be applied to plain single block ciphers



Can yield reasonable attacks in some regimes

Option 3: Hellman's Attack



Today

Continue differential cryptanalysis
Other attacks on block ciphers

Message integrity

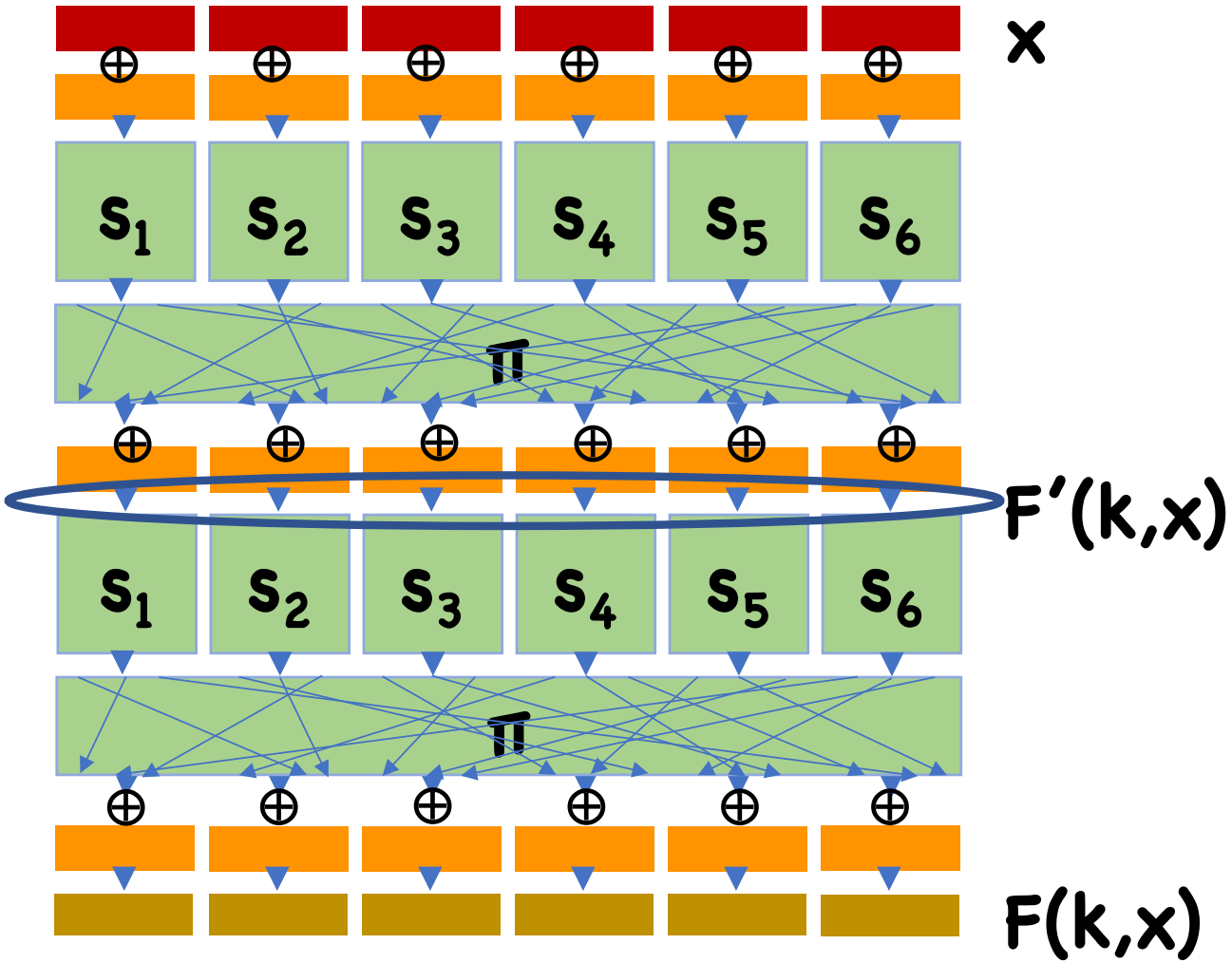
Differential Cryptanalysis

Suppose there were Δ_x, Δ_z such that, for random key k and random x_1, x_2 such that $x_1 \oplus x_2 = \Delta_x$, $F(k,x_1) \oplus F(k,x_2) = \Delta_z$ with probability $p \gg 2^{-n}$

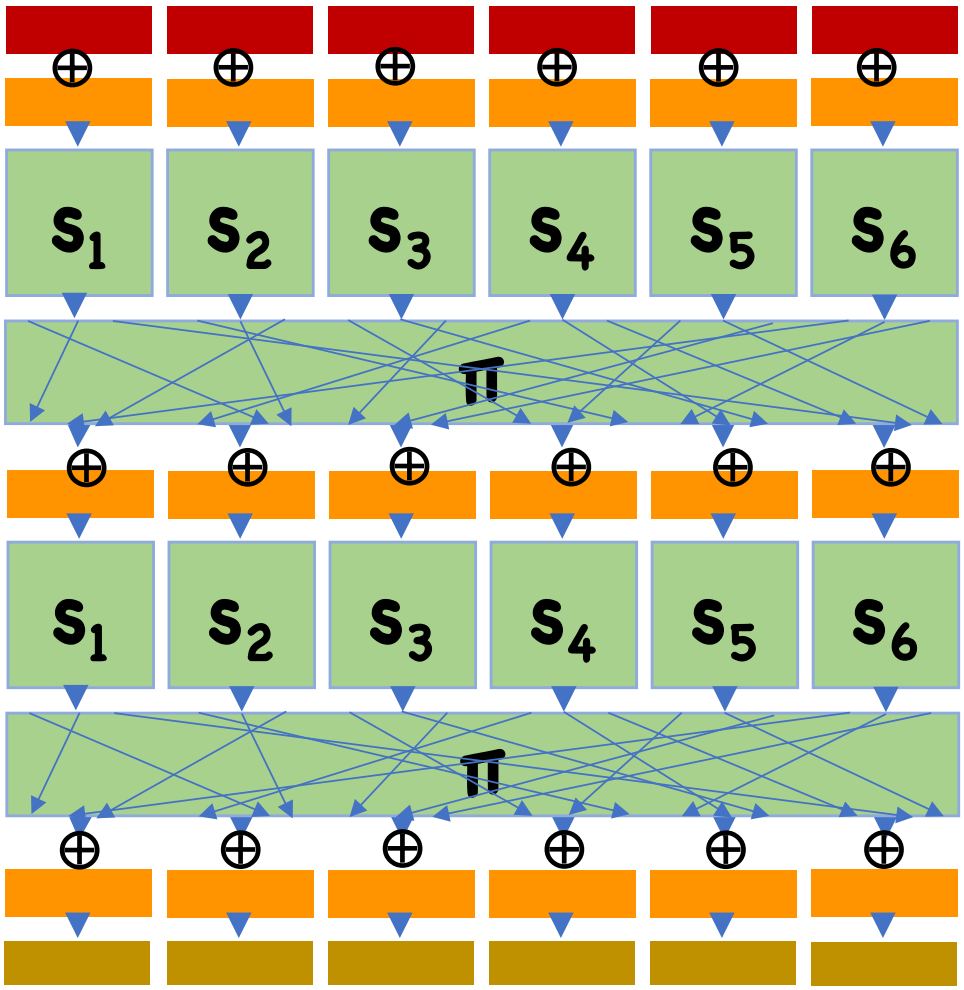
- Call (Δ_x, Δ_z) a differential
- p is probability of differential
- $\approx 2^{-n}$ is probability of differential for random permutation

Yields distinguishing attack. With some effort, can also recover secret key

Differential Cryptanalysis



Differential Cryptanalysis



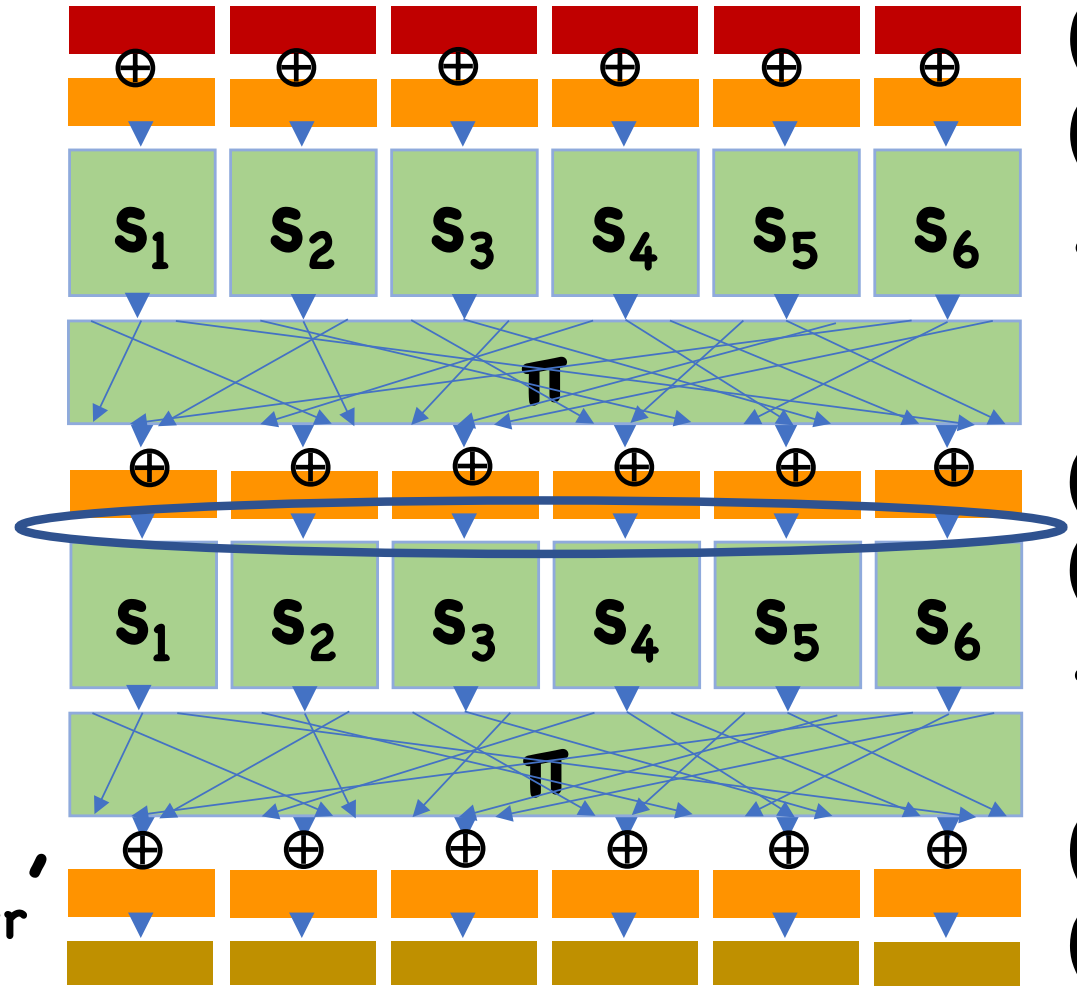
(x_0^1, x_1^1), (x_0^2, x_1^2),
(x_0^3, x_1^3), (x_0^4, x_1^4),
…

PRP Oracle

Guess k_r'

(y_0^1, y_1^1), (y_0^2, y_1^2),
(y_0^3, y_1^3), (y_0^4, y_1^4),
…

Differential Cryptanalysis



(x_0^1, x_1^1), (x_0^2, x_1^2),
(x_0^3, x_1^3), (x_0^4, x_1^4),
…

$(z'_0{}^1, z'_1{}^1)$, $(z'_0{}^2, z'_1{}^2)$,
$(z'_0{}^3, y'_1{}^3)$, $(z'_0{}^4, z'_1{}^4)$,
…

k_r'

(y_0^1, y_1^1), (y_0^2, y_1^2),
(y_0^3, y_1^3), (y_0^4, y_1^4),
…

Guess k_r'

Differential Cryptanalysis

Extending to further levels:

- One k_r is known, can un-compute last layer
- Now perform same attack on round-reduced cipher
- Repeat until all round keys have been found

Finding Differentials

So far, assumed differential given

How do we find it?

- Can’t simply brute force all possible differentials

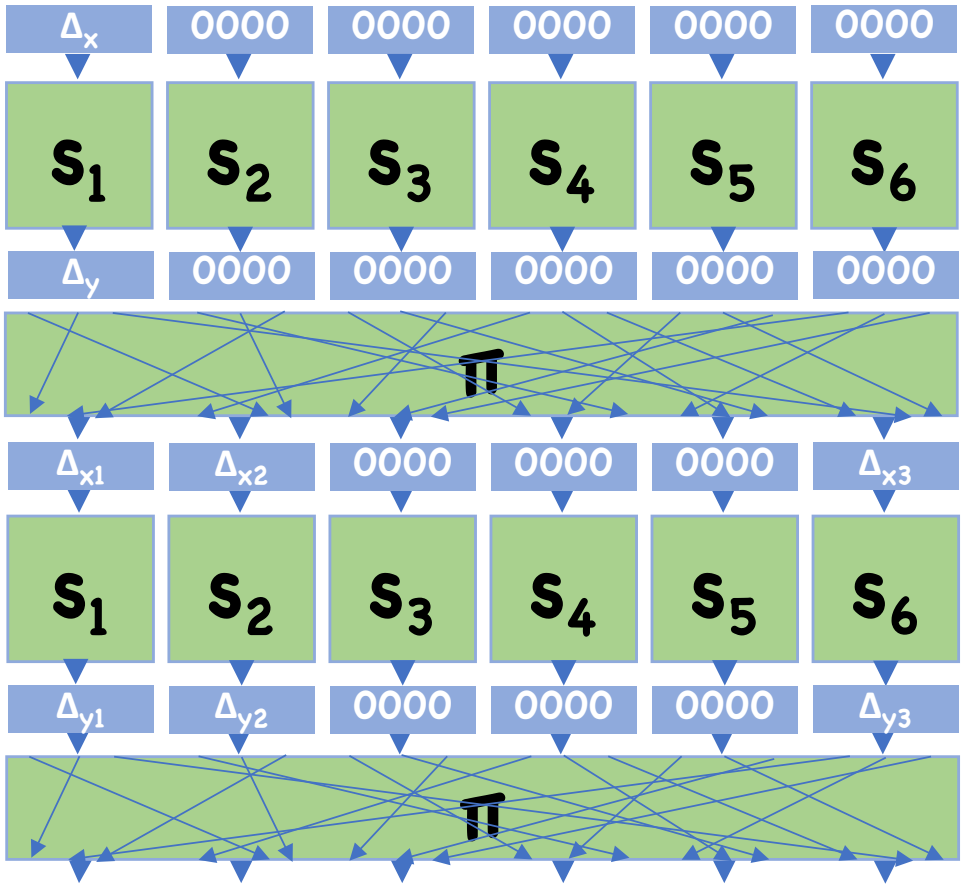
Finding Differentials

Solution: look for differentials in S-boxes

- Only 2^8 possible differences, so we can actually look for all possible differentials

- Then trace differentials through the evaluation
  - Key mixing does not affect differentials
  - Diffusion steps just shuffle differential bits

Differential Cryptanalysis



...

Differential Cryptanalysis in Practice

Used to attack real ciphers

- FEAL-8, proposed as alternative to DES in 1987
  - requires just 1000 chosen input/output pairs, 2 minutes computation time in 1990's

- Also theoretical attacks on DES
  - Requires 2^{47} chosen input/output pairs
  - Very difficult to obtain in real world applications
  - Therefore, DES is still considered relatively secure
  - Small changes to S-boxes in DES lead to much better differential attacks

Linear Cryptanalysis

High level idea: look for linear relationships that hold with too-high a probability

- E.g. $x_1 \oplus x_5 \oplus x_{17} \oplus y_3 \oplus y_6 \oplus y_{12} \oplus y_{21} = 0$

Can show that if happen with too-high probability, can completely recover key

Important feature: only requires *known* plaintext as opposed to *chosen* plaintext

- Much easier to carry out in practice
- Ex: DES can be broken with 2^{43} input/output pairs

Block Cipher Design

S-boxes are designed to minimize differential and linear cryptanalysis

- Cannot completely remove differentials/linear relations, but can minimize their probability

Increasing number of rounds helps

- Likelihood of differential decreases each round

Related Key Attacks

Properly designed crypto will always use random, independent keys for every application

However, sometimes people don't follow the rules

Related key attack: have messages encrypted under similar keys

(Recall RC4 used for encryption, **RC4(IV,k)**)

For AES 256, can attack in 2^{110} space/time

Limitations of Feistel Networks

Turns out Feistel requires block size to be large

- If number of queries ~$2^{\text{block size}/2}$, can attack

Format preserving encryption:

- Encrypted data has same form as original
- E.g. encrypted SSN is an SSN
- Useful for encrypting legacy databases

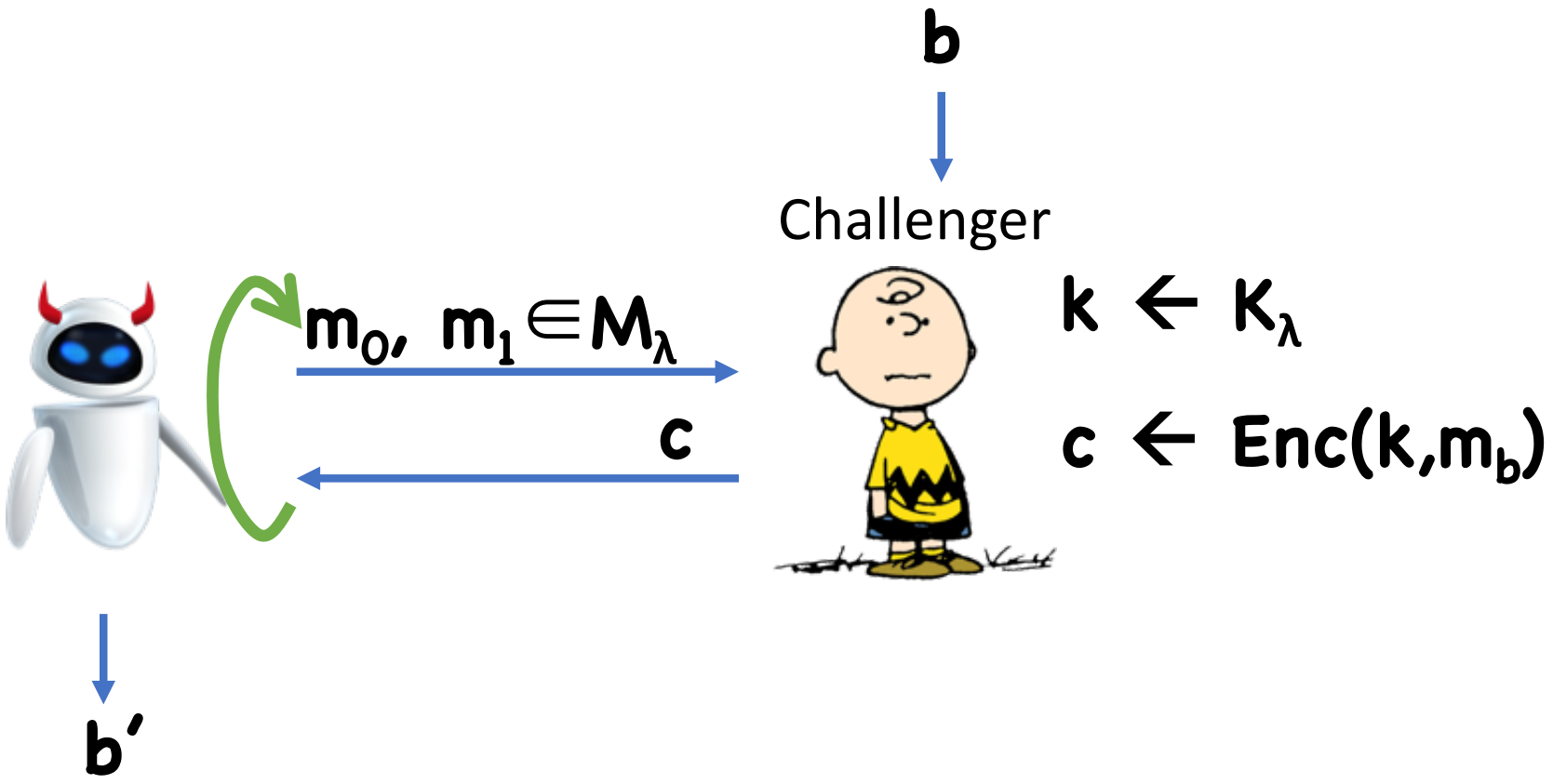
Sometimes, want a very small block size

Holiwudd Criptoe!

Device is top of the line.
AES cipher locks, brute force decryption is the only way…. It's effective, but slow. Very slow.

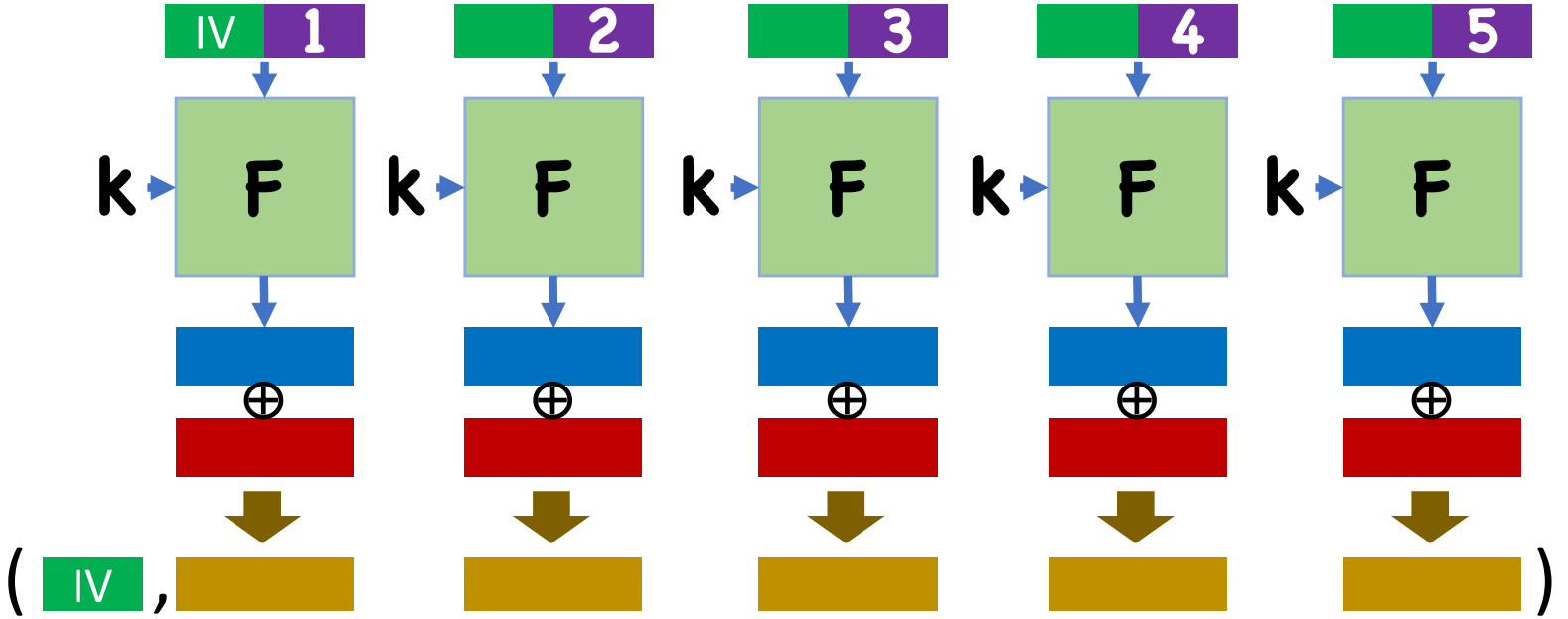
Message Integrity

Recall: CPA Security



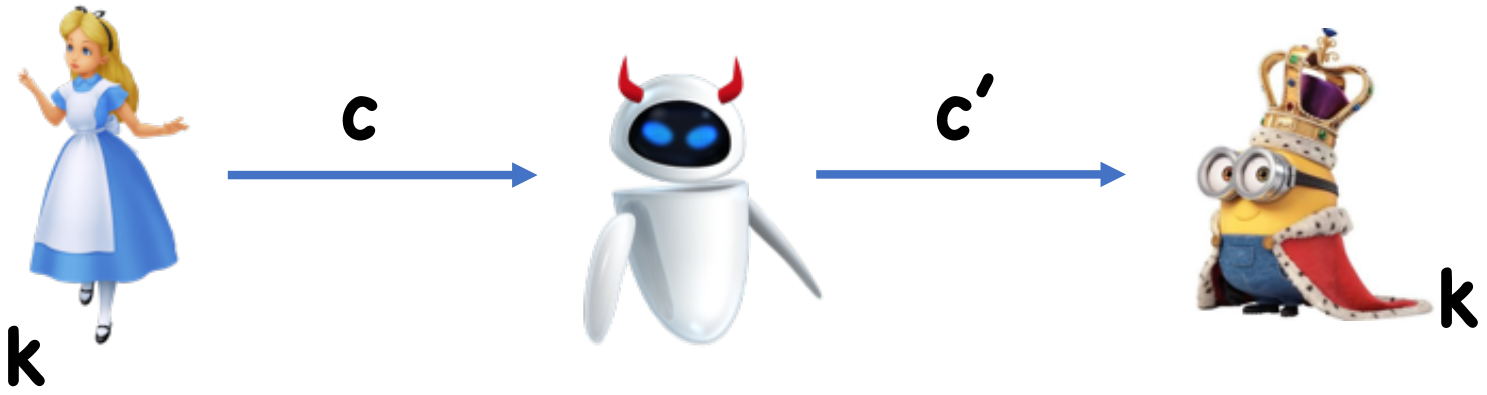
$LoR\text{-}Exp_b($  $, \lambda)$

Recall: Counter Mode (CTR)



Limitations of CPA security

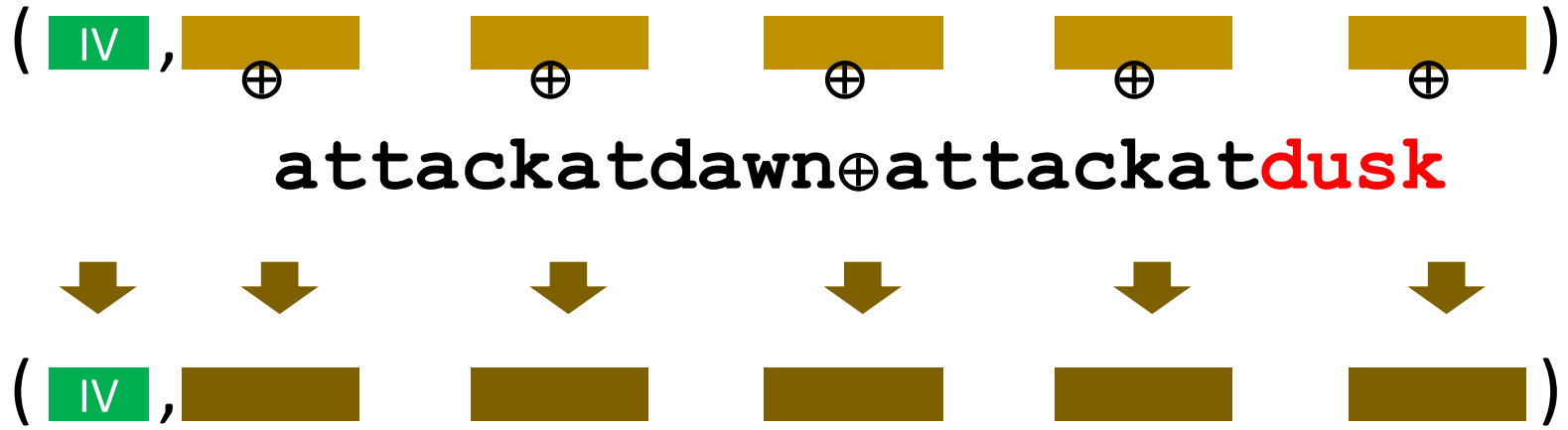
attackatdawn



attackatdusk

How?

Limitations of CPA Security



Malleability

Some encryption schemes are *malleable*

- Can modify ciphertext to cause predictable changes to plaintext

Examples: basically everything we've seen so far

- Stream ciphers
- CTR
- CBC
- ECB
- …

Message Integrity

We cannot stop adversary from changing the message in route to Bob

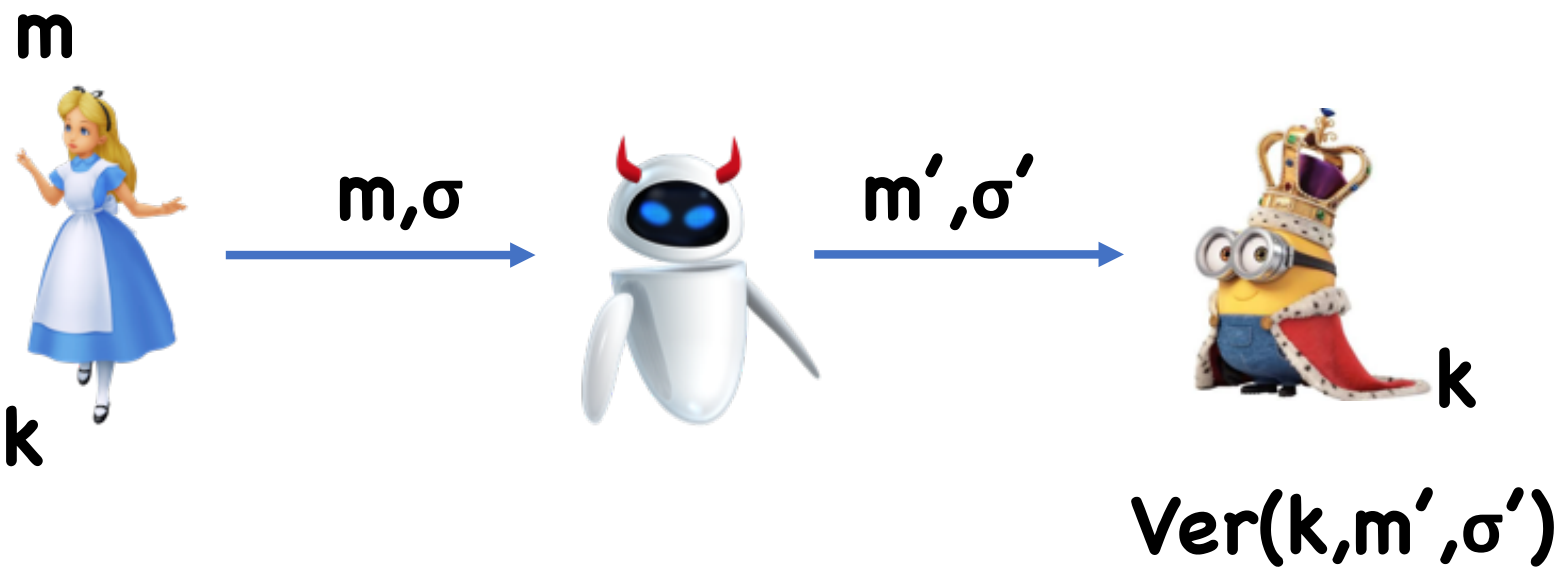
However, we can hope to have Bob perform some check on the message he receives to ensure it was sent by Alice and not modified

- If check fails, Bob rejects the message

For now, we won't care about message secrecy

- We will add it back in later

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

$m \in M_\lambda$

σ

(m^*,σ^*)

$k \leftarrow K_\lambda$

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

Output 1 iff:

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

$1CMA\text{-}Adv($  $, \lambda) = Pr[$  $\text{ outputs } 1]$

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

$$\mathbf{1CMA\text{-}Adv(}\ \text{  }\ \mathbf{, \lambda) \leq \varepsilon(\lambda)}$$

Question

Is perfect security (**$\varepsilon=0$**) 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 $\mathbf{m_0 \neq m_1 \in M_\lambda,\ \sigma_0, \sigma_1 \in T_\lambda}$

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

K = H_λ

MAC(h, m) = h(m)

Ver(h,m,σ) = (h(m) == σ)

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

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

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

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

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

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

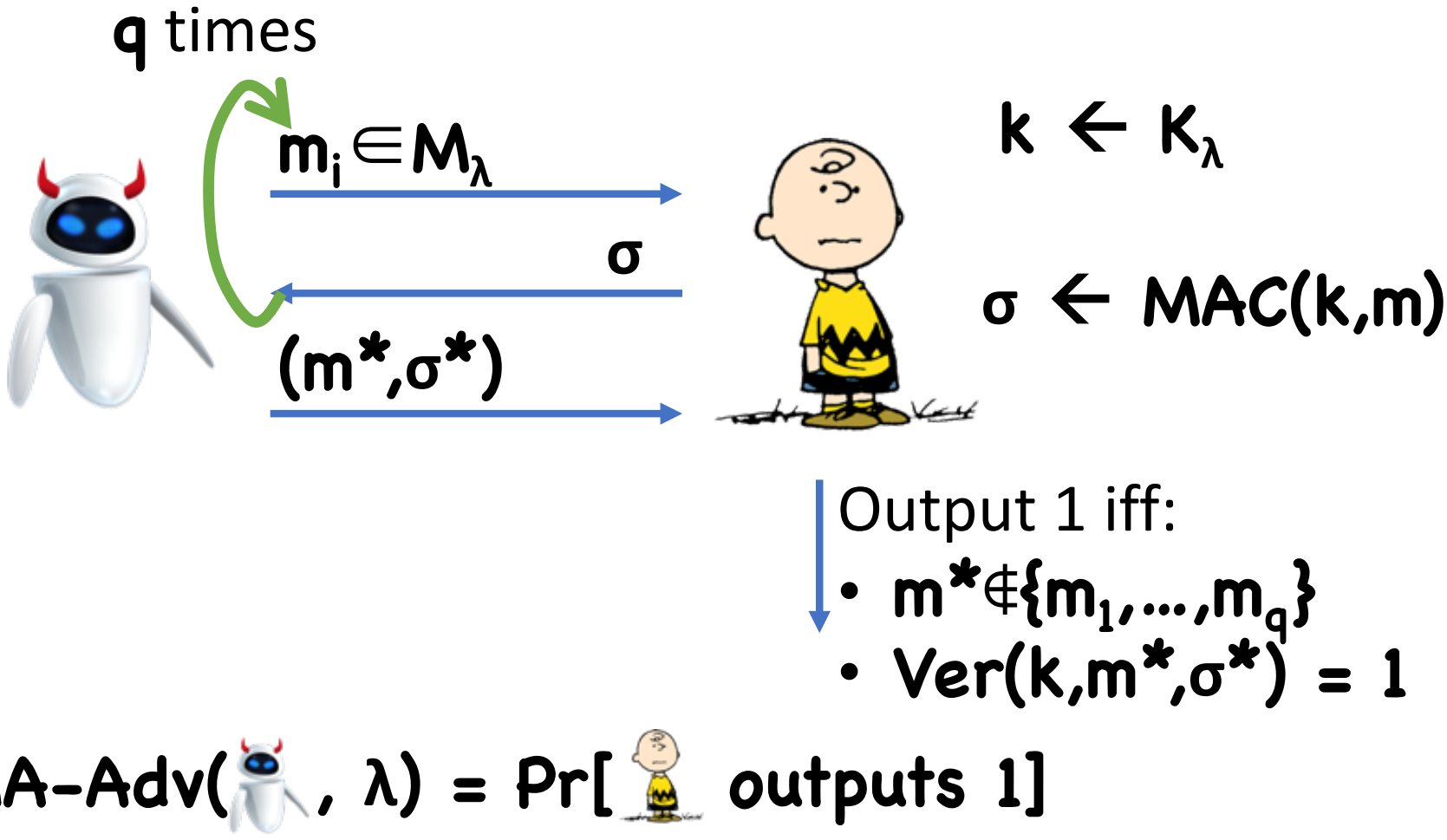
- $H_\lambda = \{h(x) = \langle a,x\rangle + b\colon a \in \mathbb{F}^n,\ b \in \mathbb{F}\}$

Multiple Use MACs?

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

Why?

q-Time MACs


q times
$m_i \in M_\lambda$
σ
(m*,σ*)
$k \leftarrow K_\lambda$
$\sigma \leftarrow MAC(k,m)$
Output 1 iff:
• $m^* \notin \{m_1,\ldots,m_q\}$
• $Ver(k,m^*,\sigma^*) = 1$
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, $\exists$ negligible **ε** such that:

$$\mathbf{CMA\text{-}Adv}(\text{  }, \lambda) \leq \varepsilon(\lambda)$$

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 $q \gg \log |K_\lambda|$ 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 r_q as follows:

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

$$\mathbf{CMA\text{-}Adv}(\text{  }, \lambda) \leq \varepsilon(\lambda)$$

Constructing MACs

Use a PRF

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

MAC(k,m) = F(k,m)
Ver(k,m,σ) = (F(k,m) == σ)

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

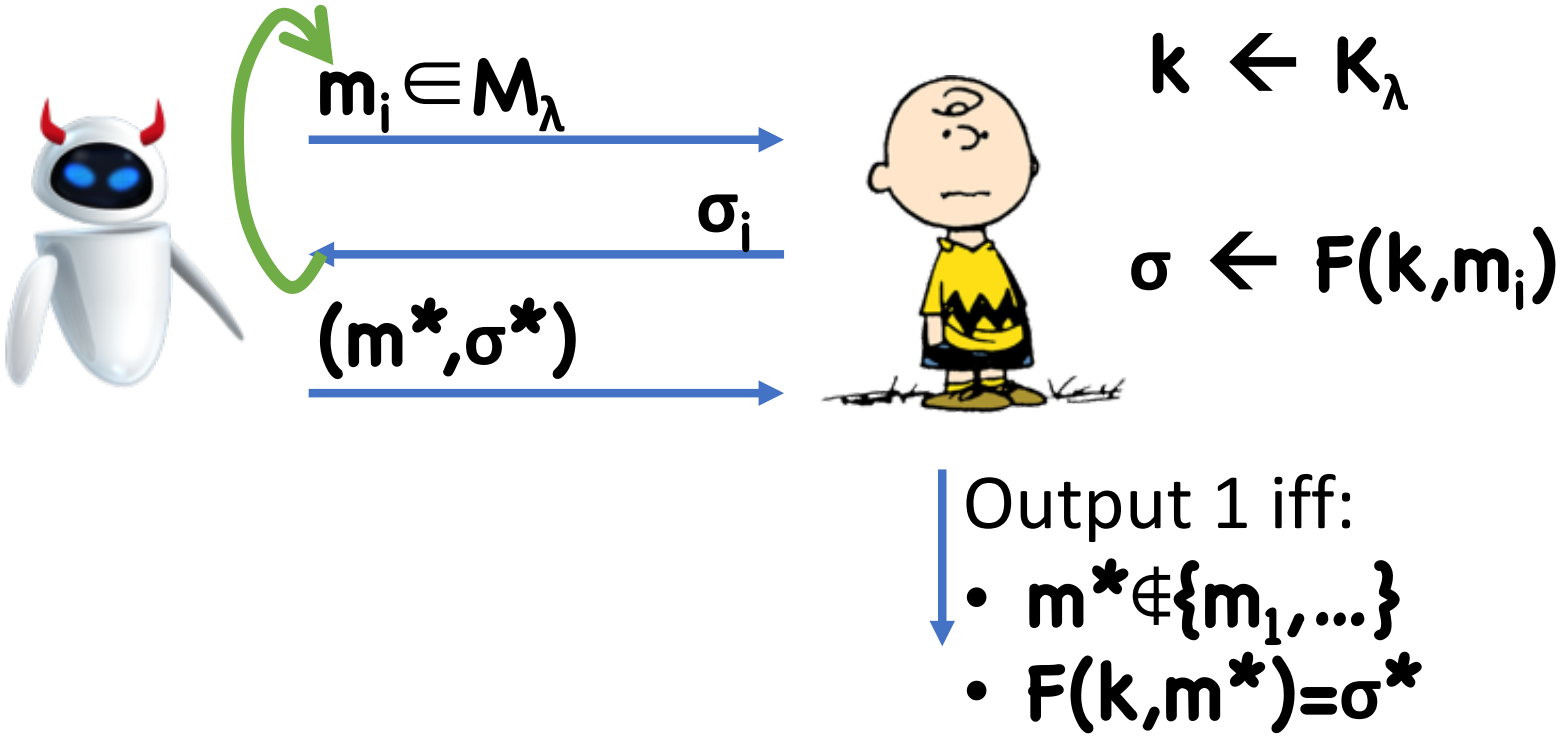
Security Proof

Assume toward contradiction polynomial time 

Hybrids!

Security Proof

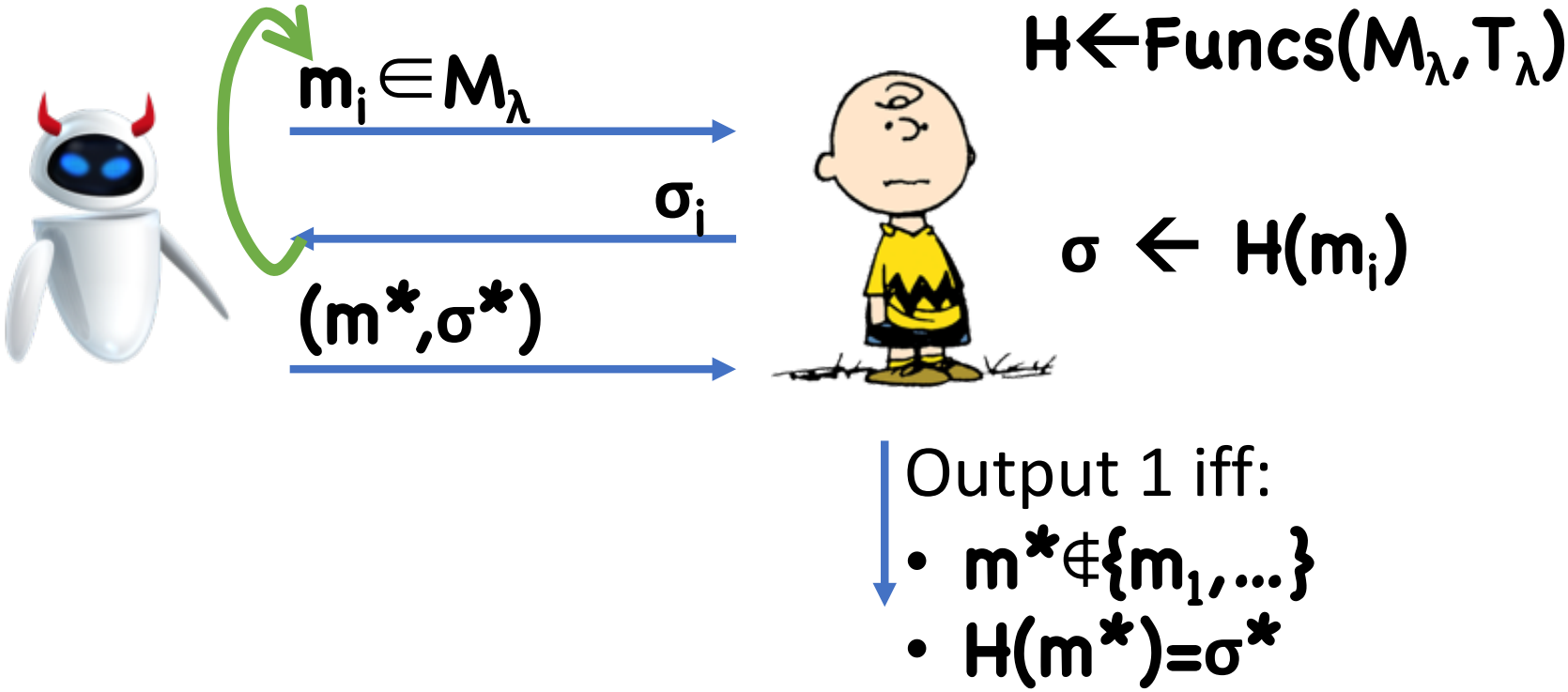
Hybrid 0



CMA Experiment

Security Proof

Hybrid 1



Security Proof

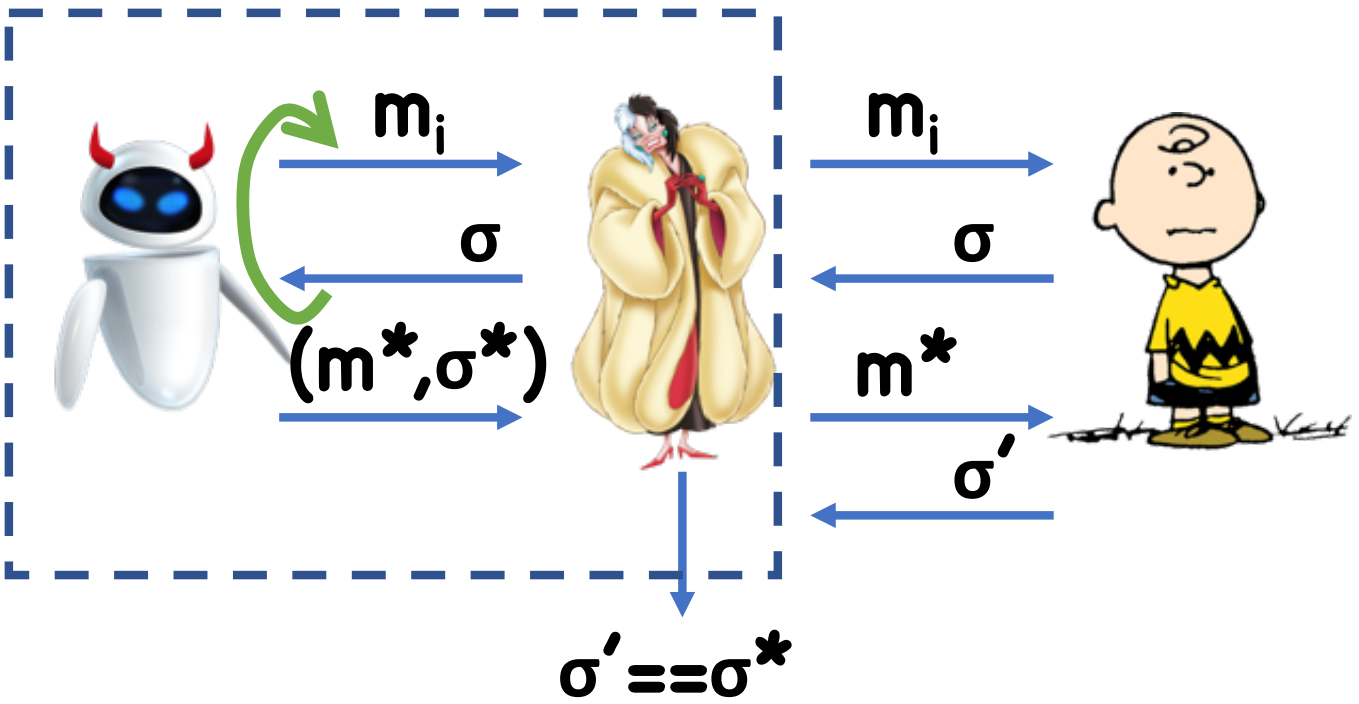
Claim: in Hybrid 1, output 1 with probability $1/|T_\lambda|$

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

Security Proof

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

Suppose not, construct PRF adversary 



MACs/PRFs for Larger Domains

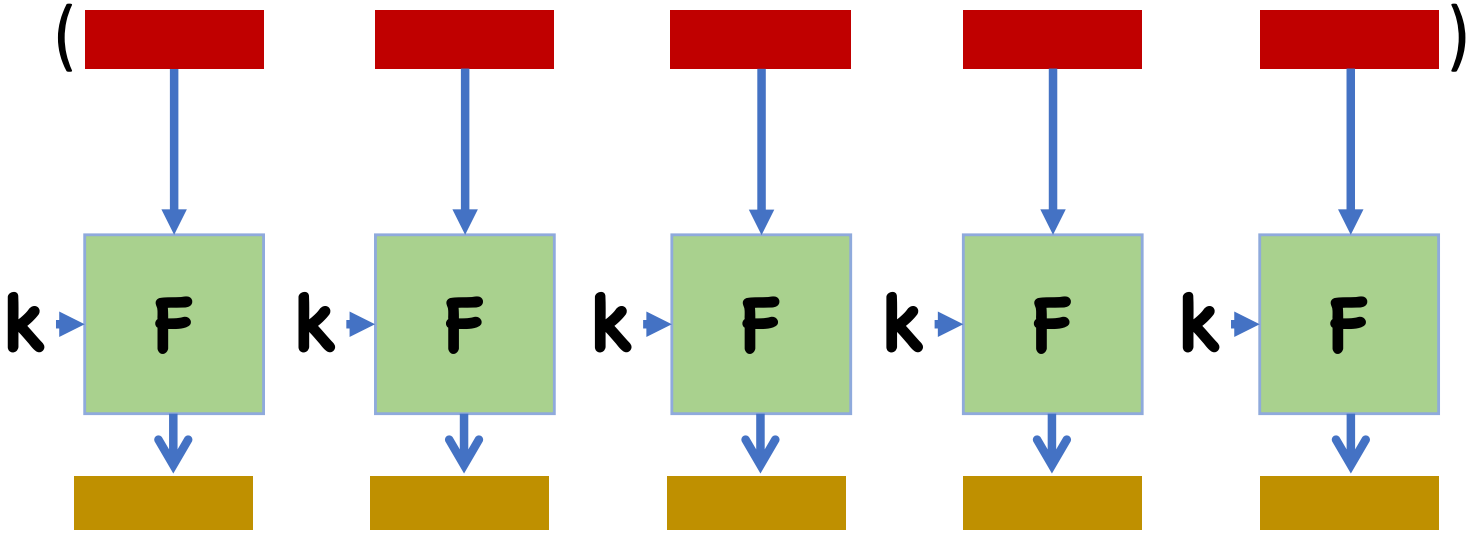
We saw that block ciphers are good PRFs

However, the input length is generally fixed

- For example, AES maximum block length is 128 bits

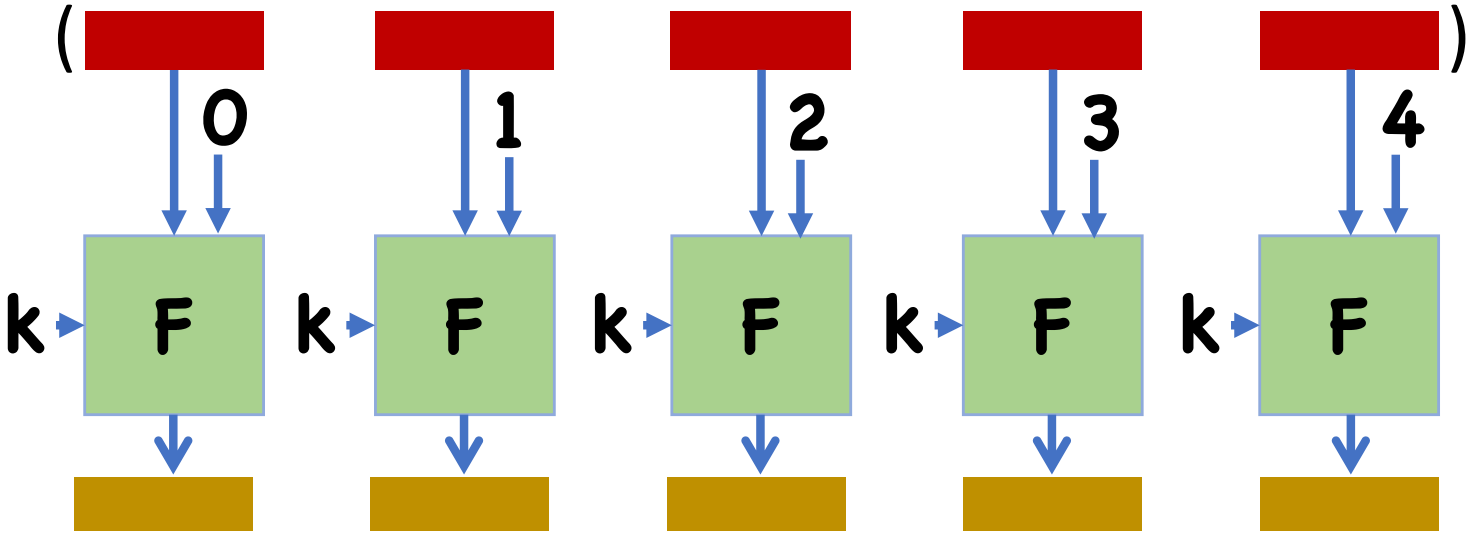
How do we handle larger messages?

Block-wise Authentication?



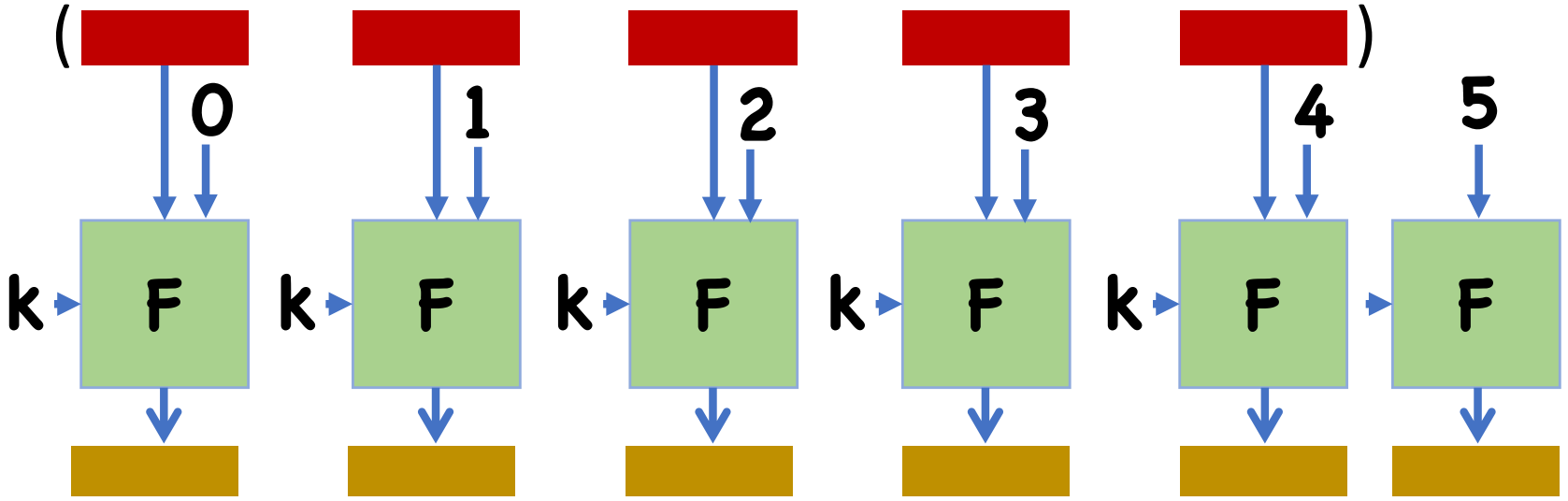
Why is this insecure?

Block-wise Authentication?



Why is this insecure?

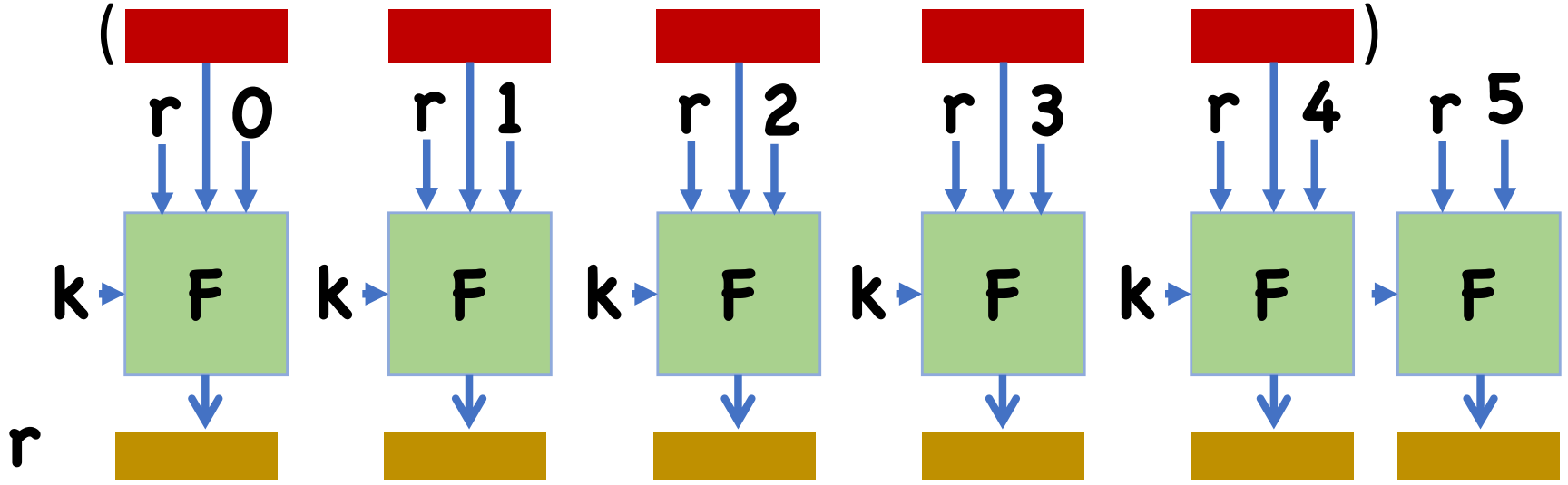
Block-wise Authentication?



Why is this insecure?

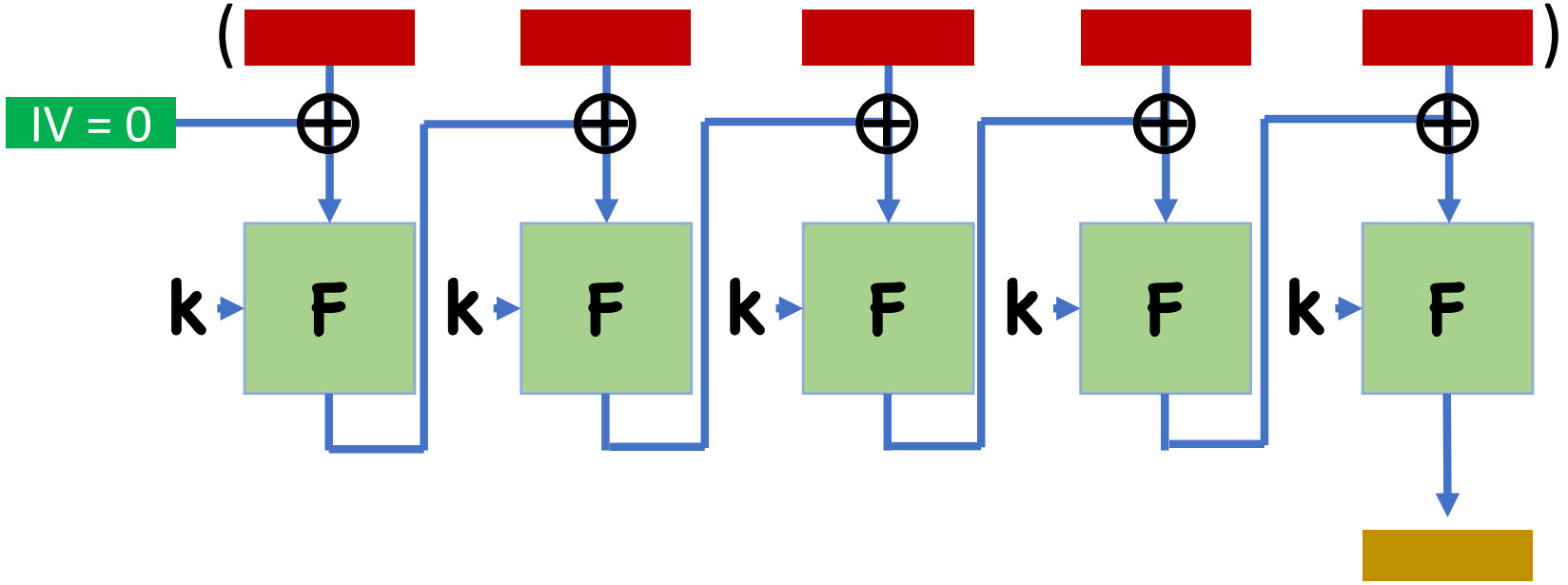
Block-wise Authentication?

r a random nonce



Secure, but not very useful in practice

CBC-MAC



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

Reminders

PR1 Due **Thursday**

- No late days