DATA PRIVACY AND SECURITY

Prof. Daniele Venturi

Master's Degree in Data Science
Sapienza University of Rome

CIS SAPIENZA
RESEARCH CENTER FOR CYBER INTELLIGENCE AND INFORMATION SECURITY

Security and Cryptography

- Involves capabilities from **different areas**
  - Mathematics, physics, computer science, networking, law, and more
- How to build a **secure** system?
  - Many aspects to consider
  - **Cryptography:** The heart of any secure system
  - **Other aspects:** Physical security, logical security, security governance, security of code and implementations

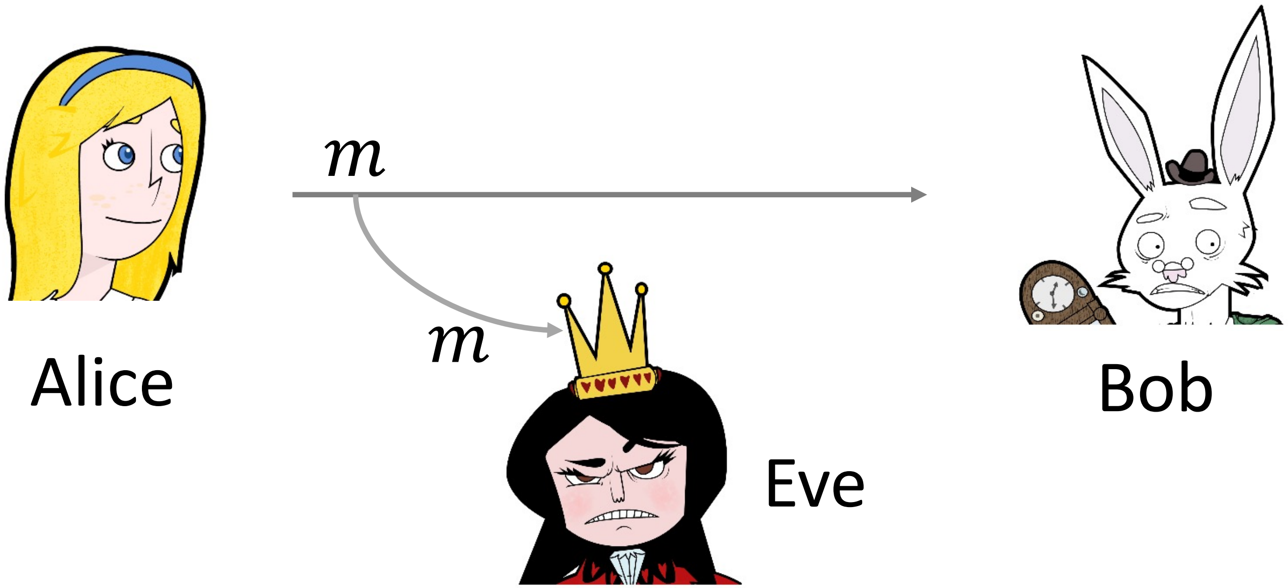
Provable Security

- In the past: The **ancient art** of secure communication
  - Examples: Caesar cipher, ENIGMA, one-time pad
- Today: A **real science**
  - Thanks to pioneers such as Silvio Micali, Shafi Goldwasser, Oded Goldreich
  - Formal **definitions** and security **proofs**
- We will give a very **high-level** overview
  - Focus on applications

CHAPTER 1: Symmetric Cryptography

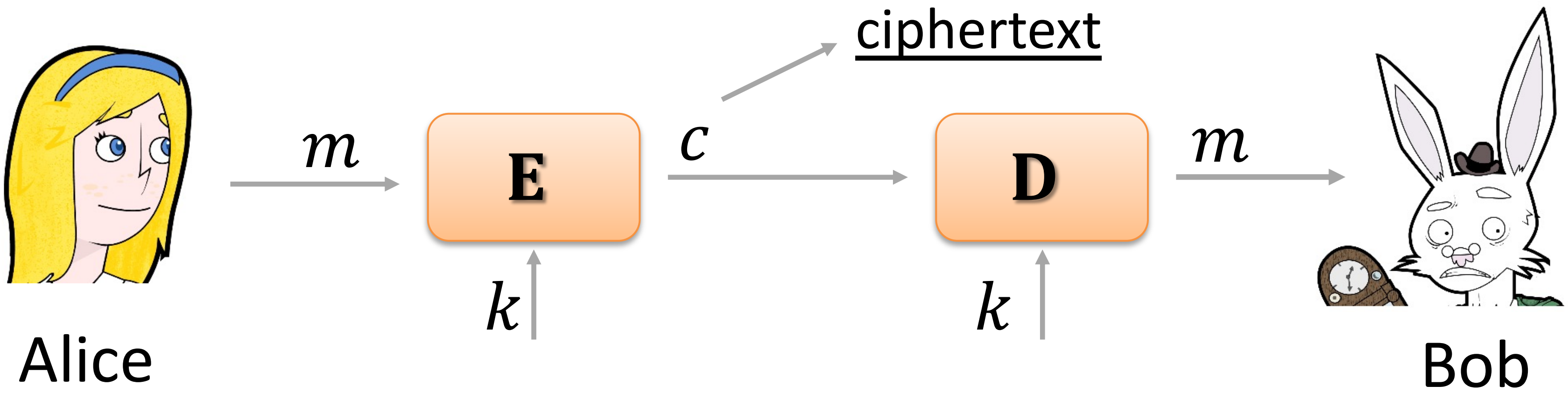
Confidential Communication

- Alice wants to send a message to Bob over some communication channel
- Eve can **listen** to the channel
- How to protect the **message content**?



Secret-Key Encryption

- Assume Alice and Bob **share** a secret key



- **Correctness:** $\mathbf{D}\big(k, \mathbf{E}(k, m)\big) = m$
- **Kerckhoffs principle:** Security only based on the secrecy of the key (**algorithms are public**)

Perfect Secrecy

- Definition due to Claude Shannon (1949)
  - Ciphertext **reveals nothing** about the plaintext

$$\Pr[M = m] = \Pr[M = m | C = c]$$

- One-time pad (binary version):
  - $\mathbf{E}(k, m) = k \oplus m$
  - $\mathbf{D}(k, c) = k \oplus c$
- Limitations:
  - **One key** per message, and message **as long as key**
  - Can be shown to be **inherent**

Computational Security

- Previous definition is information-theoretic
  - Holds even for **all-powerful** adversaries
  - Unconditional security, i.e. **no assumptions**!
- Natural relaxation: **Computational security**
  - Computationally bounded adversary (PPT Turing machine)
  - Adversary has negligible probability of success (e.g. 2^{-80})
- Advantage: **Single short key** for encrypting an **unbounded** number of messages

AES (Rijndael)

- A widely used **blockcipher**
  - Created to replace DES
- NIST call for proposals in 1997
  - Evaluation criteria: Security, costs, intellectual property, implementation and execution, versatility, key agility, simplicity
- Two rounds were performed, 15 algorithms were selected in the first and 5 in the second
  - NIST completed the evaluation on October 2, 2000 and selected **Rijndael (Daemen + Rijmen)**

AES Structure

- Block length of 128 bits (16 bytes)
  - Three key sizes: 128, 192, or 256 bits
- # of rounds: 10, 12, or 14
  - Let $s_{i,j}^{(\mathrm{in})}$ be 1 byte of the **state** at a given round (initially the first plaintext block)
  - The secret key is used to compute the **sub-keys** $k_{i,j}^{(r)}$, one for each round r
  - In each round state subject to 4 operations: **SubBytes**, **ShiftRows**, **MixColumns**, **AddRoundKey**

Arithmetic in $GF(2^8)$

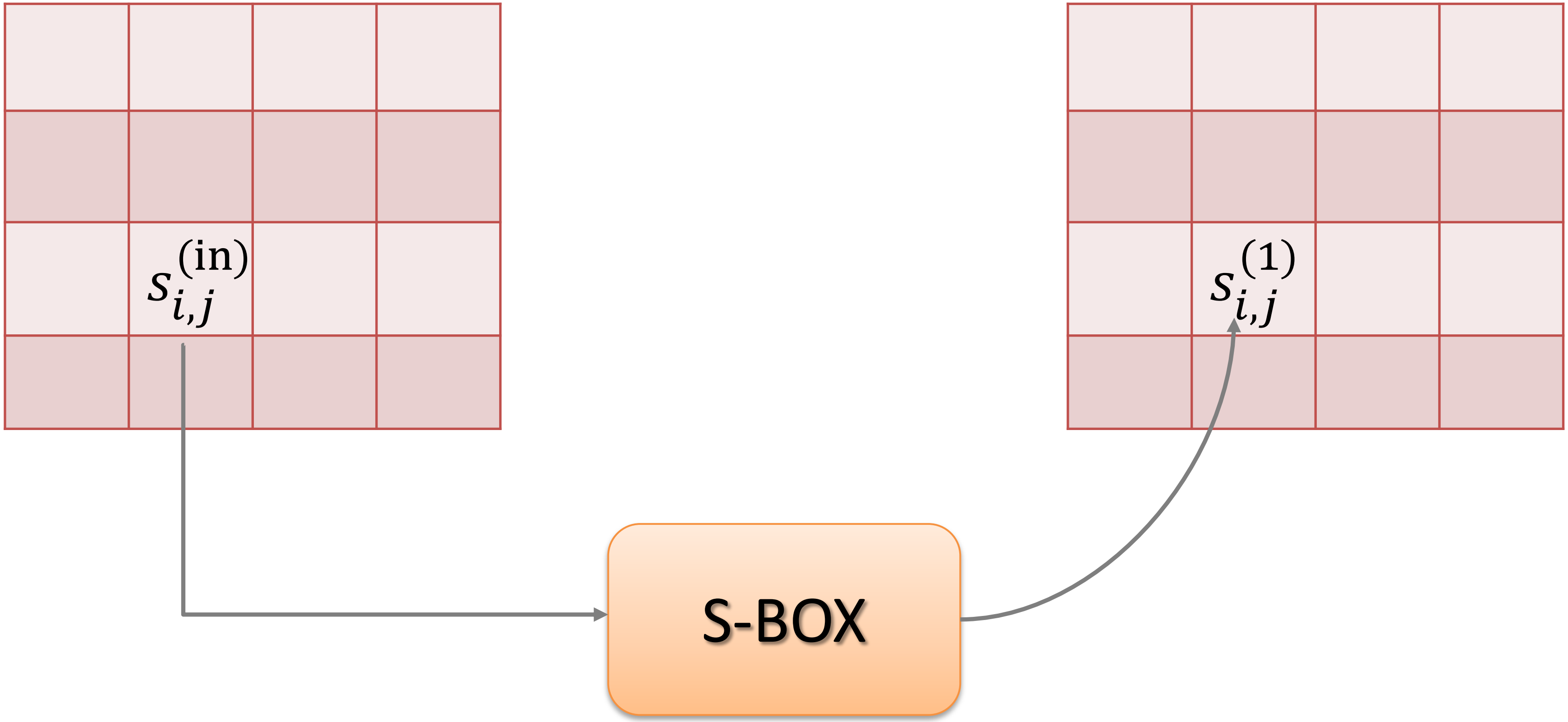
- AES uses the Galois Field $GF(2^8)$
  - 1 byte ⇒ 8 bits ⇒ 2 hexadecimal digits
  - Example: $[01101100]_2 = [6C]_{16}$
- Interpret each byte as the binary coefficients of a degree-7 polynomial
  - Sum of 2 polynomials is still a degree-7 polynomial
  - Multiplication might increase the degree
  - Modular reduction w.r.t. **irreducible polynomial**

$$h(X) = X^8 + X^4 + X^3 + X + 1$$

Arithmetic in $GF(2^8)$: Example

- $[53]_{16} \cdot [CA]_{16} = [01]_{16}$ in $GF(2^8)$
  - $[53]_{16} = X^6 + X^4 + X + 1$
  - $[CA]_{16} = X^7 + X^6 + X^3 + X$
- $[53]_{16} \cdot [CA]_{16} = X^{13} + X^{12} + X^{11} + X^{10} + X^9 + X^8 + X^6 + X^5 + X^4 + X^3 + X^2 + X$
  - By performing **long division**, it is easy to check that $X^{13} + X^{12} + X^{11} + X^{10} + X^9 + X^8 + X^6 + X^5 + X^4 + X^3 + X^2 + X \bmod X^8 + X^4 + X^3 + X + 1$ gives $X^5 + X^4 + X^3 + X + 1$ with **remainder** 1

SubBytes



AES S-BOX (1/2)

- A simple **substitution box** (lookup table)
- It maps 8-bit inputs to 8-bit outputs
  - Let $x = [x_7, x_6, x_5, x_4, x_3, x_2, x_1, x_0]_2$ be the input
  - Map x into its **multiplicative inverse** z modulo $h(X)$ and apply an **affine transformation**:

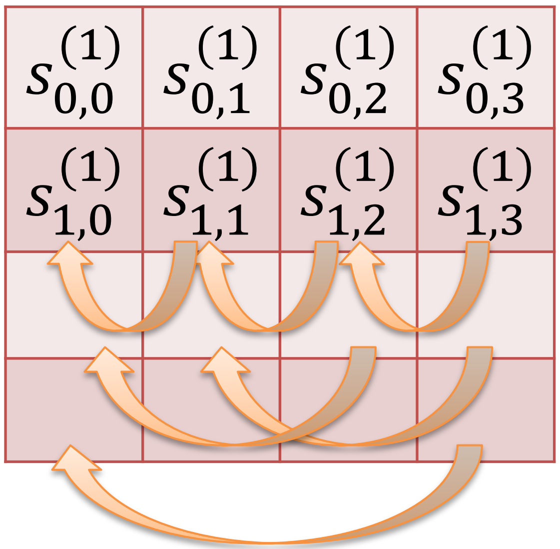
$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \\ z_6 \\ z_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

AES S-BOX (2/2)

- Input: [68]
- Output: [45]

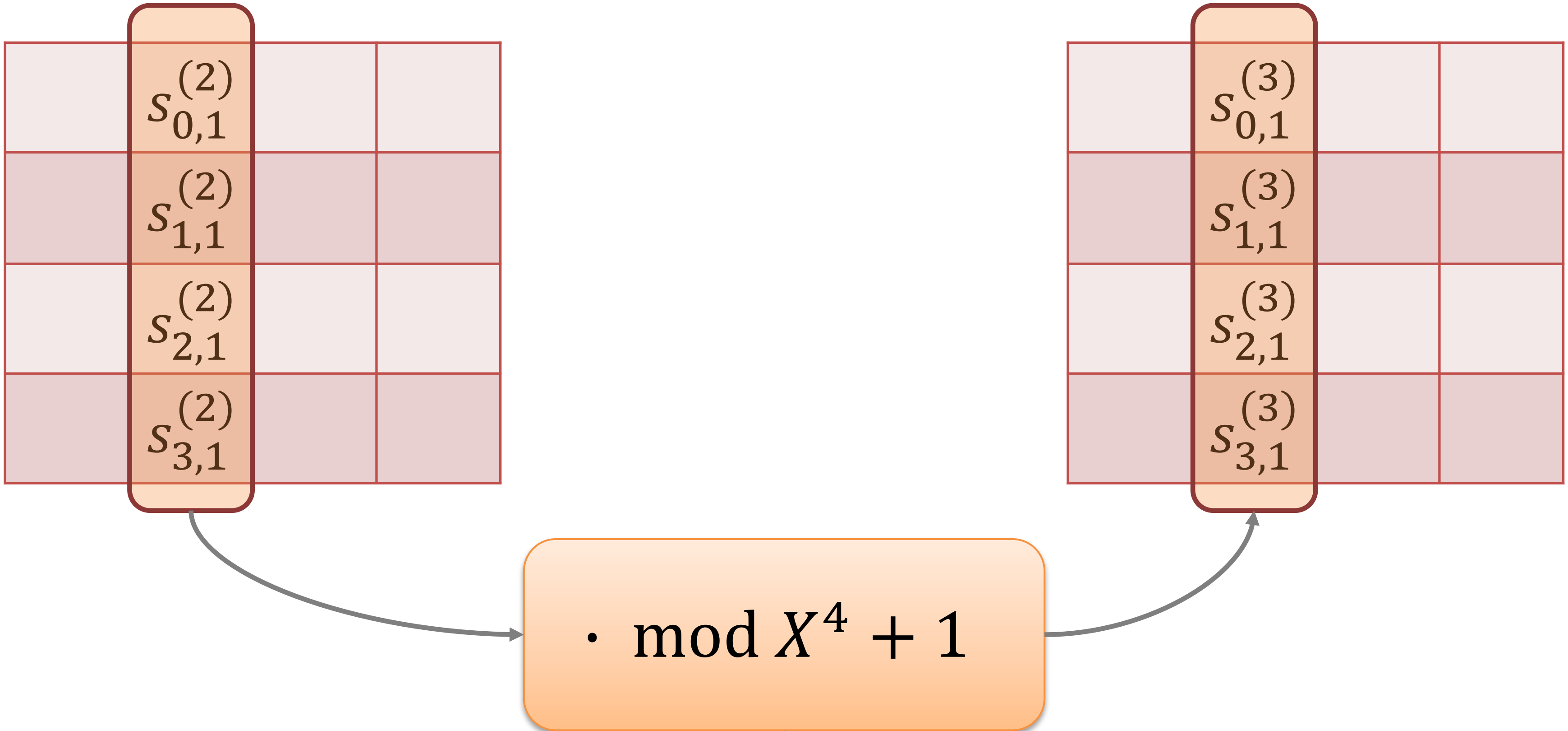
		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

ShiftRows



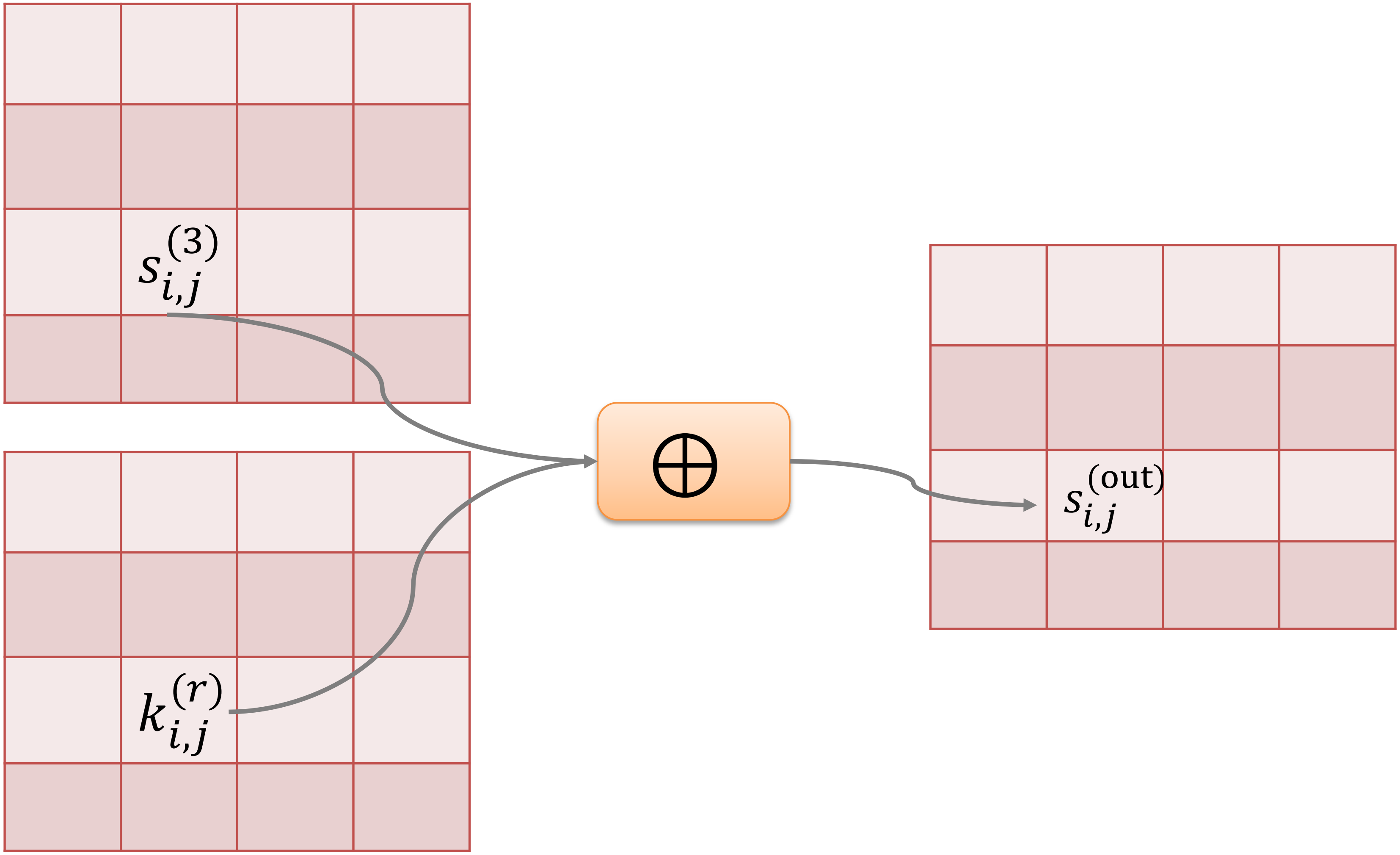
$$\begin{bmatrix} s_{0,j}^{(2)} \\ s_{1,j}^{(2)} \\ s_{2,j}^{(2)} \\ s_{3,j}^{(2)} \end{bmatrix} = \begin{bmatrix} s_{0,j}^{(1)} \\ s_{1,j-1}^{(1)} \\ s_{2,j-2}^{(1)} \\ s_{3,j-3}^{(1)} \end{bmatrix}$$

MixColumns



$$[03]X^3 + [01]X^2 + [01]X + [02]$$

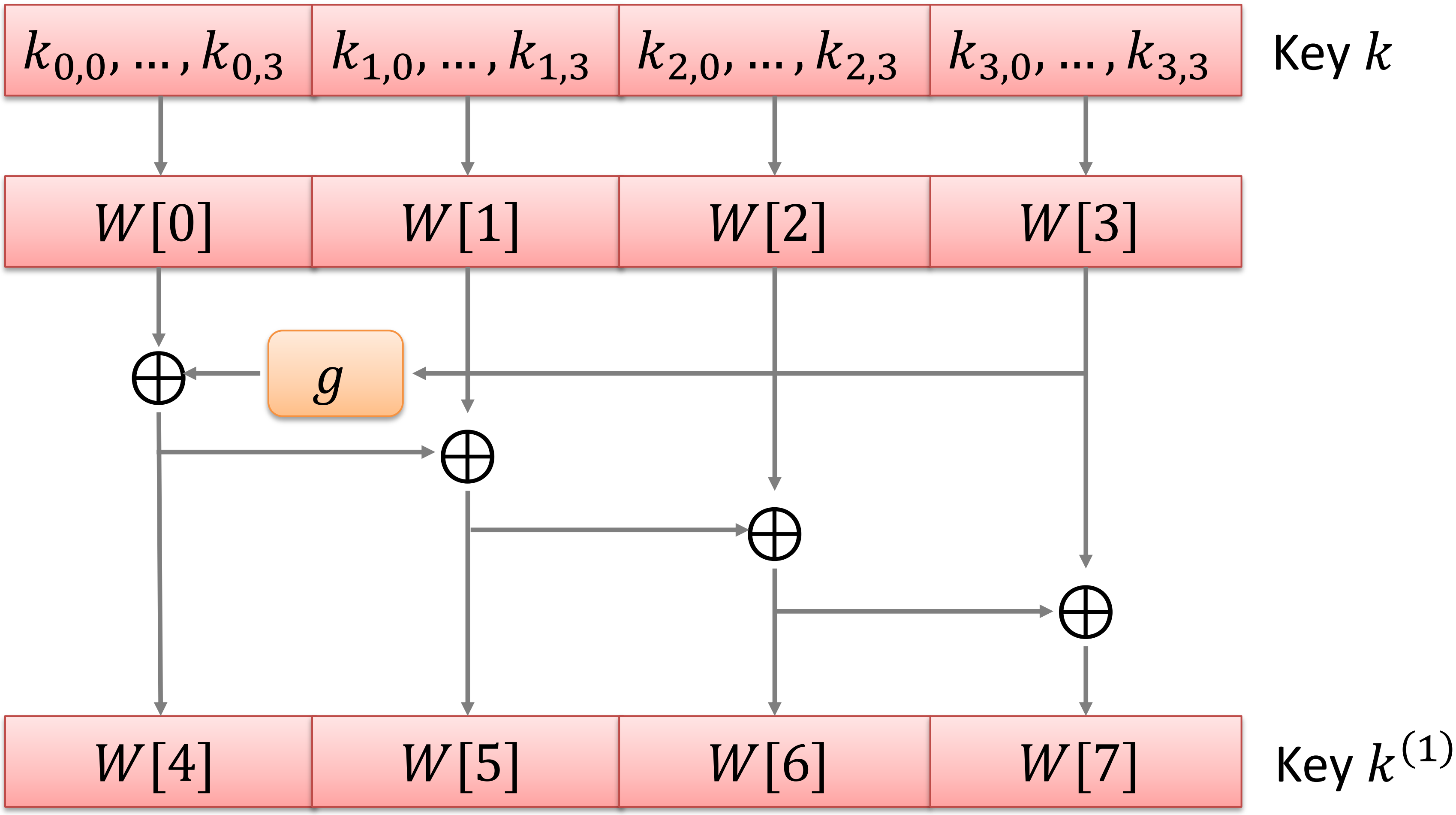
AddRoundKey



Key Schedule (1/2)

- Takes the original key k (128, 192, or 256 bits) and **derives sub-keys** $k^{(r)}$, for $r = 10,12,14$
- When $|k| = 128$ and $r = 10$:
  - Key expansion array W with 44 32-bit elements
  - $W[0], \ldots, W[3]$ equal to the original key (used for initial XOR with the plaintext – key whitening)
  - $W[4i] = W[4(i-1)] + g(W[4i-1])$
  - $W[4i+j] = W[4i+j-1] + W[4(i-1)+j]$
  - g is a non-linear function (based on the S-Box)

Key Schedule (2/2)



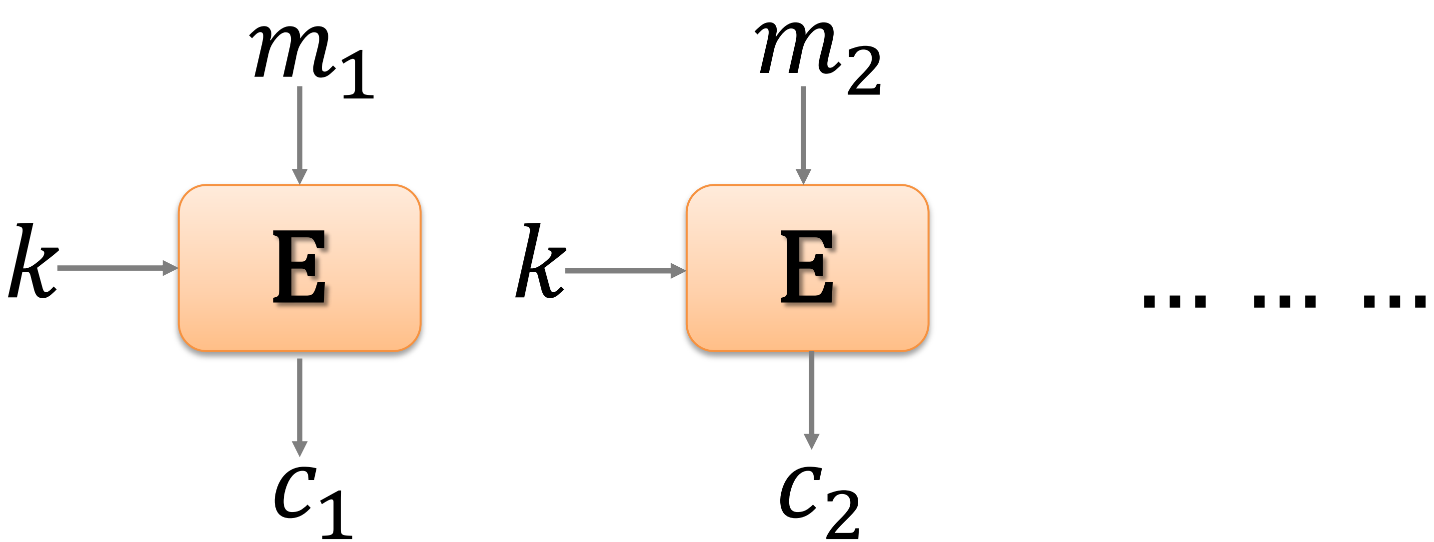
Security of AES

- No **practical attack** is known
  - Best attacks break AES with 7 rounds (128-bit key), 8 rounds (192-bit key) and 9 rounds (256-bit key)
- Brute force is out of reach: $3{,}4 \cdot 10^{38}$ possible combinations (128-bit key)
  - Best brute force attack took 5 years to crack a 64-bit key using thousands of CPUs
- But AES comes with **no proof of security**!

Modes of Operation

- Block ciphers encrypt **fixed size blocks**
  - E.g., AES block size is 128 bits
- Plaintext messages may have an **arbitrary length**: Use a **mode of operation**
  - Segment data & encrypt and chain multiple blocks
  - Might require to pad messages to make their length a multiple of the block size
- 4 modes defined for DES in ANSI standard "ANSI X3.106-1983 Modes of Use"

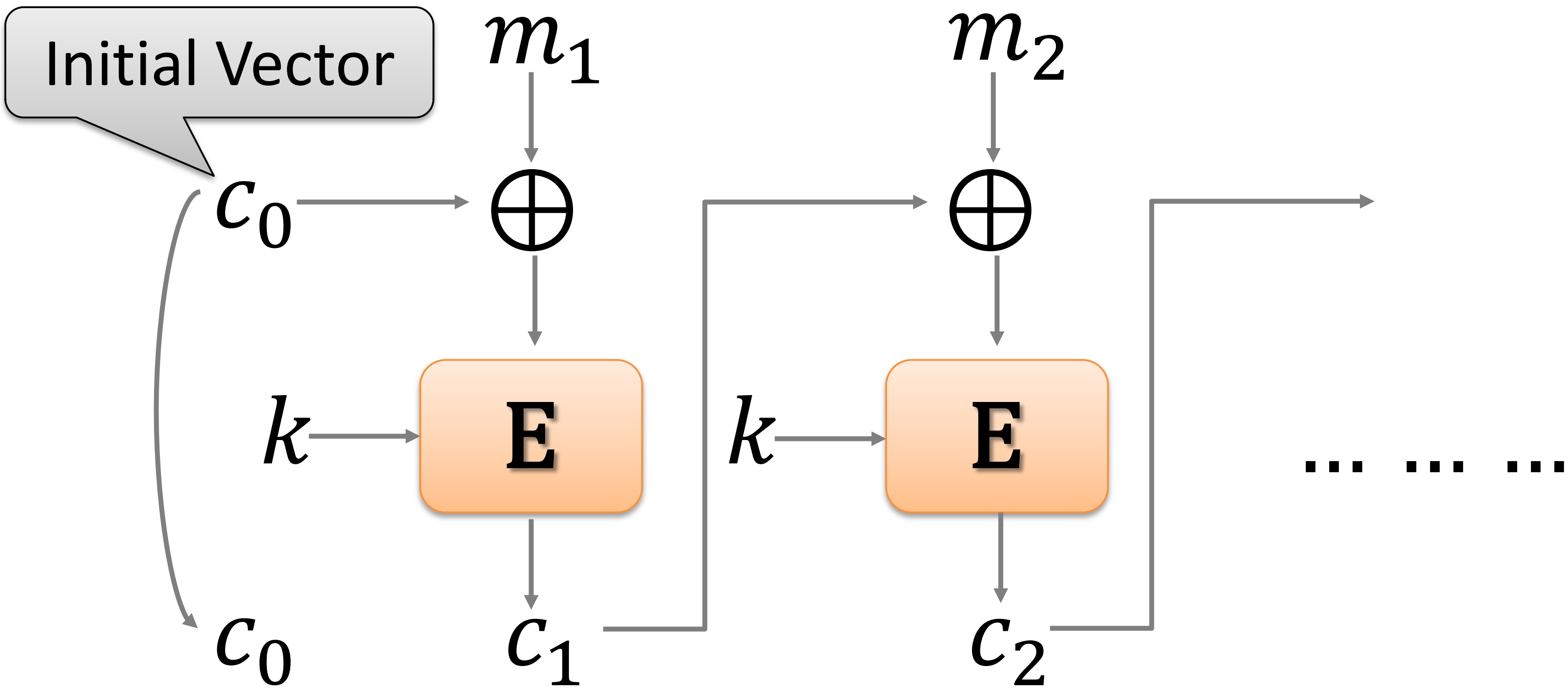
ECB Mode



$$\forall i\colon\ c_i = \mathbf{E}(k, m_i)$$

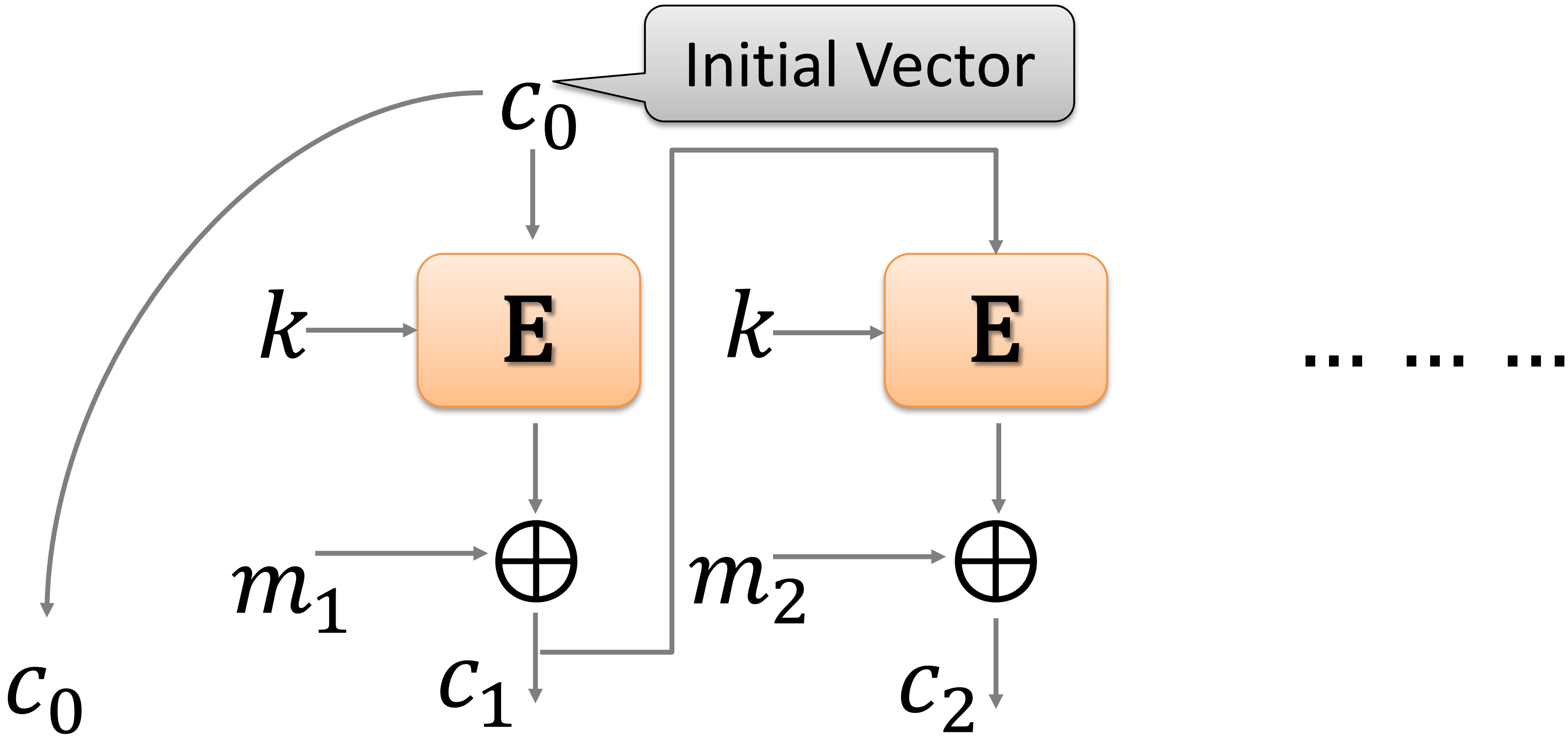
CIS SAPIENZA
RESEARCH CENTER FOR CYBER INTELLIGENCE AND INFORMATION SECURITY

CBC Mode



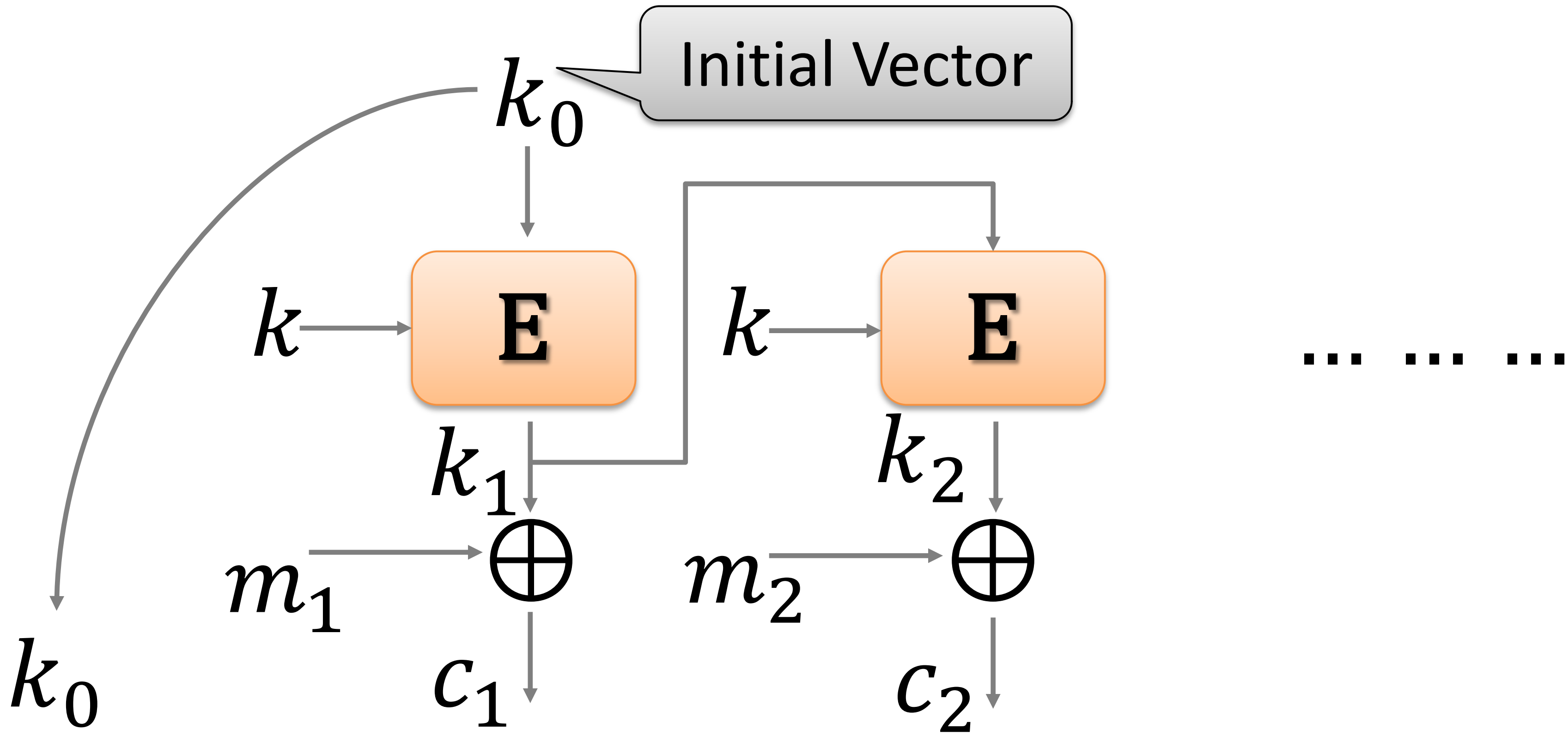
$$\forall i:\ c_i = \mathbf{E}(k, c_{i-1} \oplus m_i)$$

CFB Mode



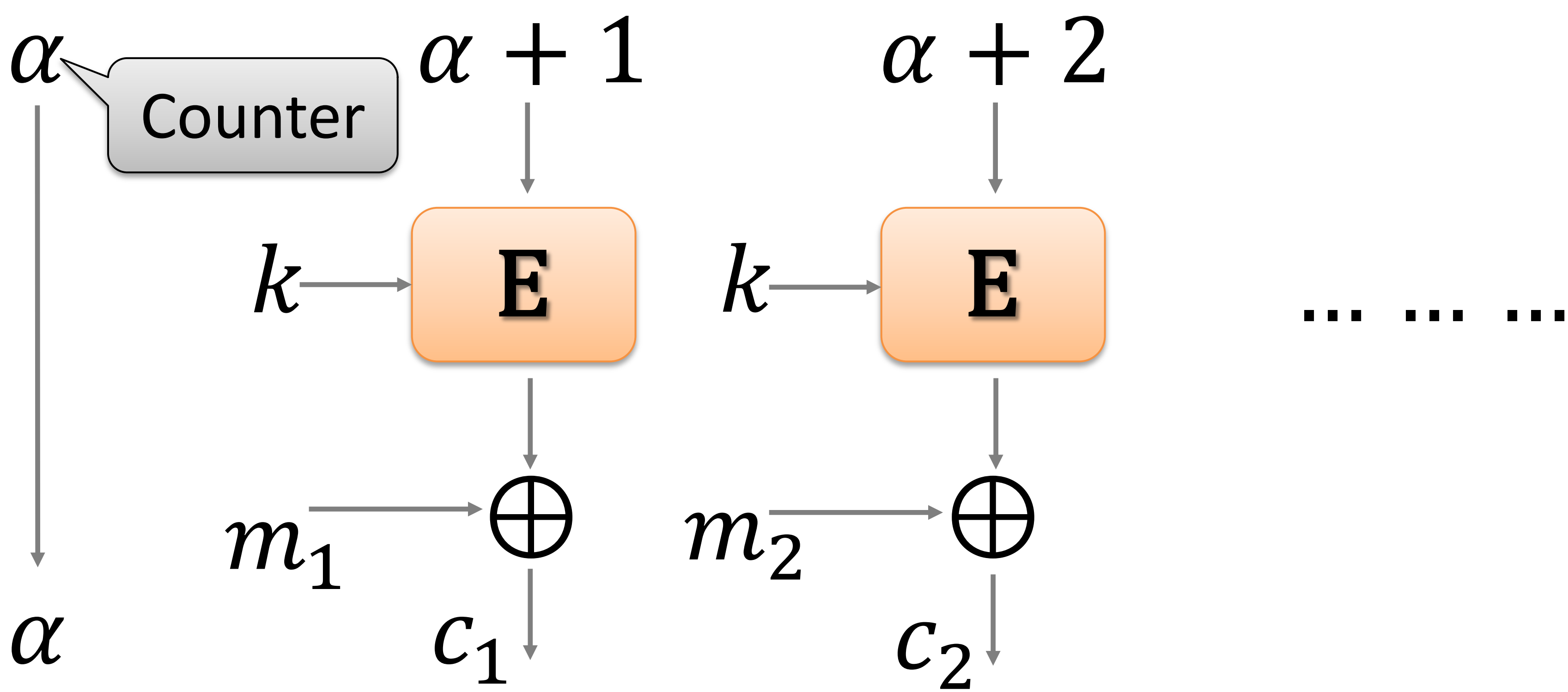
$$\forall i:\ c_i = m_i \oplus \mathbf{E}(k, c_{i-1})$$

OFB Mode



$$\forall i\colon\ c_i = m_i \oplus \mathbf{E}(k, k_{i-1})$$

CTR Mode



$$\forall i: \ c_i = m_i \oplus \mathbf{E}(k, (\alpha + i) \bmod 2^n)$$

Comparison

- ECB: Identical plaintext blocks are encrypted into **identical** ciphertext blocks

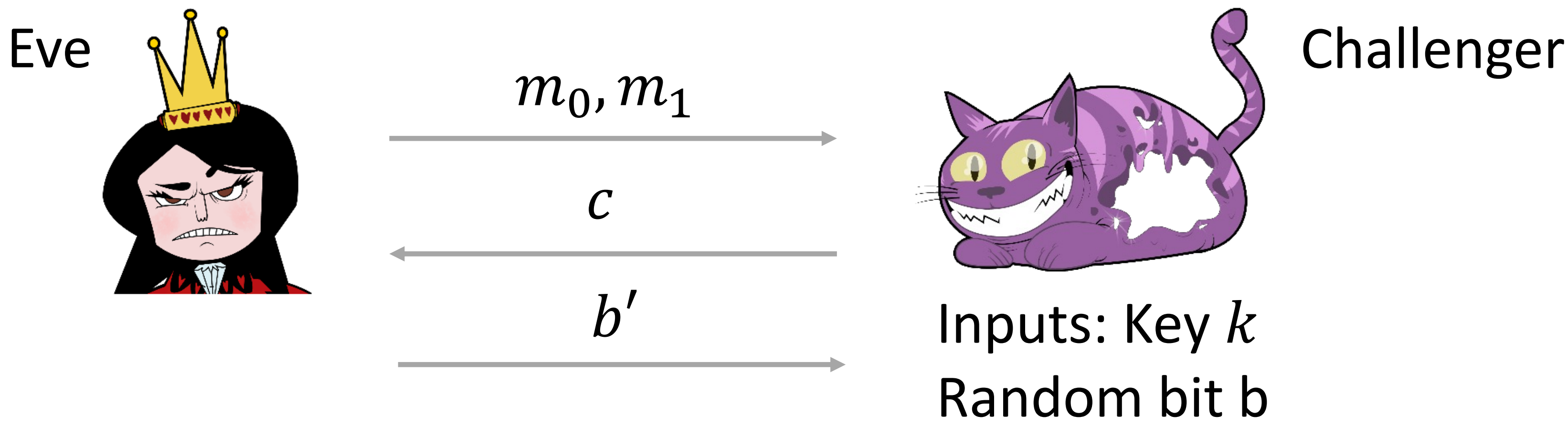
Mode	‖Enc	‖Dec	$Access	Security
ECB	✓	✓	✓	✗
CBC	✗	✓	✓	✓
CFB	✗	✓	✓	✓
OFB	✗	✗	✗	✓
CTR	✓	✓	✓	✓

Security of Block Ciphers

- Rule of thumb (Shannon): A good block cipher should have both **confusion** and **diffusion**
  - Confusion means there is a **complex relation** between ciphertext and plaintext
  - Diffusion roughly means that a **one-bit** flip in the plaintext changes **each bit** of the ciphertext with probability $\approx 1/2$
- But can we define more **formally** what it means for a cipher to be secure?

One-Time Security

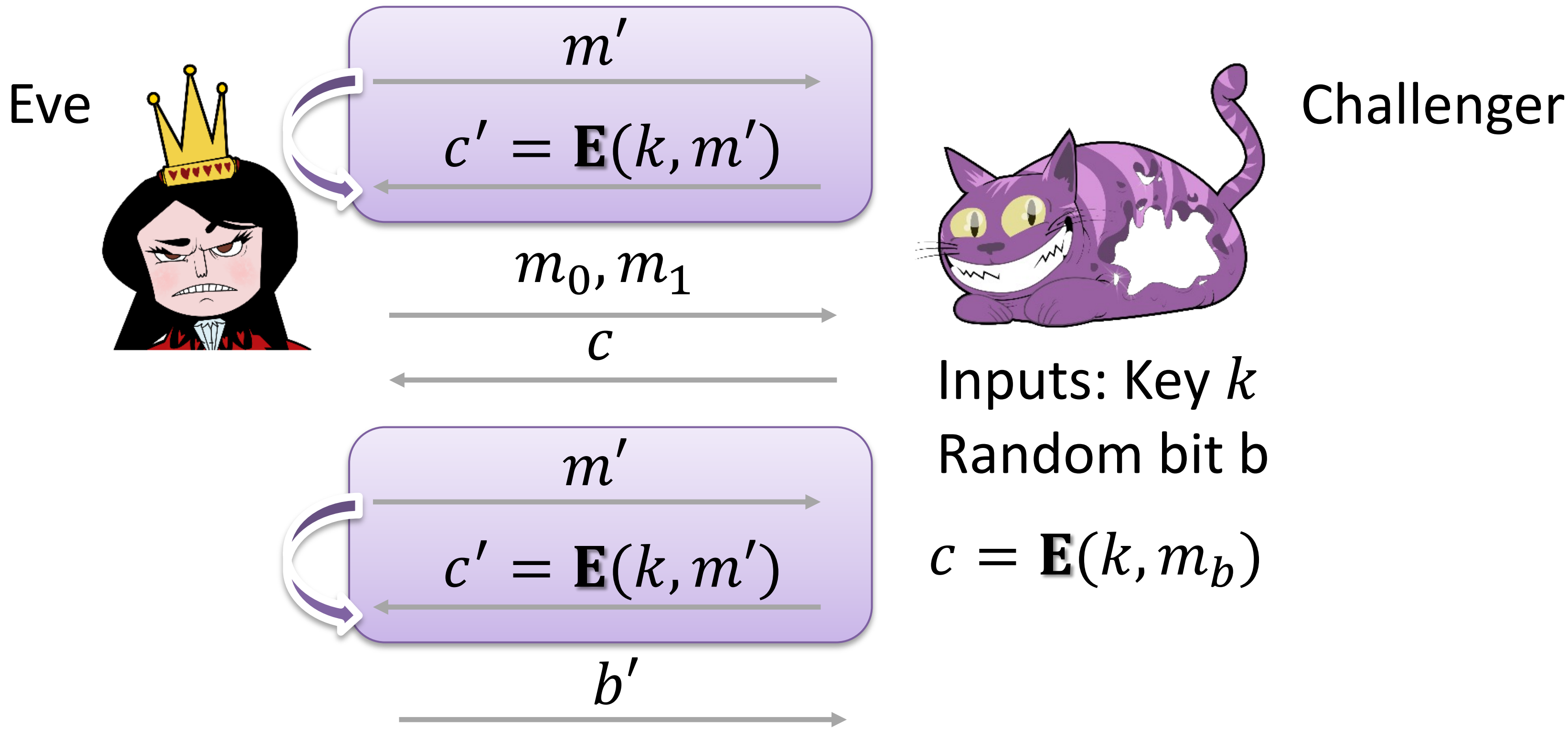
- The **indistinguishability paradigm**



$$c = \mathbf{E}(k, m_b)$$

  - Hard to **guess** b w.p. better than 1/2
  - No encryption/decryption capabilities

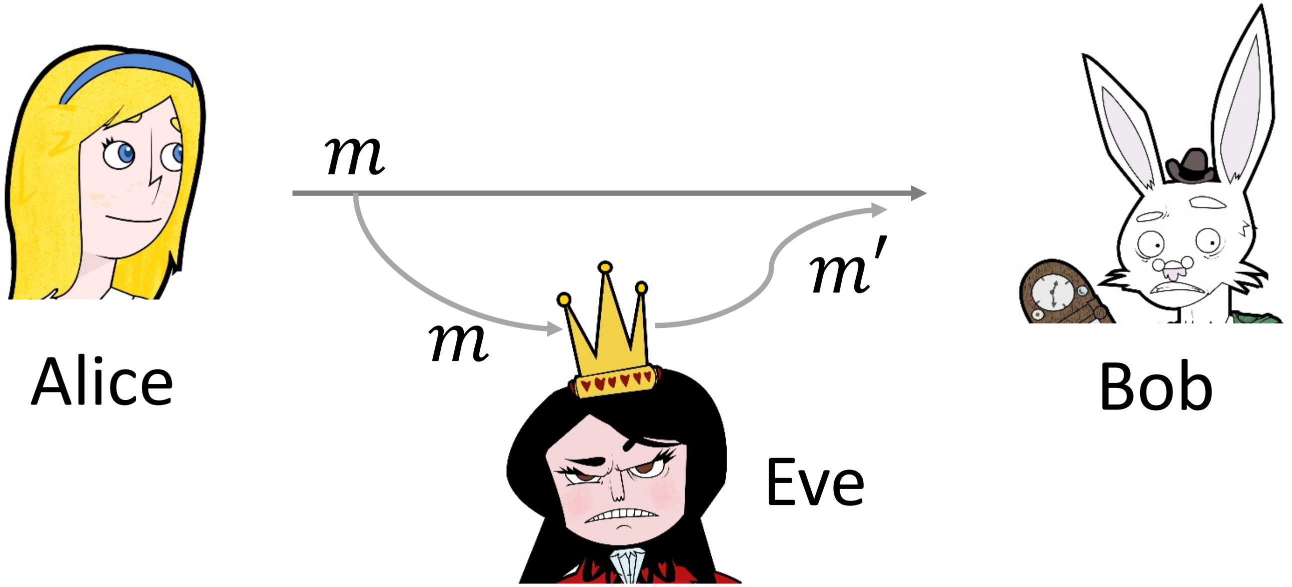
Chosen-Plaintext Attacks (CPA) Security



- Adversary can ask **encryption** queries
- Requires **randomness**!

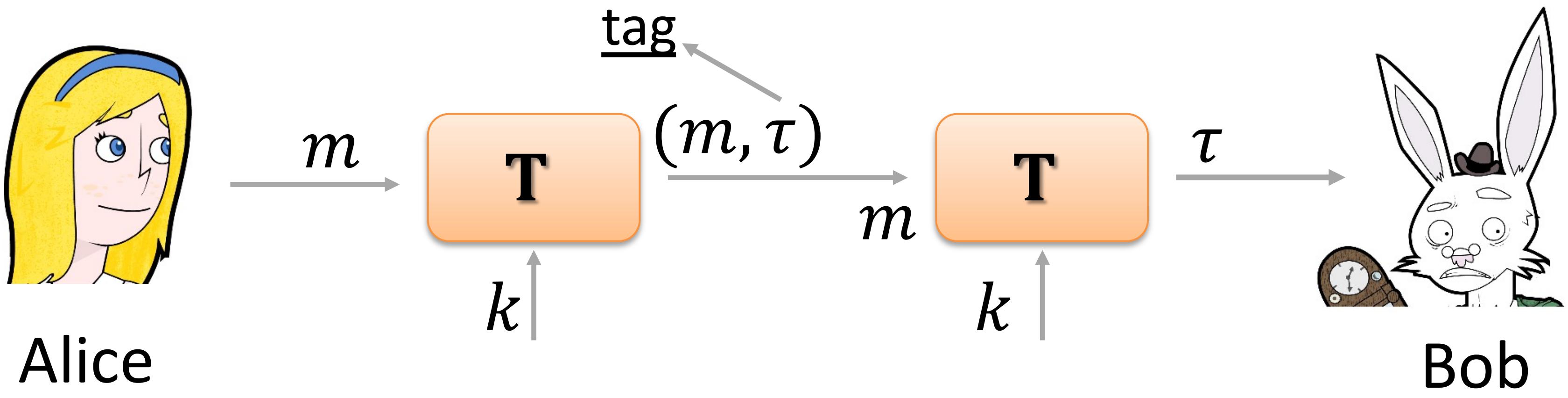
Authenticated Communication

- Alice wants to send a message to Bob over some communication channel
- Eve can **modify** the message
- How to protect the message **authenticity**?



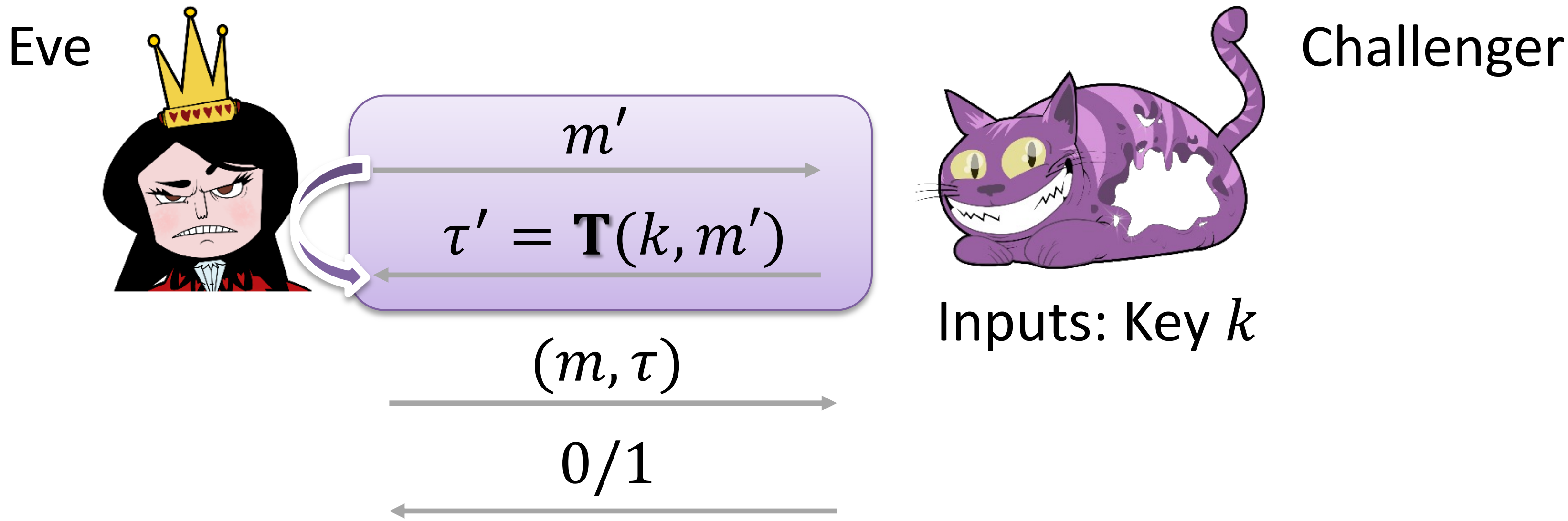
Message Authentication Codes

- Assume Alice and Bob **share** a secret key



- **Correctness:** By definition
- **Security:** Should be hard to **forge** a tag on a message without knowing the key

Unforgeability



- Adversary wins iff (m, τ) is **valid** and m is **fresh** (i.e. not asked during tag queries)
  - Reply attacks not covered by definition

CBC-MAC

- Use AES in CBC mode
- Fix $IV = 0^n$ and output **only** the last block
  - I.e., for $m = (m_1, \dots, m_t)$ where $m_i \in \{0,1\}^n$ compute $\tau_i = \mathbf{F}(k, \tau_{i-1} \oplus m_i)$, where $\tau_0 = IV$, and return $\tau = \tau_t$
  - Only secure for **fixed length** messages, for variable length messages need to encrypt the output with an indepedent key (i.e. $\tau' = \mathbf{F}(k', \tau)$)
  - Insecure in case **all blocks** are output

Why Fixed Length?

- Suppose we use CBC-MAC to autenticate **variable length** messages
- Adversary picks arbitrary $m_1, m_2 \in \{0,1\}^n$ and obtains tags on m_1 and $m_2 \oplus \tau_1$

$$\tau_1 = \mathbf{F}(k, m_1); \tau_2 = \mathbf{F}(k, m_2 \oplus \tau_1)$$

- Output forgery $m^* = m_1 || m_2$ and $\tau^* = \tau_2$

$$\tau_2 = \mathbf{F}(k, m_2 \oplus \tau_1) = \mathbf{F}(k, \mathbf{F}(k, m_1) \oplus m_2))$$

Why Only the Last Block?

- Suppose CBC-MAC outputs **all blocks**
- Adversary picks arbitrary $m_1, m_2 \in \{0,1\}^n$ and obtains tag $\tau_1 || \tau_2$ on $m_1 || m_2$

$$\tau_1 = \mathbf{F}(k, m_1); \tau_2 = \mathbf{F}(k, m_2 \oplus \tau_1)$$

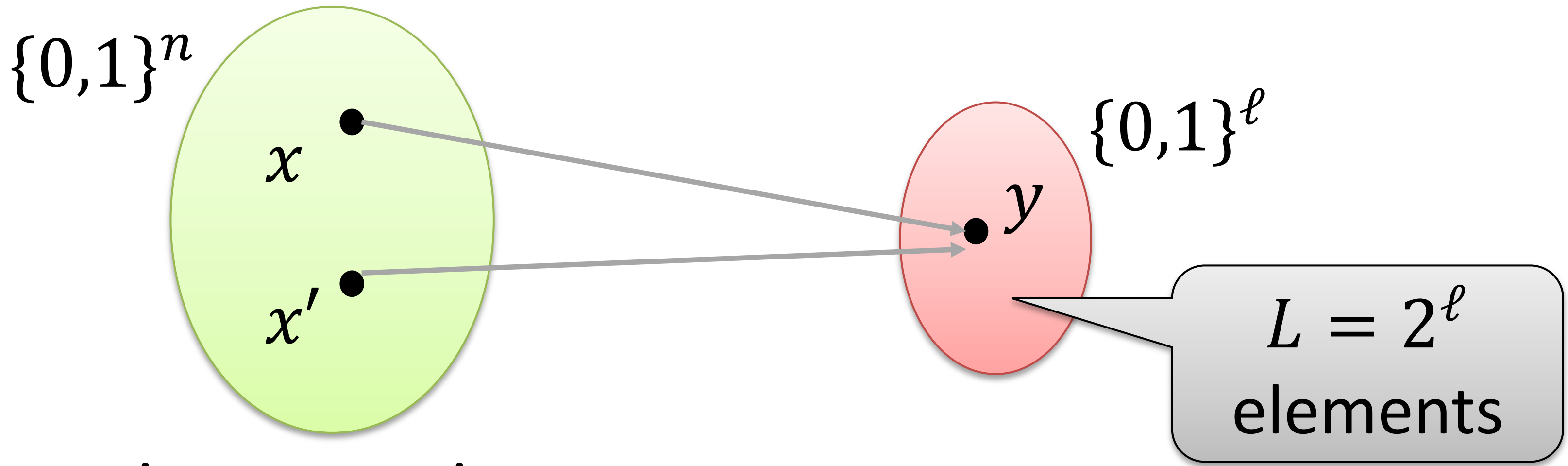
- Output forgery $m^* = \tau_1 \oplus m_2 || \tau_2 \oplus m_1$ and $\tau^* = \tau_2 || \tau_1$

$$\mathbf{F}(k, \mathbf{F}(k, m_2 \oplus \tau_1) \oplus \tau_2 \oplus m_1) = \mathbf{F}(k, m_1)$$

Why not a Random IV?

- Suppose that for each tag we sample **random** $\tau_0 \leftarrow_\$ \{0,1\}^n$ and output (τ_0, τ_t) as tag
  - Here, t is the number of n-bit blocks in a message
- Adversary picks arbitrary $m \in \{0,1\}^n$ and obtains tag (τ_0, τ_1) where $\tau_1 = \mathbf{F}(k, \tau_0 \oplus m)$
- Output forgery $m^* = \tau_0$ and $\tau^* = (m, \tau_1)$

Cryptographic Hashing



- Security properties:
  - **One wayness:** Given y, find x such that $\mathbf{H}(x) = y$
  - **Weak collision resistance:** Given x, find $x' \neq x$ s.t. $\mathbf{H}(x) = \mathbf{H}(x')$
  - **Strong collision resistance:** Find x and x' s.t. $\mathbf{H}(x) = \mathbf{H}(x')$ but $x \neq x'$

Brute Force Attacks

- Assume $\mathbf{H}$ to be a **random hash function**
- **One wayness:** Given y choose $x_1, \ldots, x_q$ and hope that $\mathbf{H}(x_i) = y$ for some $i \in [q]$
  - Success probability: $\leq q/L$ (union bound)
- **Weak collision resistance:** Similar to above
- **Strong collision resistance:** Choose distinct $x_1, \ldots, x_q$ and hope to find a collision

$$\Pr[\exists i \neq j : y_i = y_j] \leq \sum_{i \neq j} \Pr[y_i = y_j] \leq \frac{q^2}{2L}$$

CIS SAPIENZA
Research Center for Cyber Intelligence and Information Security

The Birthday Paradox

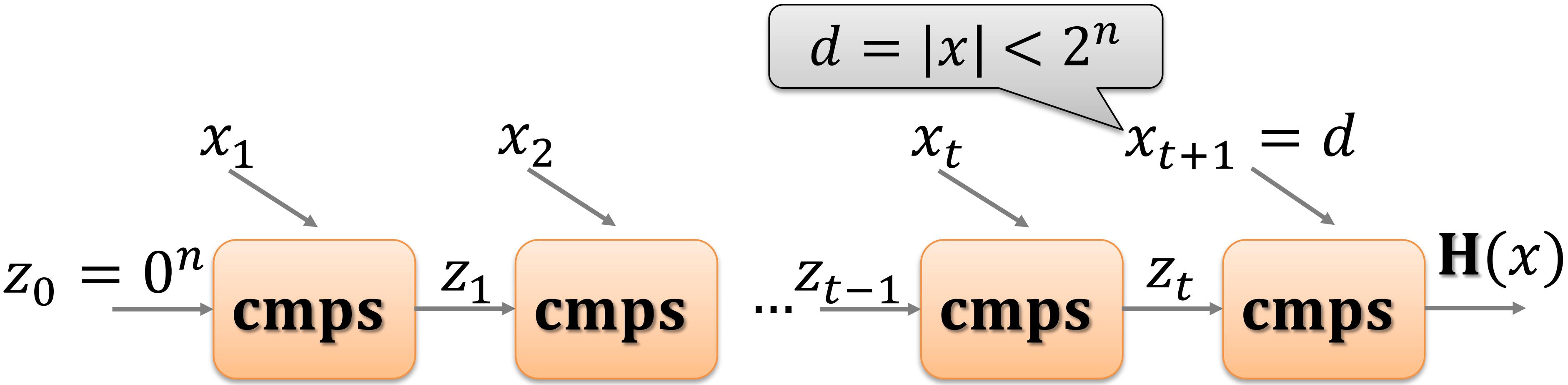
- Suppose $y_1, \ldots, y_q$ are **random**
  - Let $NoColl_i$ be the event that **no collision** occurs within $y_1, \ldots, y_i$
  - $\Pr[NoColl_{i+1} | NoColl_i] = (1 - i/L)$

$$\Pr\left[NoColl_q\right] = \prod_{i=1}^{q-1}\left(1 - \frac{i}{L}\right) \leq \prod_{i=1}^{q-1} e^{-\frac{i}{L}} = e^{-\sum_{i=1}^{q-1}\frac{i}{L}}$$
$$= e^{-q(q-1)/2L}$$

- Thus, $1 - \Pr[NoColl_q] \geq \frac{q(q-1)}{4L}$
  - Success w.p. $\geq 1/2$ whenever $q \approx \sqrt{L}$

Merkle-Damgaard

- Let **cmps** be a **compression function** outputting ℓ' bits out of ℓ bits
  - **cmps** is collision resistant, but domain is **fixed**
- A construction due to Merkle and Damgaard yields a collision resistant hash function for **arbitrary domains**



Davies-Meyer

- Compression functions can be constructed from **block ciphers**

$$\mathbf{cmps}(x_1, x_2) = x_2 \oplus \mathbf{AES}(x_1, x_2)$$

- Analysis requires to assume idealization of AES
  - Because the input is used **as the key**
  - Ideal cipher: Block-cipher as a **random permutation** for **every choice** of the key

Hash & MAC

- Typical (but **flawed**) construction of a MAC based on a hash function

$$\mathbf{T}(k, m) = \mathbf{H}(k||m)$$

- Attack based on **length extension** (for Merkle-Damgaard-based constructions)
  - Let $m^* = m||d||m_{t+1}$ and tag $\tau^* = \mathbf{cmps}(\mathbf{cmps}(\tau||m_{t+1})||d + 2)$ for $\tau = \mathbf{H}(k||m)$ and $d = |m|$

HMAC

- Solution: Hash **twice**!

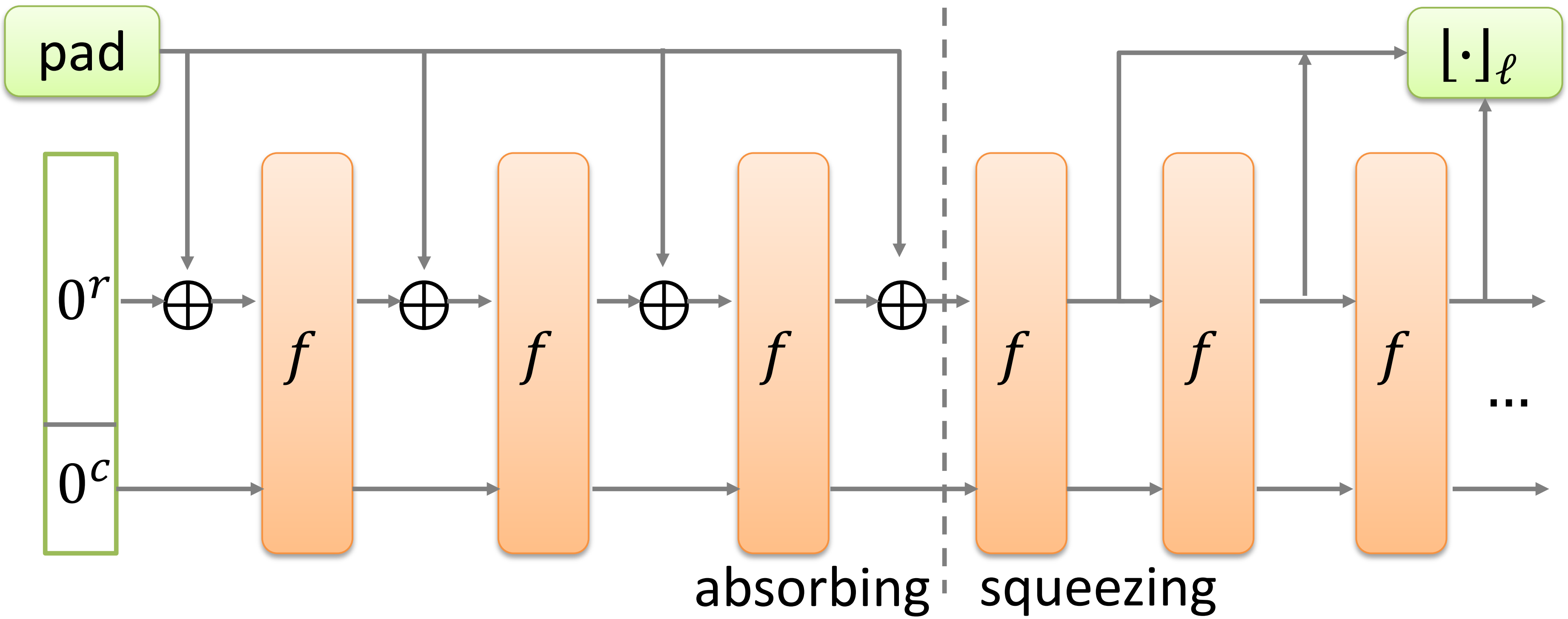
$$\mathbf{HMAC}(k, m) = \mathbf{H}\left(k^{+} \oplus opad || \mathbf{H}(k^{+} \oplus ipad || m)\right)$$

  - k^{+}: Key k padded with zeroes to the left
  - $opad$: $5C5C \ldots 5C$ (in HEX)
  - $ipad$: $3636 \ldots 36$ (in HEX)
- Internet **standard** RFC 2104
- Can work with any of SHA-2 or SHA-3

SHA-3

- 2005-2006: NIST thinks about SHA-3 contest
  - MD5 and SHA-1 were damaged by attacks
  - SHA-2 based on the same principles
- October 2008: Deadline for proposals
  - More efficient than SHA-2
  - Output lengths: 224, 256, 384, 512 bits
  - Security: collision resistant (weak and strong)
- October 2, 2012: NIST announces **Keccak** as SHA-3 winner

The Sponge Construction



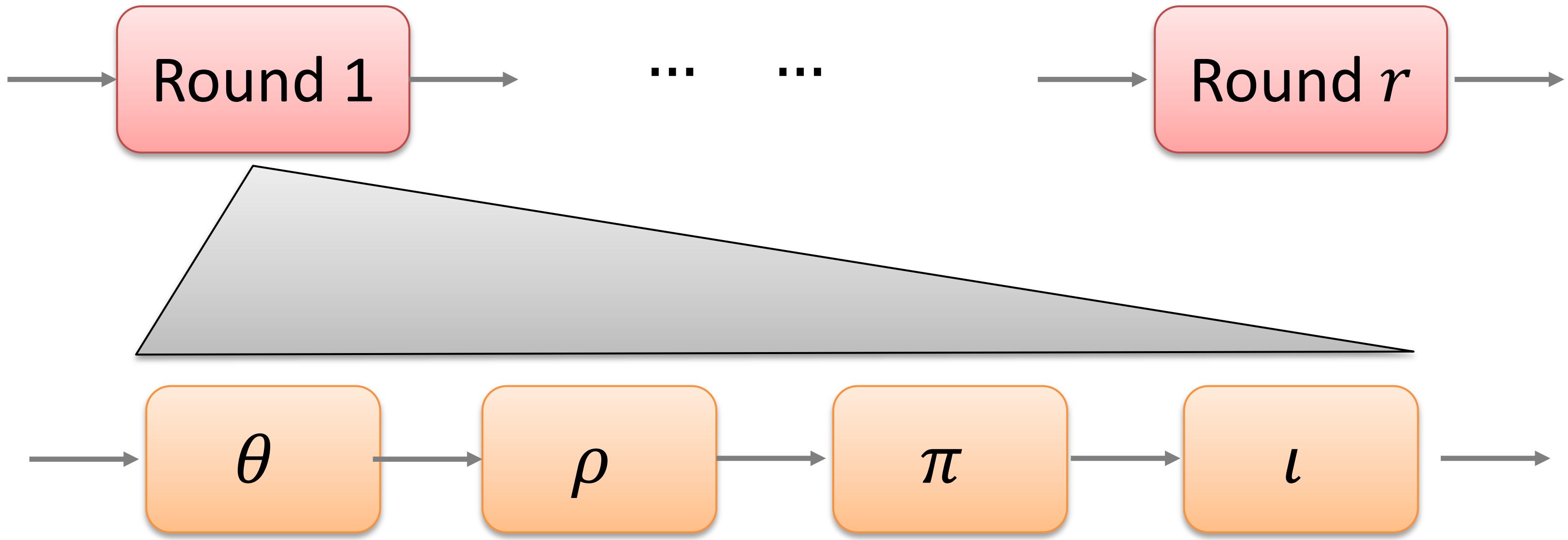
- Can be used as a stream cipher, or a MAC too
- Security for **ideal** f is roughly $q(q-1)/2^{c+1}$
  - q = # of calls to f

CIS SAPIENZA
RESEARCH CENTER FOR CYBER INTELLIGENCE AND INFORMATION SECURITY

Inside Keccak

- **Absorbing:** The message blocks are padded and processed
- **Squeezing:** An output of configurable length is produced
- Parameters:
  - $b = r + c$ it's the **state width**, with $b = 25 \cdot 2^l$ for values $l = 0,1, \dots, 6$
  - r is the **bit rate** (length of single blocks)
  - c is the **capacity** (security parameter)
  - SHA-3: Always $b = 1600$

The Keccak f-Permutation

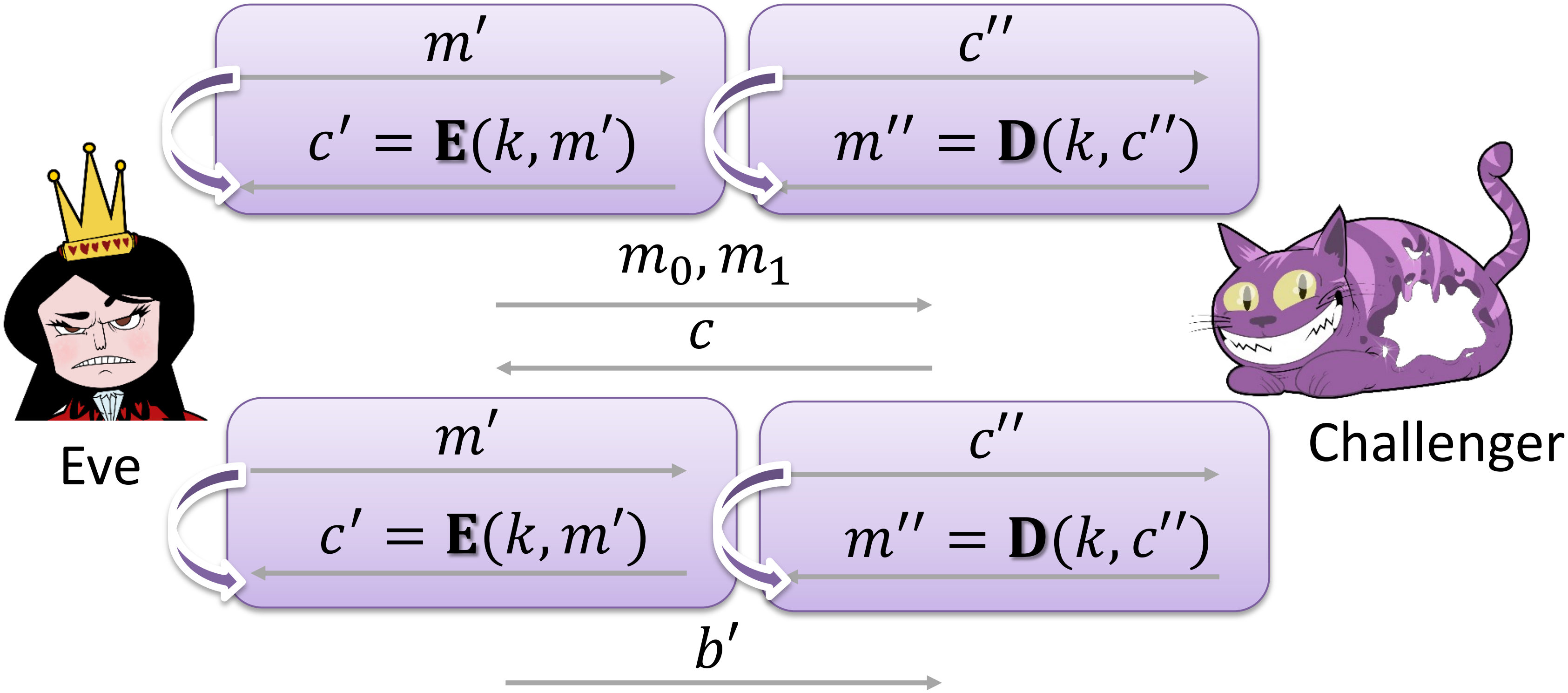


- A **permutation** over b bits
- Variable number of rounds $r = 12 + 2l$
  - SHA-3: $l = 6$ and thus $r = 24$
- The functions θ, ρ, π, ι use XOR, AND, and NOT

Combining Encryption and Authentication

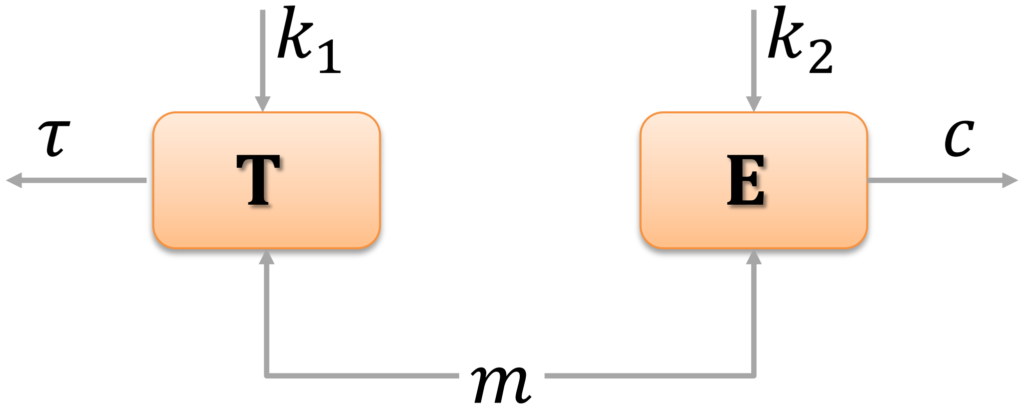
- Eand authentication **separate goals**
- Can we achieve **both** at the same time?
- Intuitively we want that both
  - The ciphertext should hide the plaintext
  - It should be hard to compute a ciphertext without knowing the secret key
- This is called **authenticated encryption**

Chosen-Ciphertext Attacks (CCA) Security



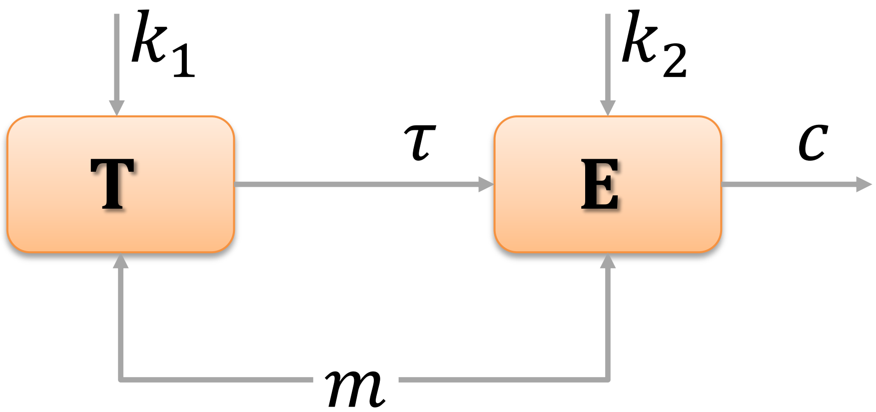
- Both **encryption** and **decryption** queries
  - Cannot query on challenge ciphertext
- Captures **non-malleability**

Encrypt-and-Authenticate



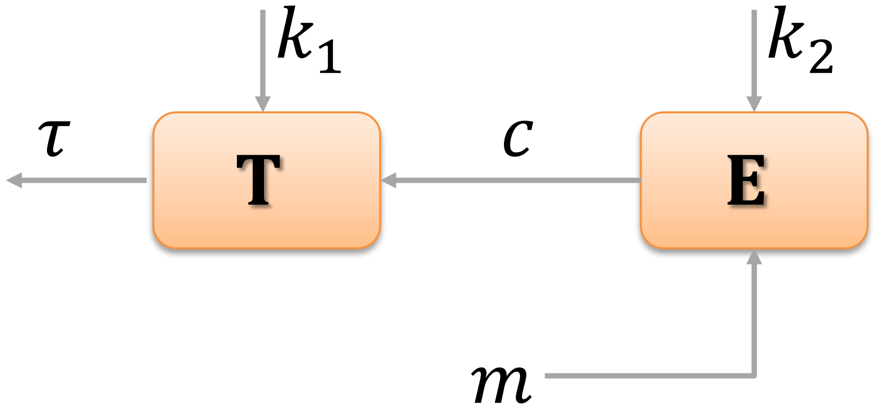
- Output: $c' = (c, \tau)$
- Insecure **in general**
  - Consider the function $\mathbf{T}$ that **reveals** the first bit of m; this is **still** UF-CMA, but now c' is **not** even CPA secure

Authenticate-then-Encrypt



- Output: c
- Insecure **in general**
  - Consider the function $\mathbf{E}'$ that first encrypts m using a CPA secure $\mathbf{E}$ and then **encodes** each bit using two bits: $0 \rightarrow 00$ and $1 \rightarrow 01$ or 10
  - Ciphertexts containing 11 are **invalid**

Encrypt-then-Authenticate

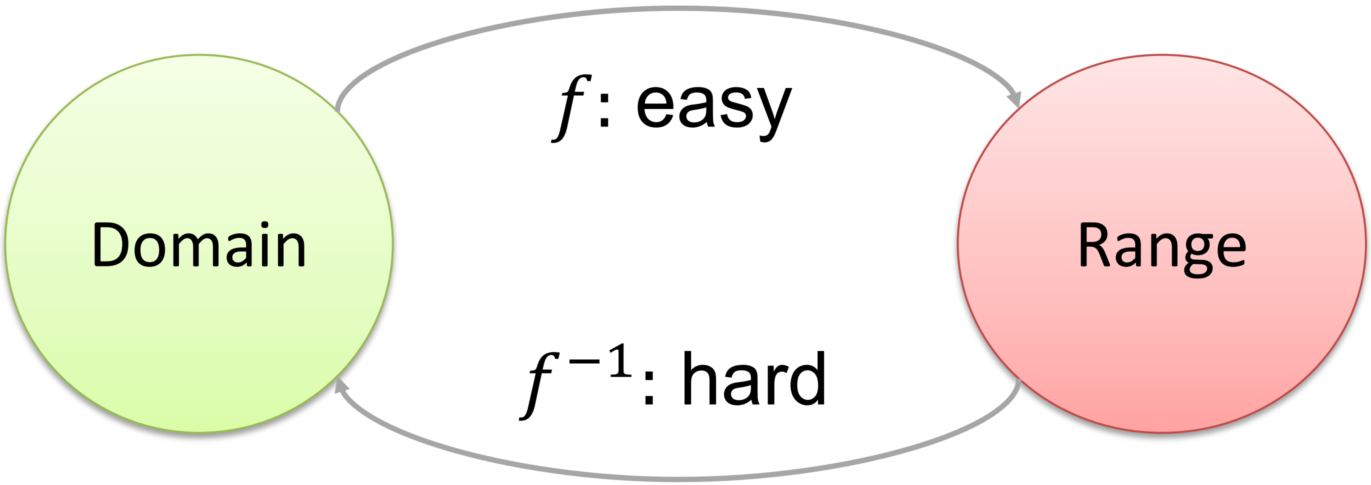


- Output: $c' = (c, \tau)$
- **Always secure!**
  - For **any instantiation** of secure $\mathbf{E}$ and $\mathbf{T}$

A Brief Tour of Minicrypt

One-Way Functions

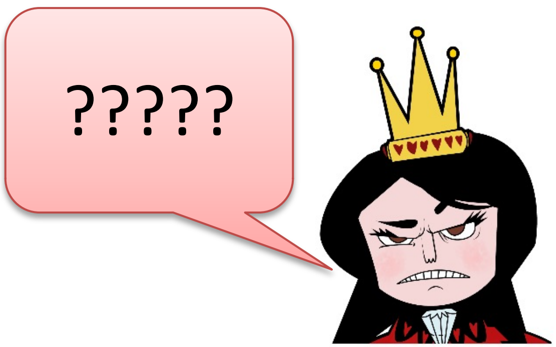
- Functions that are **easy to compute** but **hard to invert**



- Intimately connected to $P \neq NP$
- Minicrypt: There are OWFs but **no public-key cryptography** is possible

Pseudorandom Generators

- A PRG expands a truly random (but **short**) seed into a **much longer** sequence that **looks random** (but it's not!)



$$\mathbf{G}(s) = y \in \{0,1\}^n$$

$$y \leftarrow_{\$} \{0,1\}^n$$

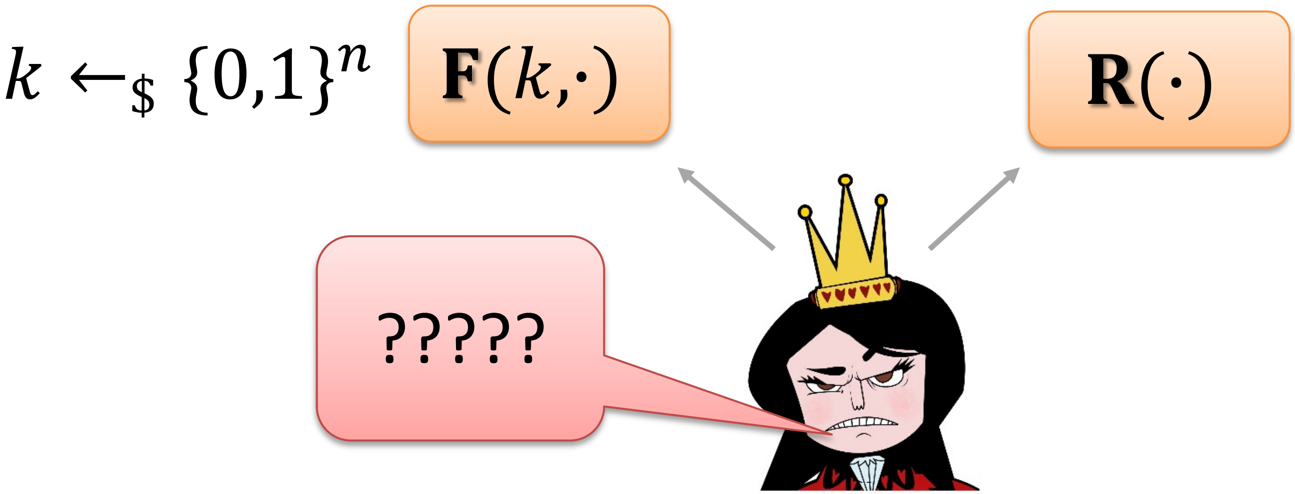
- OWF⇔PRG⇔SKE (one-time)
- One-time secure SKE: $\mathbf{E}(k,m) = \mathbf{G}(k) \oplus m$

PRGs from OWFs

- Given y, which bits of x are **hard to compute**?
  - We know x is hard to compute, but maybe one can always compute **the first bit** of x
- **Hard-core bit:** We say h is hard core for f if given $y = f(x)$ it is hard to find the bit $h(x)$
  - **Fundamental fact:** Every OWF has a hard-core bit!
- If f is a one-way **permutation** (OWP), $\mathbf{G}(s) = f(s)||h(s)$ is a PRG with **1-bit stretch**
  - **Amplification:** Let $s_0 = s$, run $\mathbf{G}(s_i) = s_{i+1}||b_i$ for each $i = 0,1,2,\ldots$ and output $b_1, b_2, \ldots$

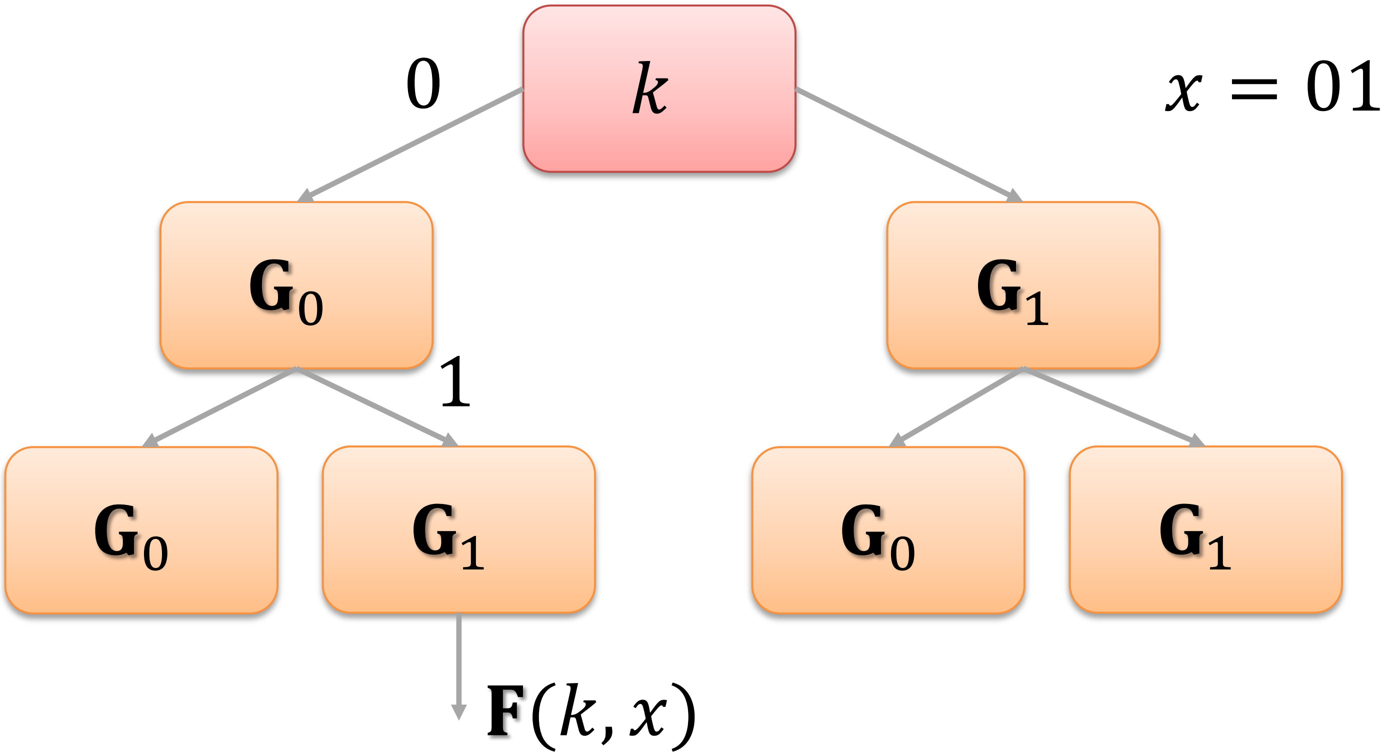
Pseudorandom Functions

- Consider a keyed function $\mathbf{F}(k, x)$ mapping $\{0,1\}^n$ into $\{0,1\}^n$ (for a fixed key k)
- Hard to distinguish $\mathbf{F}(k,\cdot)$ from **truly random function** $\mathbf{R}(\cdot)$

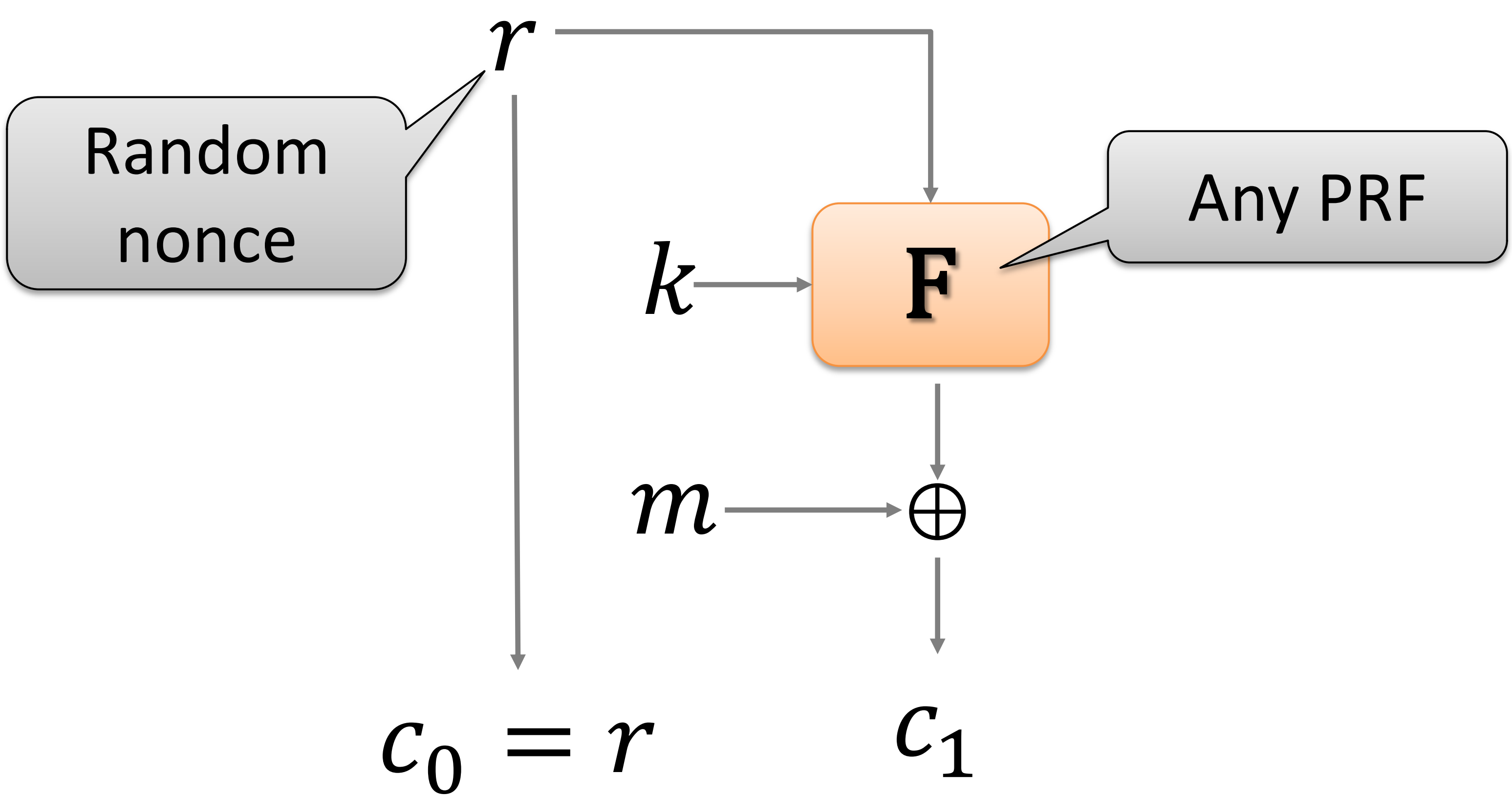


The GGM Tree

- PRG ⇒ PRF (other direction also true)
- Let $\mathbf{G}(s) = (\mathbf{G}_0(s), \mathbf{G}_1(s))$ be a **length doubling** PRG



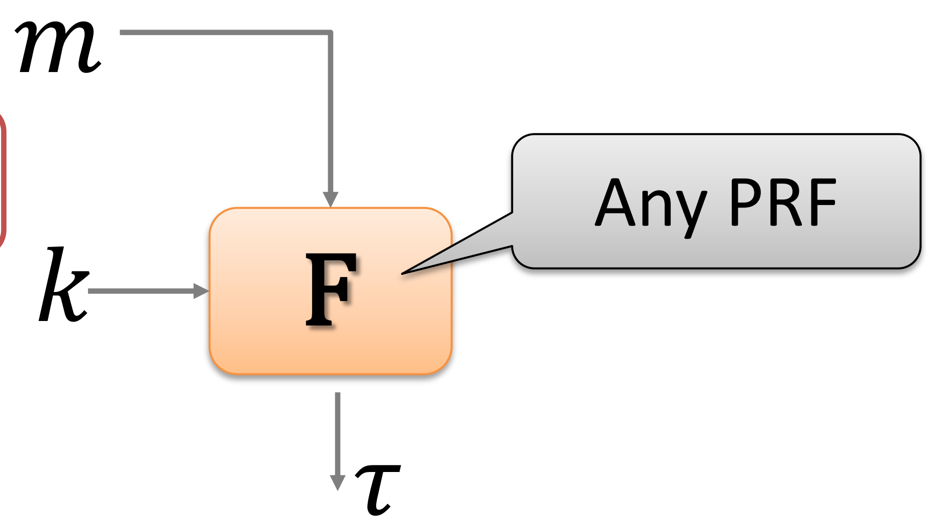
CPA-Secure SKE from PRFs



$$\mathbf{F}(k, c_0) \oplus c_1 = m$$

PRFs as MACs

$$\mathbf{T}(k,m) = \mathbf{F}(k,m)$$

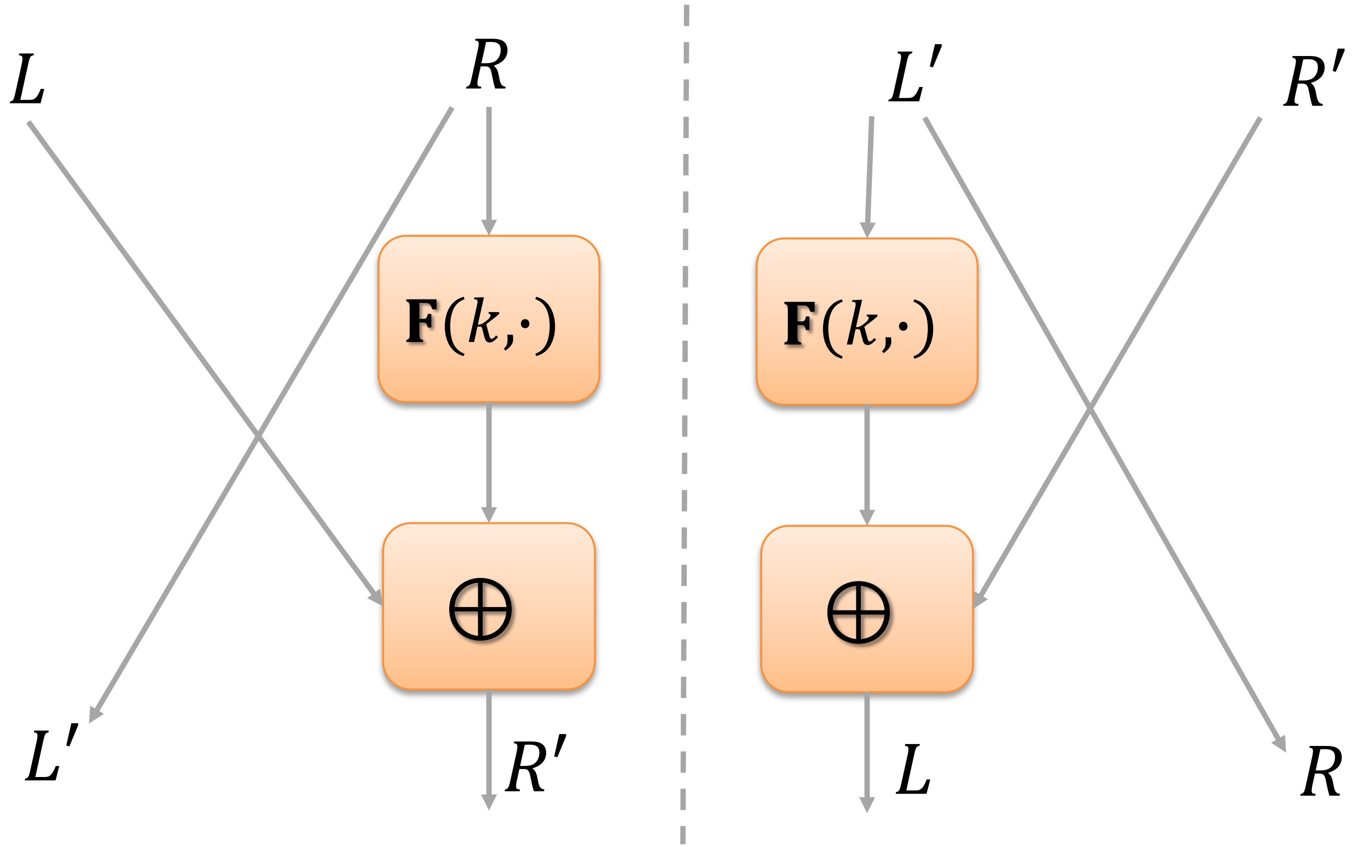


- Every PRF is a **fixed-length** MAC
- **Domain extension:**

CR Hash Function

$$\mathbf{T}'(k,m) = \mathbf{F}(k,\mathbf{H}(m))$$

Feistel Networks



$$\Psi_F(L,R) = \big(R, L \oplus \mathbf{F}(k,R)\big)$$

$$\Psi_F^{-1}(L',R') = (R' \oplus \mathbf{F}(k,L'), L')$$

Luby-Rackoff Theorems

- Define the r-round **Feistel network** $\Psi_{\mathcal{F}}[r]$ as:

$$\Psi_{F_1,\ldots,F_r}(L,R) = \Psi_{F_r}\left(\Psi_{F_{r-1}}\left(\ldots\left(\Psi_{F_1}(L,R)\right)\right)\right)$$

$$\Psi^{-1}_{F_1,\ldots,F_r}(L',R') = \Psi^{-1}_{F_1}\left(\Psi^{-1}_{F_2}\left(\ldots\left(\Psi^{-1}_{F_r}(L',R')\right)\right)\right)$$

  - Here, $\mathcal{F}$ is a family of PRFs (**independent** keys)
- Fundamental Fact: If $\mathcal{F}$ is a PRF, then $\Psi_{\mathcal{F}}[3]$ is a **pseudorandom permutation** (PRP)
  - And $\Psi_{\mathcal{F}}[4]$ is a **strong PRP** (i.e., adversary can access **inverse** permutation)