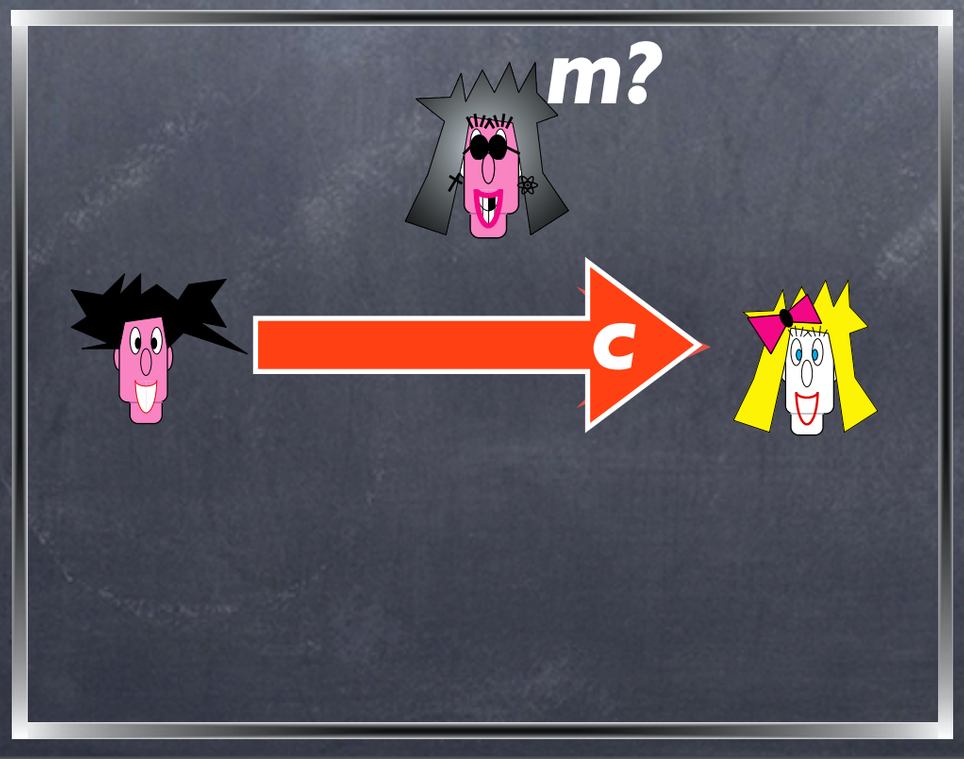
Post-Quantum Cryptography #4

Prof. Claude Crépeau
McGill University

http://crypto.cs.mcgill.ca/~crepeau/WATERLOO

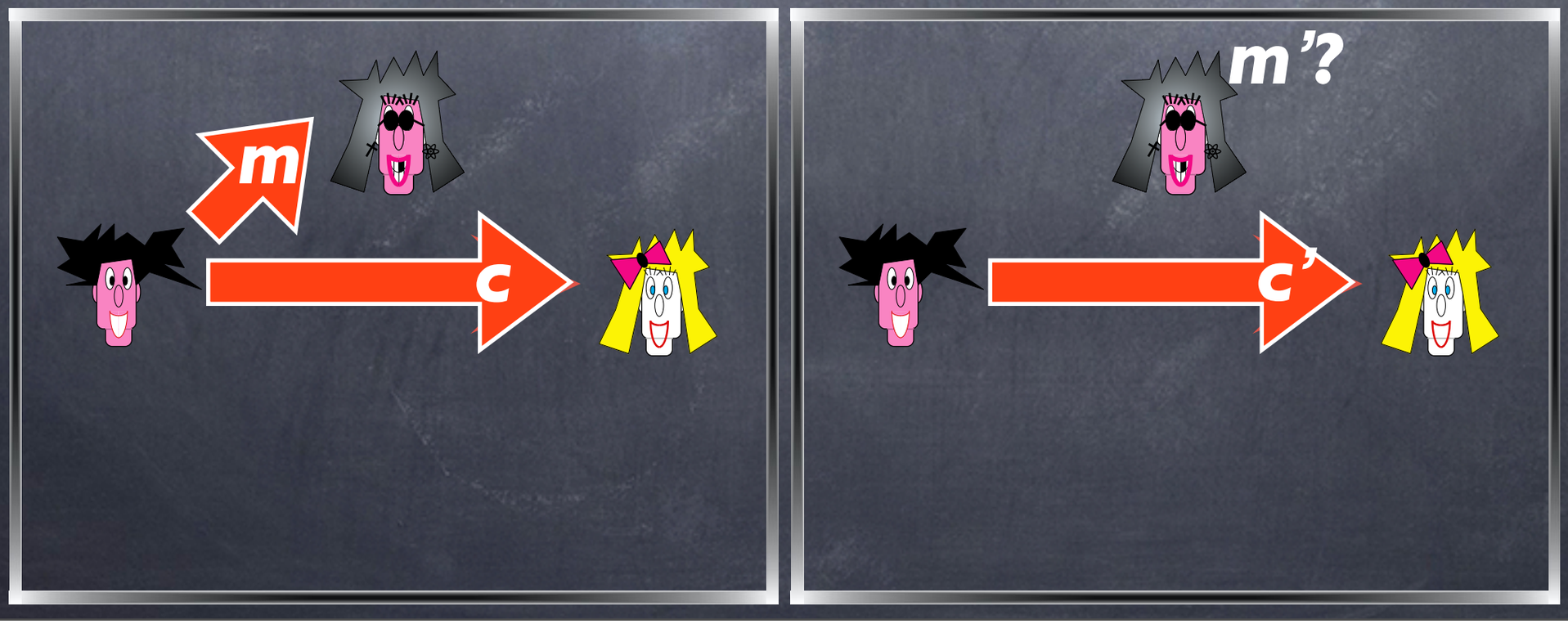
Attack scenarios

- **<u>Ciphertext-only attack:</u>** This is the most basic type of attack and refers to the scenario where the adversary just observes a ciphertext (or multiple ciphertexts) and attempts to determine the underlying plaintext (or plaintexts).



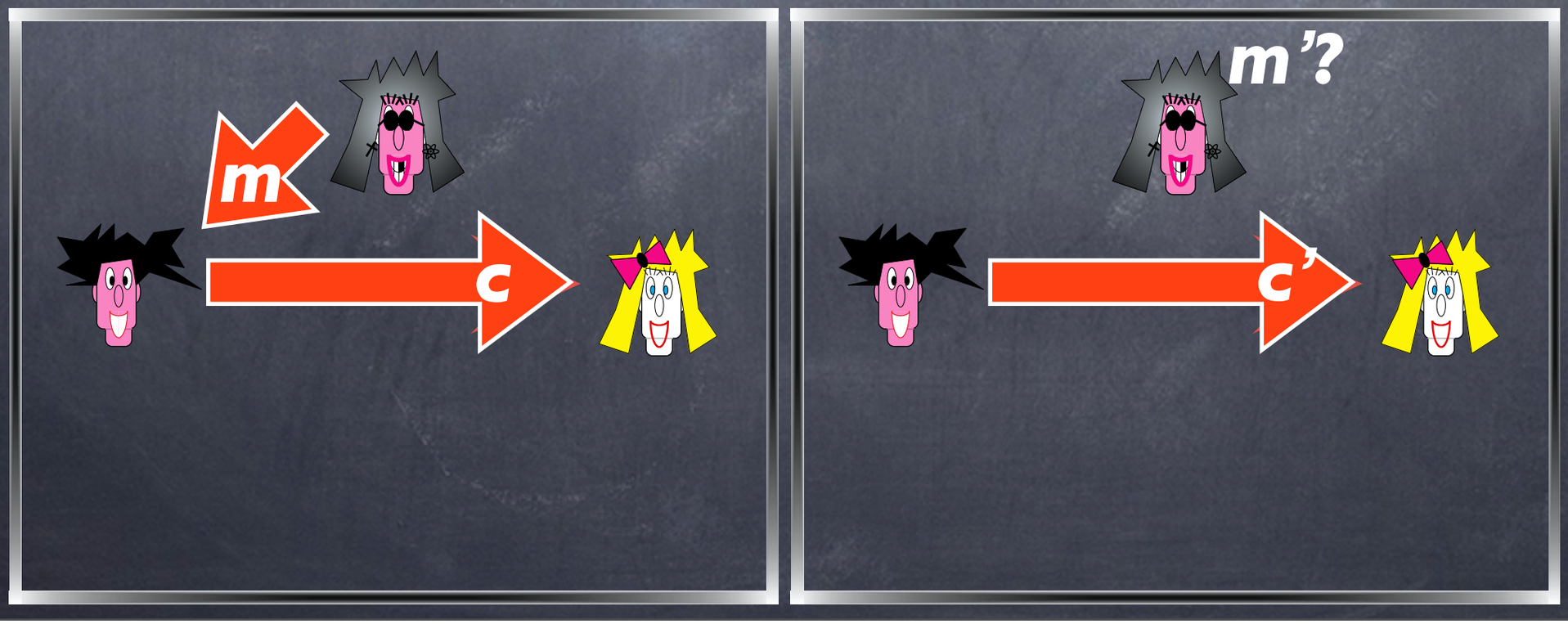
Attack scenarios

- **<u>Known-plaintext attack:</u>** The adversary learns one or more pairs of plaintexts/ciphertexts encrypted under the same key. The aim is to determine the plaintext that was encrypted in some other ciphertext.



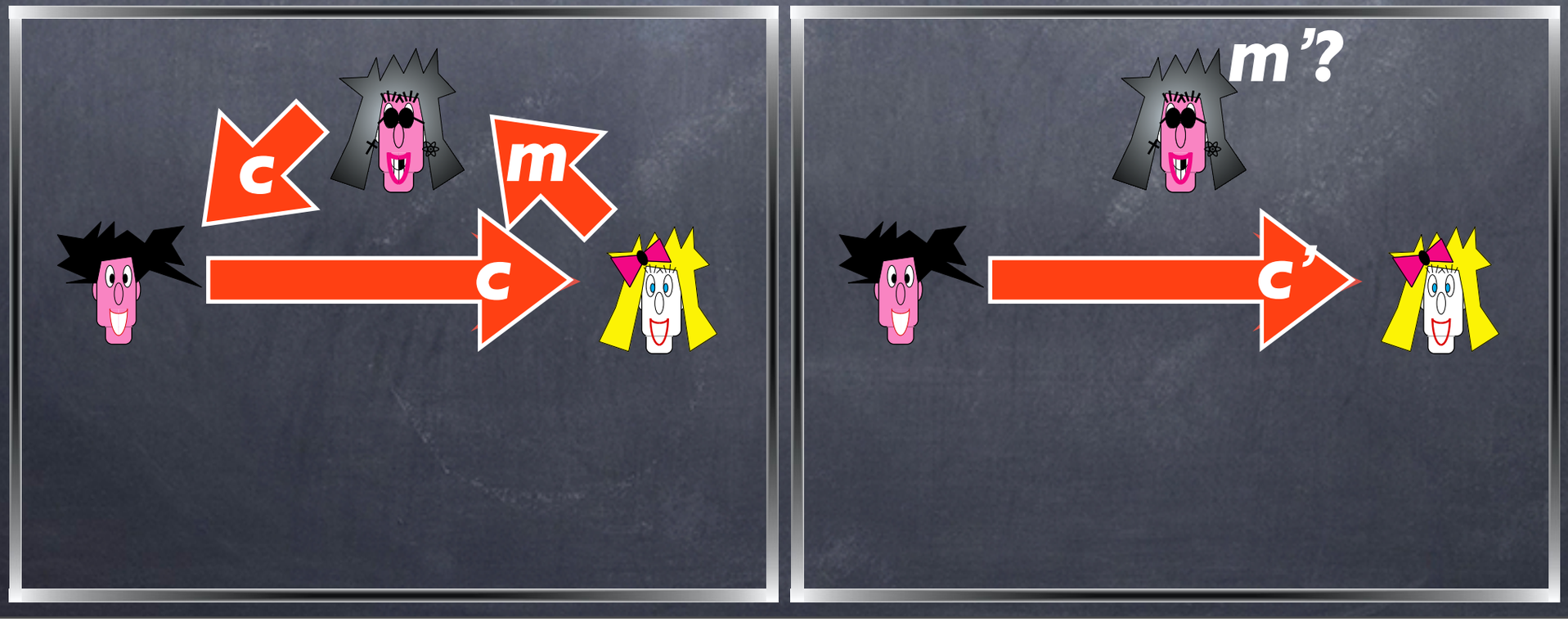
Attack scenarios

- **<u>Chosen-plaintext attack:</u>** The adversary has the ability to obtain the encryption of plaintexts of its choice. It then attempts to determine the plaintext that was encrypted in some other ciphertext.



Attack scenarios

- **<u>Chosen-ciphertext attack:</u>** The adversary is even given the capability to obtain the decryption of ciphertexts of its choice. The adversary's aim, once again, is to determine the plaintext that was encrypted in some other ciphertext.



What is secure encryption?

Answer 1 — an encryption scheme is secure if no adversary can find the *secret key* when given a ciphertext.

secure encryption.

Answer 2 — an encryption scheme is secure if no adversary can find the *plaintext* that corresponds to the ciphertext.

secure encryption.

Answer 3 — an encryption scheme is secure if no adversary can determine *any character* of the plaintext that corresponds to the ciphertext.

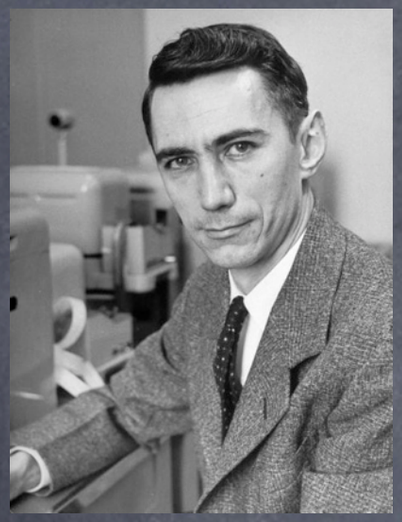
secure encryption.

Answer 4 — an encryption scheme is secure if no adversary can derive any *meaningful information* about the plaintext from the ciphertext.

- Definitions of security should suffice for all potential applications.

secure encryption.

The Final Answer — an encryption scheme is secure if no adversary can compute any *function* of the plaintext from the ciphertext.



Perfect Secrecy

DEFINITION 2.1 *An encryption scheme* **(Gen, Enc, Dec)** *over a message space* M *is* perfectly secret *if for every probability distribution over* M*, every message* $\mathbf{m} \in M$*, and every ciphertext* $\mathbf{c} \in C$ *for which* **Pr[C = c] > 0** :

$$\mathbf{Pr[}M = m \mid C = c\mathbf{]} = \mathbf{Pr[}M = m\mathbf{]}.$$

An equivalent formulation

LEMMA 2.2 *An encryption scheme* **(Gen, Enc, Dec)** *over a message space* M *is perfectly secret if and only if for every probability distribution over* M*, every message* $\mathbf{m} \in M$*, and every ciphertext* $\mathbf{c} \in C$*:*

$$\mathbf{Pr[C = c \mid M = m] = Pr[C = c]}.$$

Perfect indistinguishability

LEMMA 2.3 *An encryption scheme* **(Gen, Enc, Dec)** *over a message space* M *is perfectly secret if and only if for every probability distribution over* M*, every* $\mathbf{m_0}, \mathbf{m_1} \in M$*, and every* $\mathbf{c} \in C$*:*

$$\mathbf{Pr[C = c \mid M = m_0] = Pr[C = c \mid M = m_1].}$$

Adversarial indistinguishability.

Adversarial indistinguishability.

- This other definition is based on an *experiment* involving an adversary ***A***, and formalizes ***A***'s inability to distinguish the encryption of one plaintext from the encryption of another; we thus call it *adversarial indistinguishability*.

Adversarial indistinguishability.

- This other definition is based on an *experiment* involving an adversary ***A***, and formalizes ***A***'s inability to distinguish the encryption of one plaintext from the encryption of another; we thus call it *adversarial indistinguishability*.
- This definition will serve as our starting point when we introduce the notion of computational security in the next chapter.

Adversarial indistinguishability.

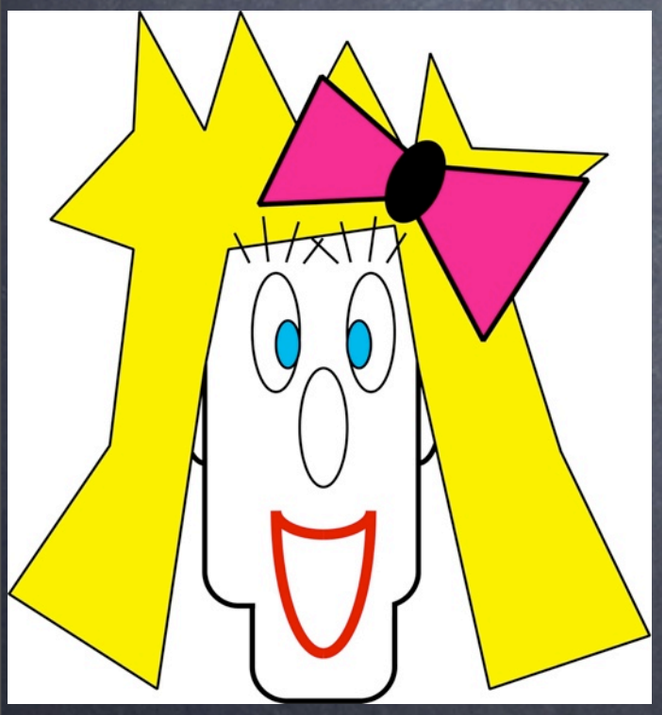
Adversarial indistinguishability.

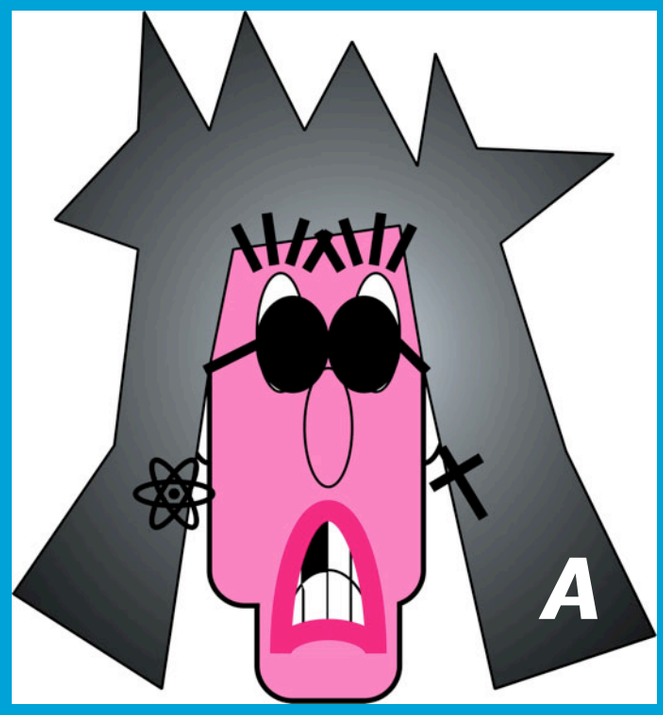
- The experiment is defined for any encryption scheme **Π = (Gen, Enc, Dec)** over message space M and for any adversary ***A***.

Adversarial indistinguishability.

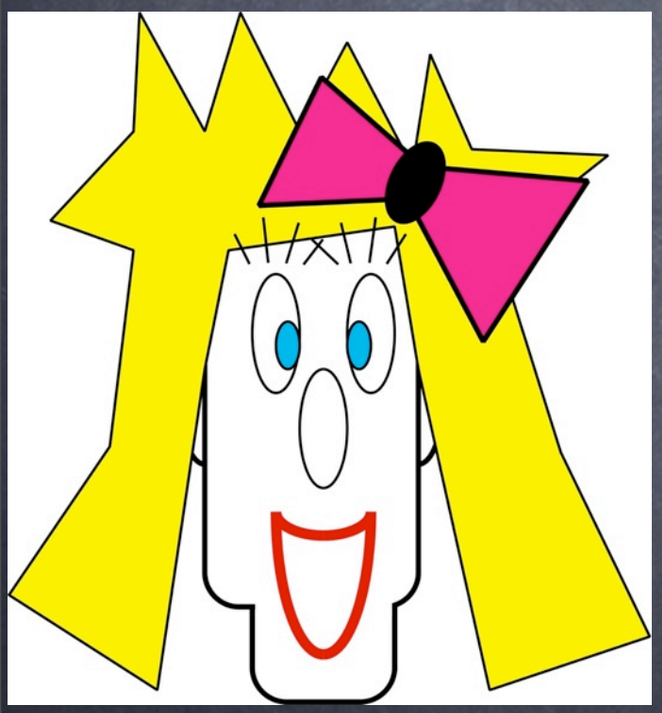
- The experiment is defined for any encryption scheme **Π = (Gen, Enc, Dec)** over message space M and for any adversary ***A***.
- We let $\mathbf{PrivK}^{eav}_{A,\Pi}$ denote an execution of the experiment for a given **Π** and ***A***. The experiment is defined as follows:

$\mathrm{PrivK}^{eav}_{A,\Pi}$

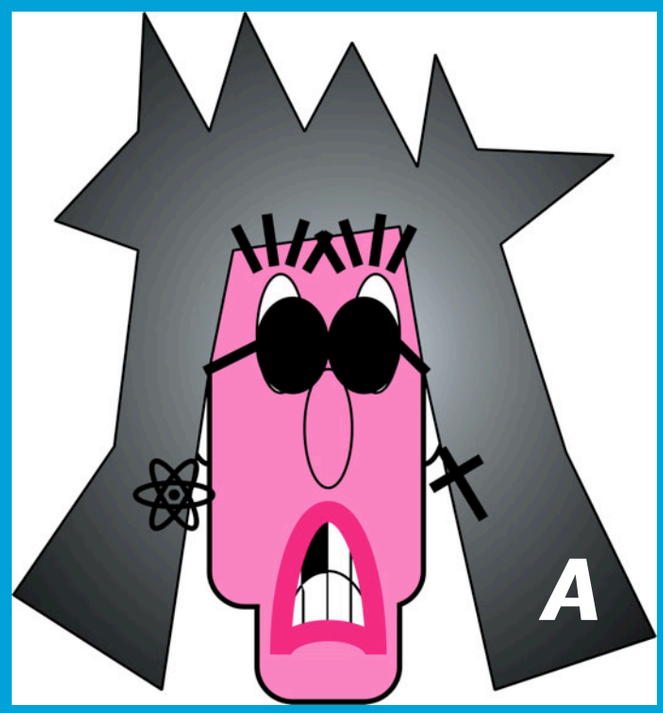




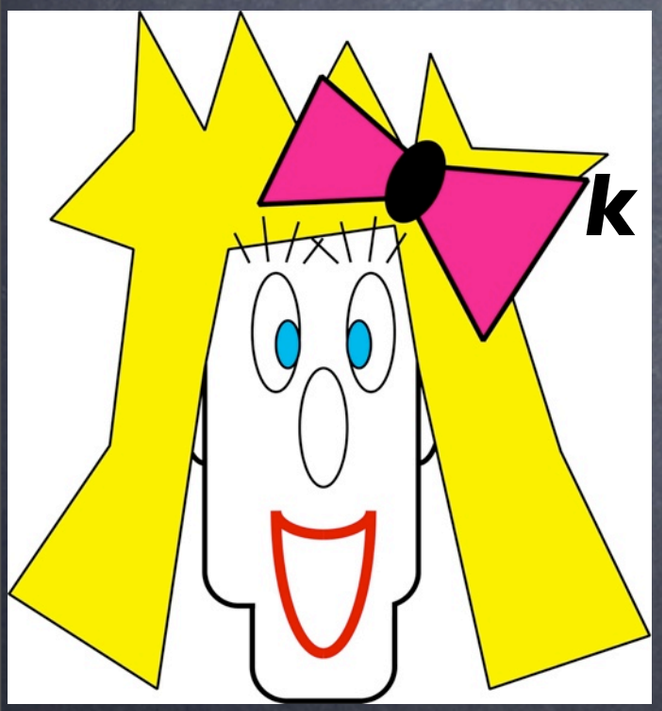
$\mathbf{PrivK}^{eav}_{A,\Pi}$



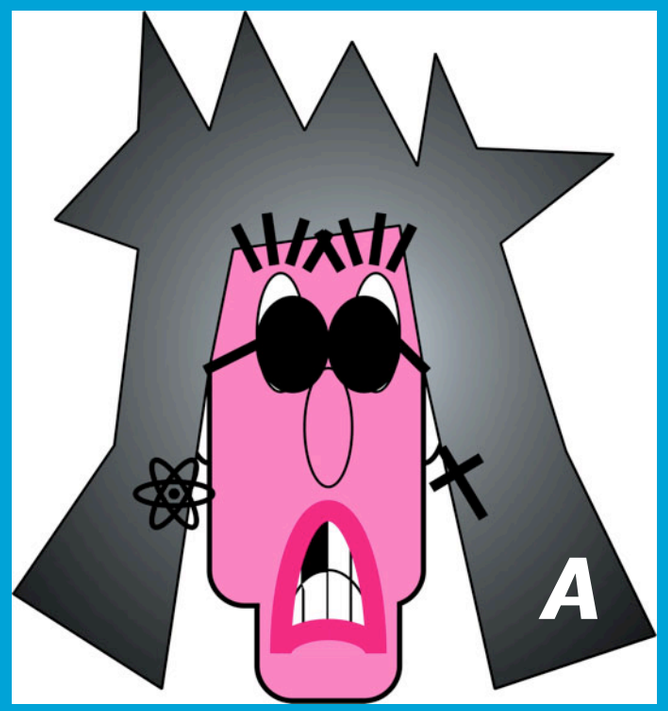
$m_0, m_1 \in \mathcal{M}$



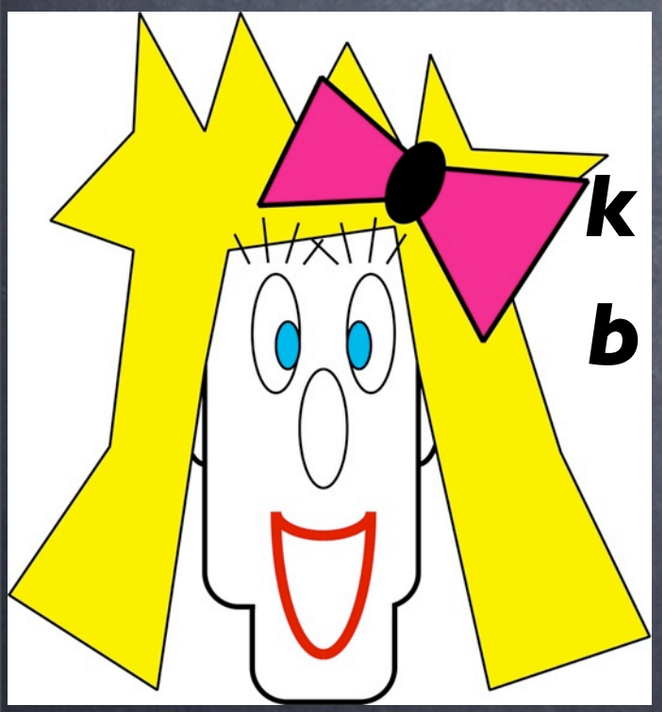
$\mathbf{PrivK}^{eav}_{A,\Pi}$



$m_0, m_1 \in M$



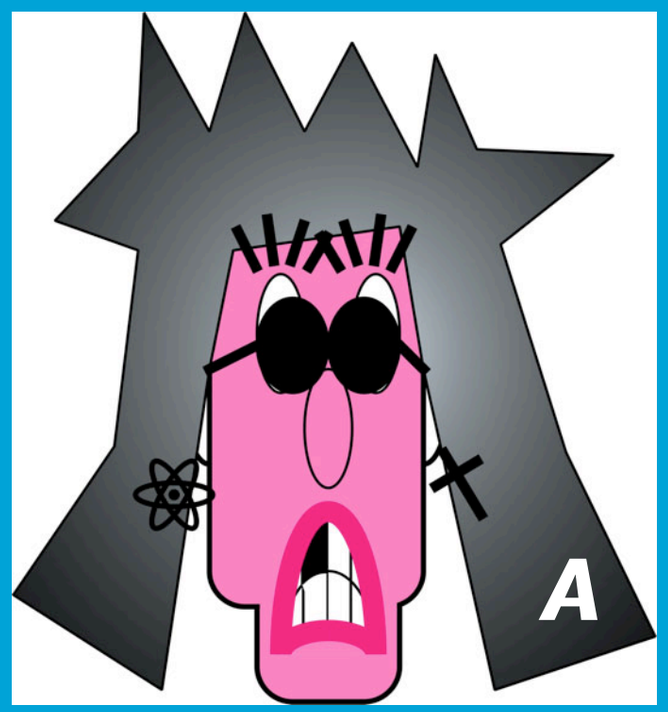
$\mathbf{PrivK}^{eav}_{A,\Pi}$



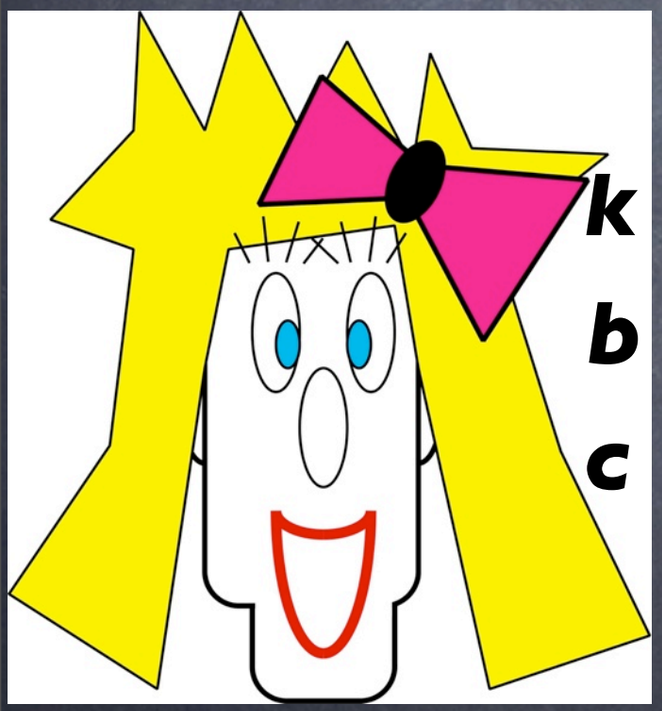
$k \leftarrow \mathbf{Gen}$

$b \leftarrow \{0, 1\}$

$m_0, m_1 \in \mathcal{M}$



$\mathbf{PrivK}^{eav}_{A,\Pi}$

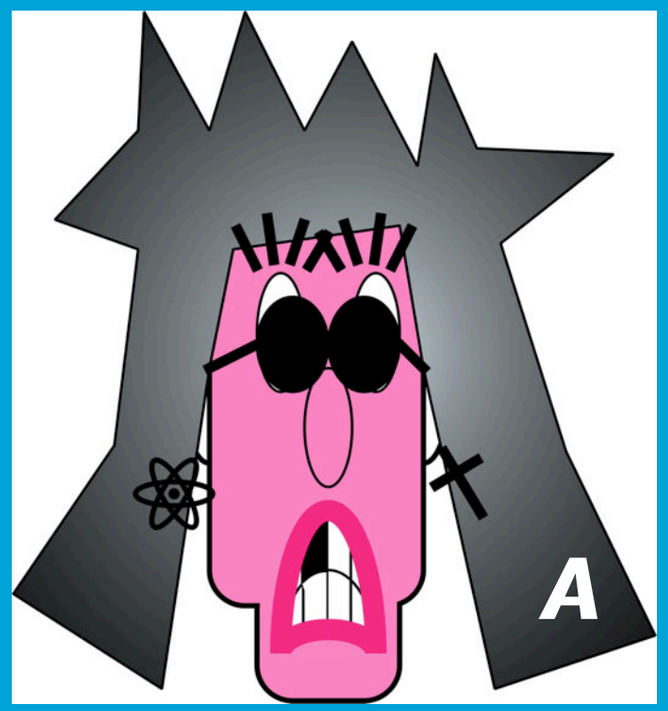


$k \leftarrow \mathbf{Gen}$

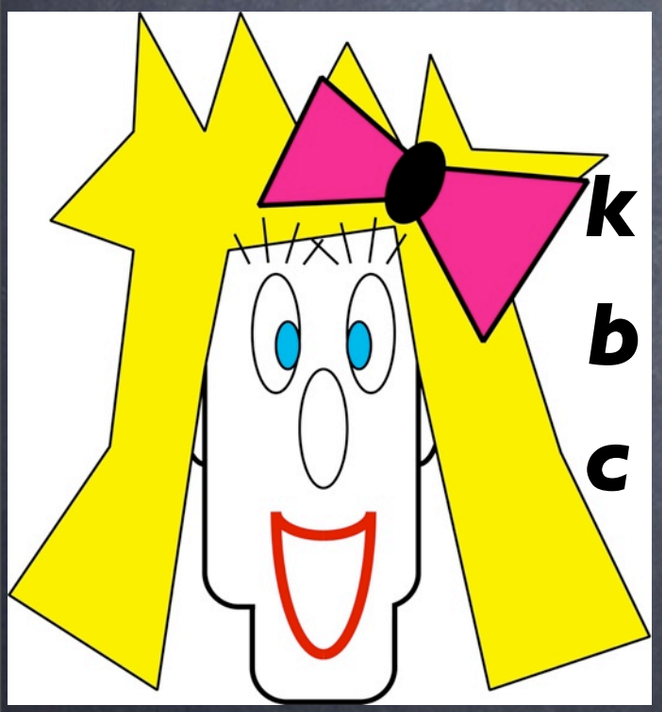
$b \leftarrow \{0, 1\}$

$c \leftarrow \mathbf{Enc}_k(m_b)$

$m_0, m_1 \in M$



$\mathbf{PrivK}^{eav}_{A,\Pi}$



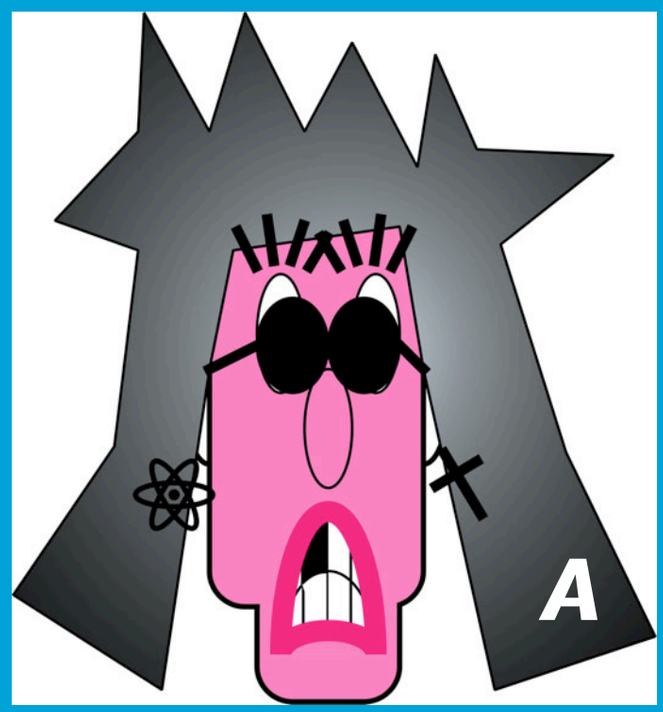
$k \leftarrow \mathbf{Gen}$

$b \leftarrow \{0, 1\}$

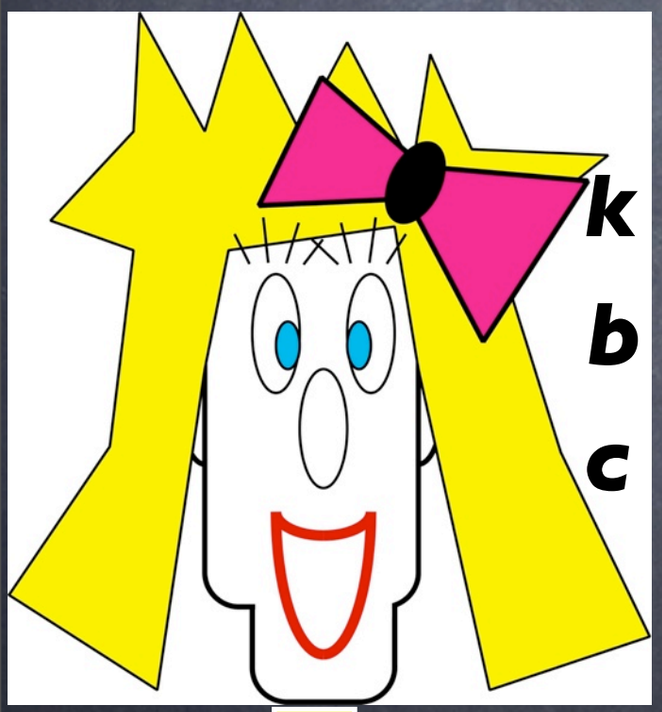
$c \leftarrow \mathbf{Enc}_k(m_b)$

$m_0, m_1 \in M$

c



$\mathbf{PrivK}^{eav}_{A,\Pi}$



$k \leftarrow \mathrm{Gen}$

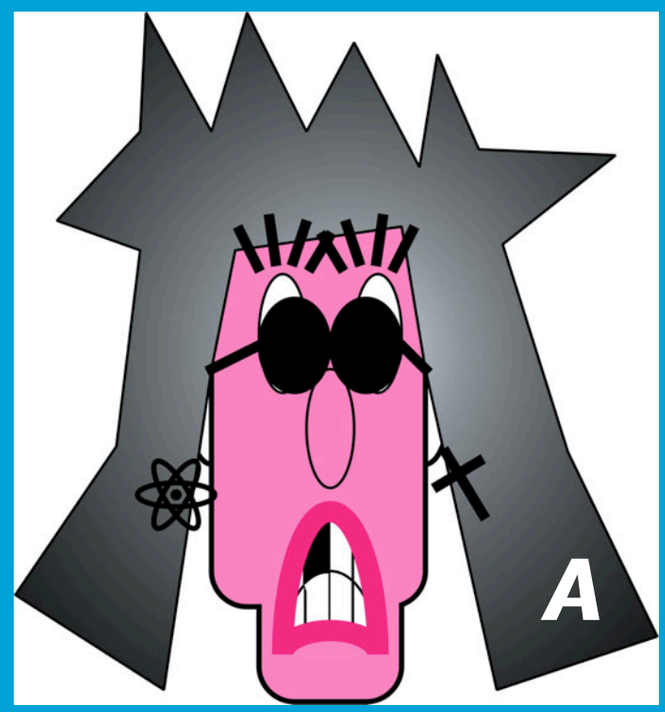
$b \leftarrow \{0, 1\}$

$c \leftarrow \mathrm{Enc}_k(m_b)$

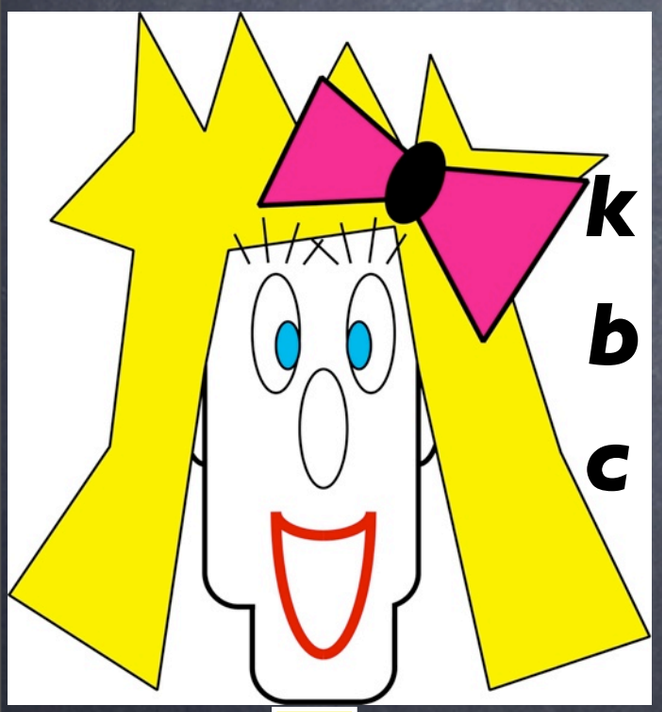
b

$m_0, m_1 \in M$

c



$\mathbf{PrivK}^{eav}_{A,\Pi}$



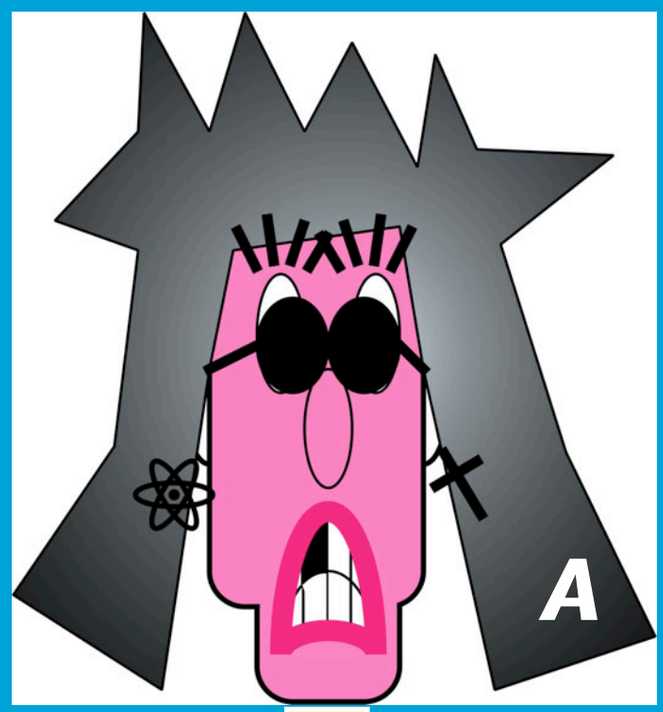
$k \leftarrow \mathrm{Gen}$

$b \leftarrow \{0, 1\}$

$c \leftarrow \mathrm{Enc}_k(m_b)$

$m_0, m_1 \in M$

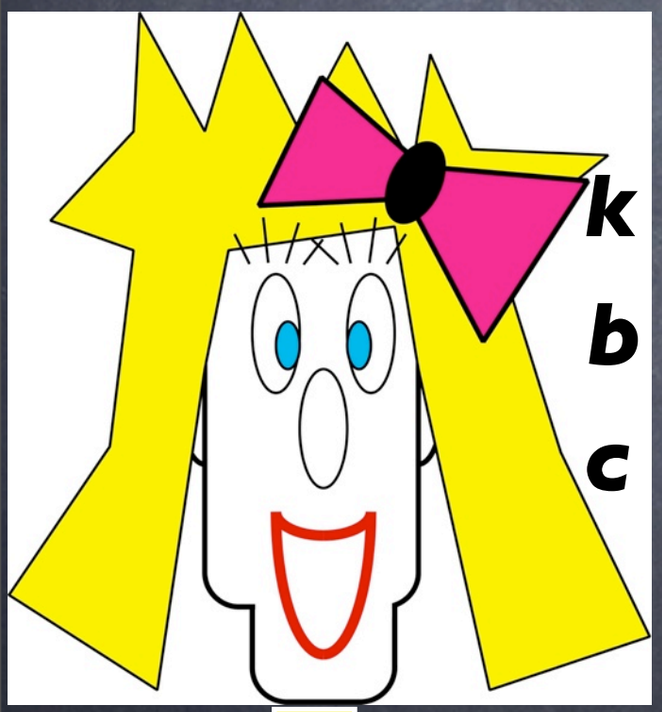
c



b'

b

$\mathbf{PrivK}^{eav}_{A,\Pi}$



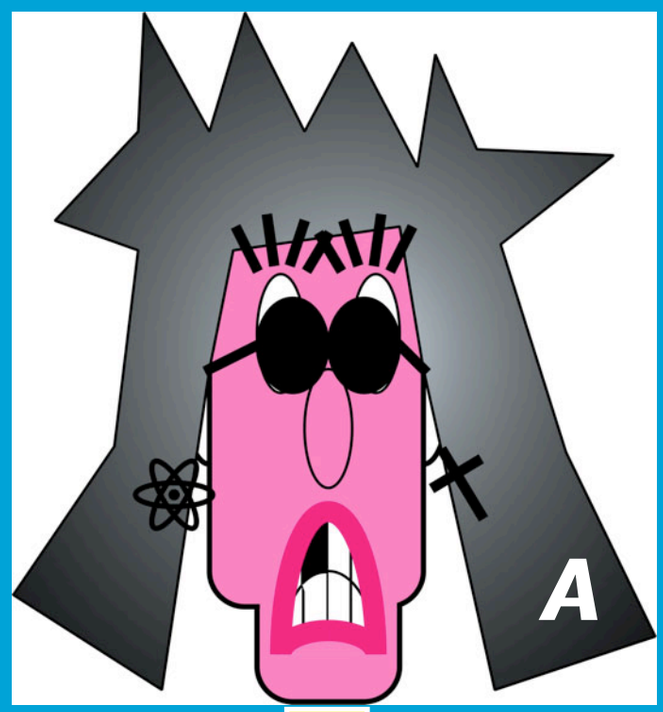
$k \leftarrow \mathrm{Gen}$

$b \leftarrow \{0, 1\}$

$c \leftarrow \mathrm{Enc}_k(m_b)$

$m_0, m_1 \in M$

c



b

b'

$b = b'$?

Adversarial indistinguishability.

Adversarial indistinguishability.

$\mathbf{PrivK}^{eav}_{A,\Pi}$:

Adversarial indistinguishability.

$\mathbf{PrivK}^{eav}_{A,\Pi}$:

1. *Adversary* ***A*** *outputs a pair of messages* $m_0, m_1 \in \mathcal{M}$.

Adversarial indistinguishability.

$\mathbf{PrivK}^{eav}_{A,\Pi}$:

1. *Adversary* ***A*** *outputs a pair of messages* $\mathbf{m_0}, \mathbf{m_1} \in \mathcal{M}$.
2. *A random key* ***k*** *is generated by running* **Gen**, *and a random bit* $\mathbf{b} \leftarrow \{\mathbf{0}, \mathbf{1}\}$ *is chosen (by some imaginary entity that is running the experiment with* ***A***.) *A ciphertext* $\mathbf{c} \leftarrow \mathbf{Enc}_k(\mathbf{m}_b)$ *is computed and given to* ***A***.

Adversarial indistinguishability.

$\mathbf{PrivK}^{eav}_{A,\Pi}$:

1. *Adversary* ***A*** *outputs a pair of messages* $\mathbf{m_0}, \mathbf{m_1} \in \mathcal{M}$.

2. *A random key* ***k*** *is generated by running* **Gen**, *and a random bit* $\mathbf{b} \leftarrow \{\mathbf{0}, \mathbf{1}\}$ *is chosen (by some imaginary entity that is running the experiment with* ***A***.) *A ciphertext* $\mathbf{c} \leftarrow \mathbf{Enc}_k(\mathbf{m}_b)$ *is computed and given to* ***A***.

3. ***A*** *outputs a bit* $\mathbf{b}'$.

Adversarial indistinguishability.

$\mathbf{PrivK}^{eav}_{A,\Pi}$:

1. *Adversary* ***A*** *outputs a pair of messages* $m_0, m_1 \in M$.

2. *A random key* ***k*** *is generated by running* **Gen**, *and a random bit* $b \leftarrow \{0, 1\}$ *is chosen (by some imaginary entity that is running the experiment with* ***A***.*) A ciphertext* $c \leftarrow \mathbf{Enc}_k(m_b)$ *is computed and given to* ***A***.

3. ***A*** *outputs a bit* b'.

4. *The output of the experiment is defined to be* **1** *if* $b' = b$, *and* **0** *otherwise.*

Adversarial indistinguishability.

Adversarial indistinguishability.

- We write $\mathbf{PrivK}^{eav}_{A,\Pi} = 1$ if the output is **1** and in this case we say that ***A*** succeeded.

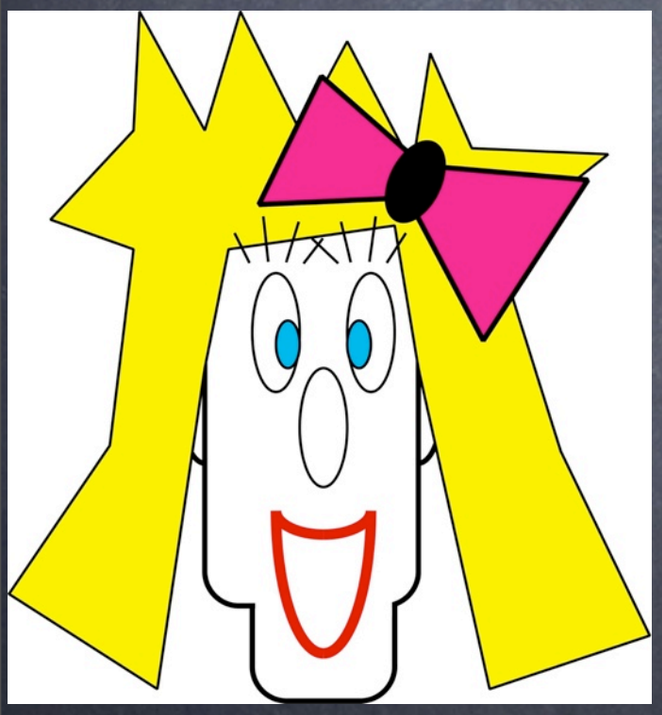
Adversarial indistinguishability.

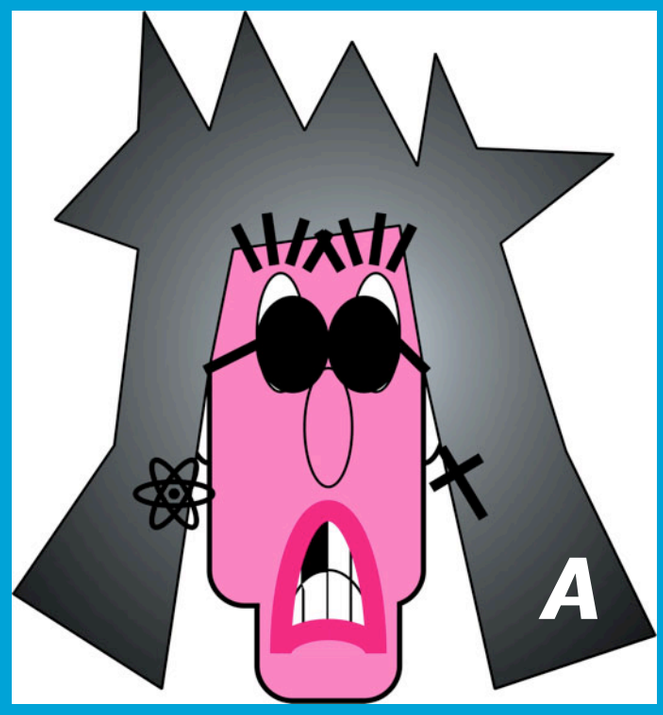
- We write $\mathbf{PrivK}^{eav}_{A,\Pi} = \mathbf{1}$ if the output is **1** and in this case we say that ***A*** succeeded.
- One should think of ***A*** as trying to guess the value of ***b*** that is chosen in the experiment, and ***A*** succeeds when its guess ***b′*** is correct.

Adversarial indistinguishability.

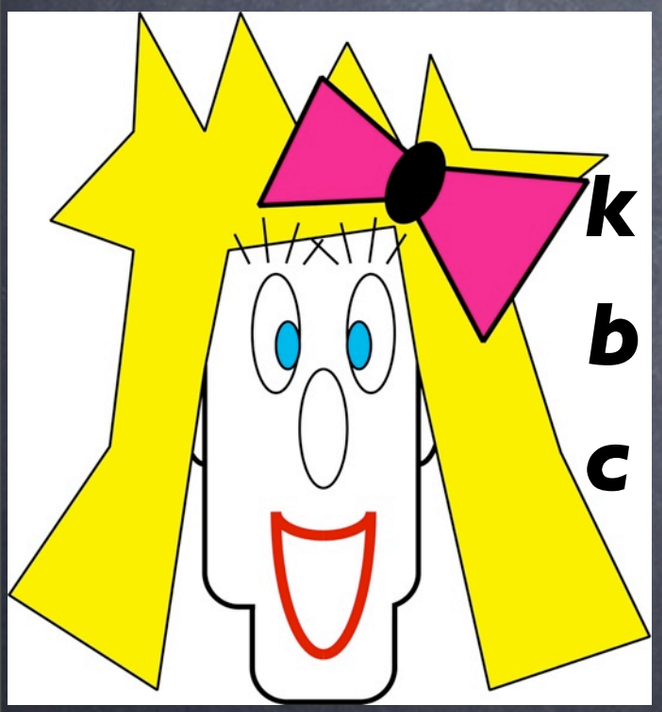
- We write $\mathbf{PrivK}^{eav}_{A,\Pi} = 1$ if the output is **1** and in this case we say that ***A*** succeeded.
- One should think of ***A*** as trying to guess the value of ***b*** that is chosen in the experiment, and ***A*** succeeds when its guess ***b′*** is correct.
- The alternate definition we now give states that an encryption scheme is perfectly secret if *no* adversary ***A*** can succeed with probability any better than $1/2$.

$\mathrm{PrivK}^{eav}_{A,\Pi}$





$\mathbf{PrivK}^{eav}_{A,\Pi}$



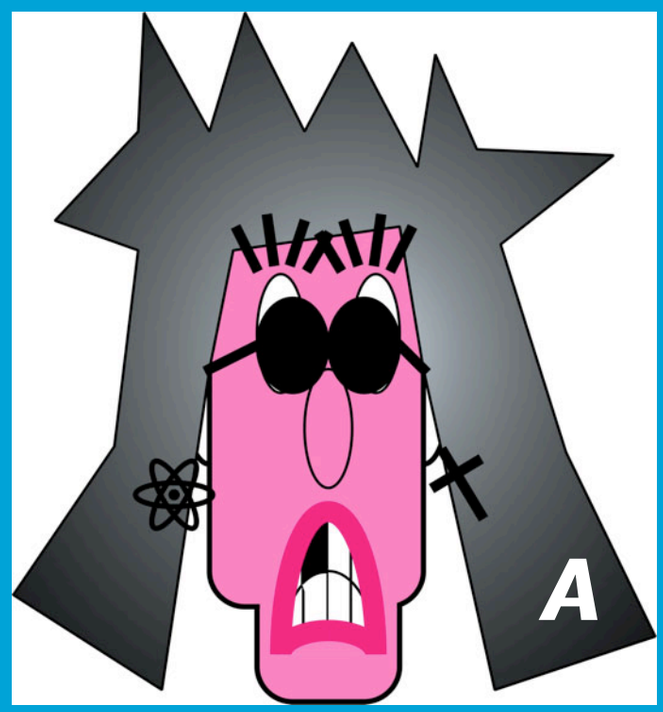
$k \leftarrow \mathbf{Gen}$

$b \leftarrow \{0, 1\}$

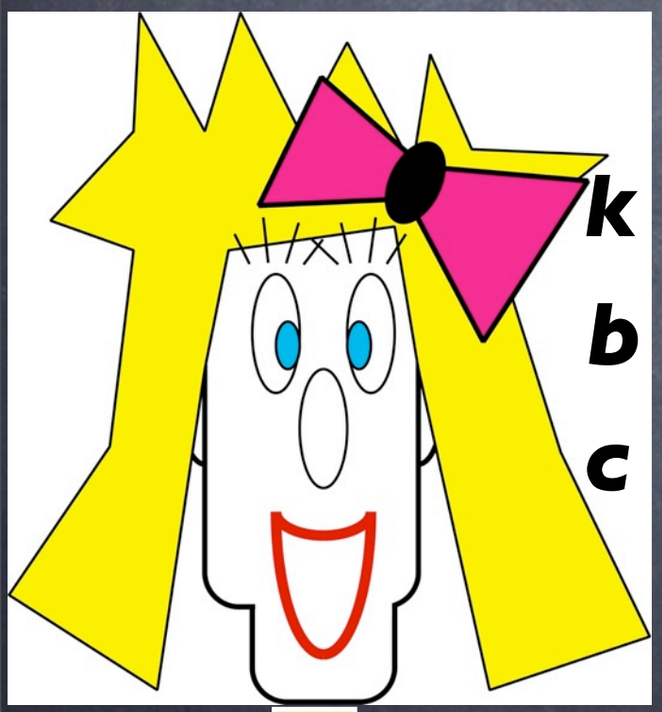
$c \leftarrow \mathbf{Enc}_k(m_b)$

$m_0, m_1 \in M$

c



$\mathbf{PrivK}^{eav}_{A,\Pi}$



$k \leftarrow \mathrm{Gen}$

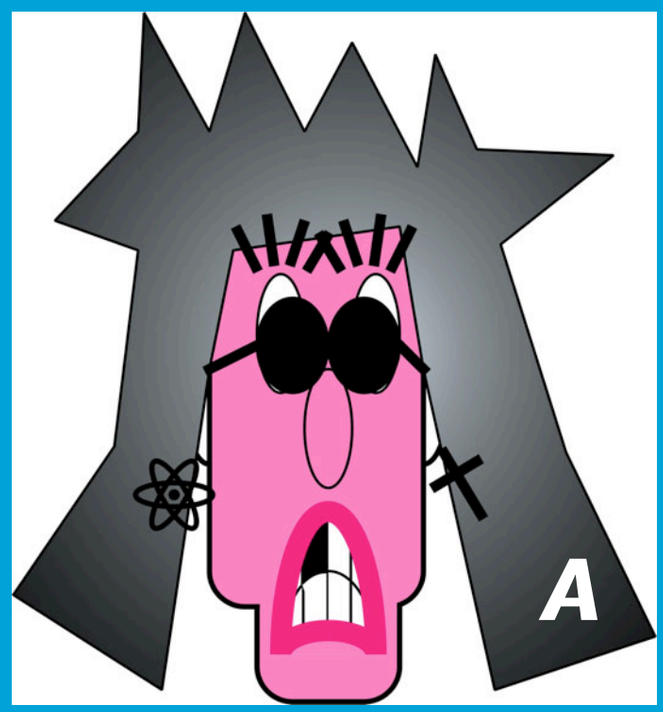
$b \leftarrow \{0, 1\}$

$c \leftarrow \mathrm{Enc}_k(m_b)$

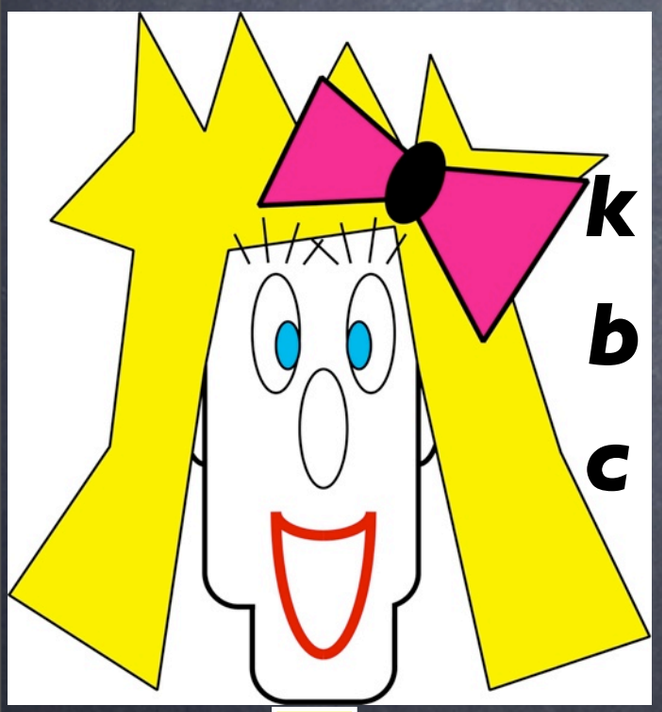
b

$m_0, m_1 \in M$

c



$\mathbf{PrivK}^{eav}_{A,\Pi}$



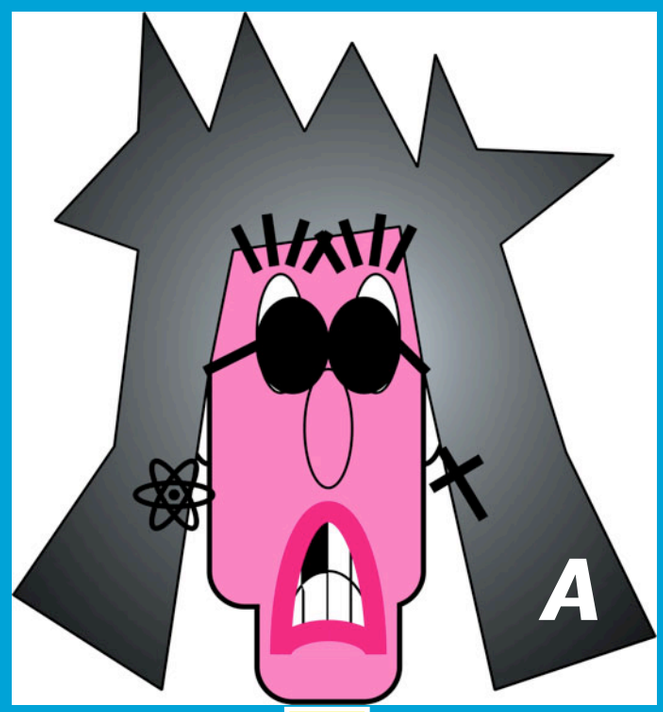
$k \leftarrow \mathrm{Gen}$

$b \leftarrow \{0, 1\}$

$c \leftarrow \mathrm{Enc}_k(m_b)$

$m_0, m_1 \in M$

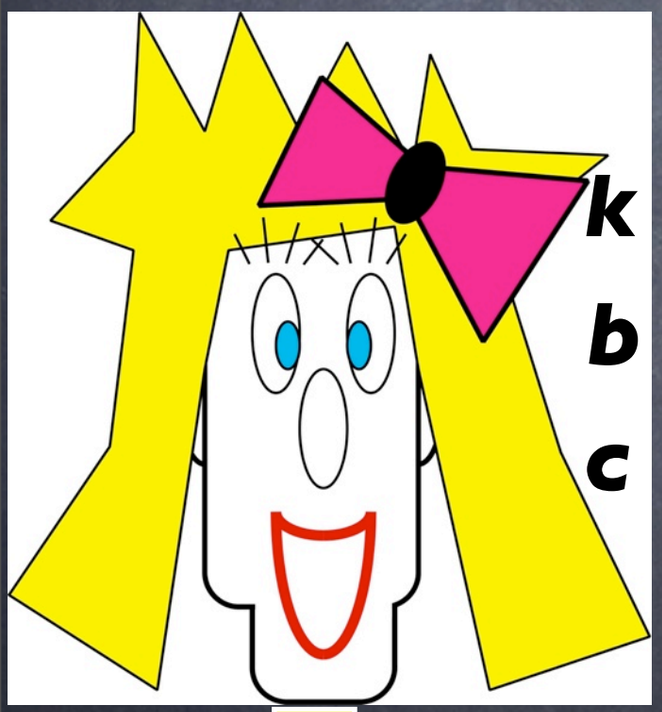
c



b'

b

$\mathbf{PrivK}^{eav}_{A,\Pi}$



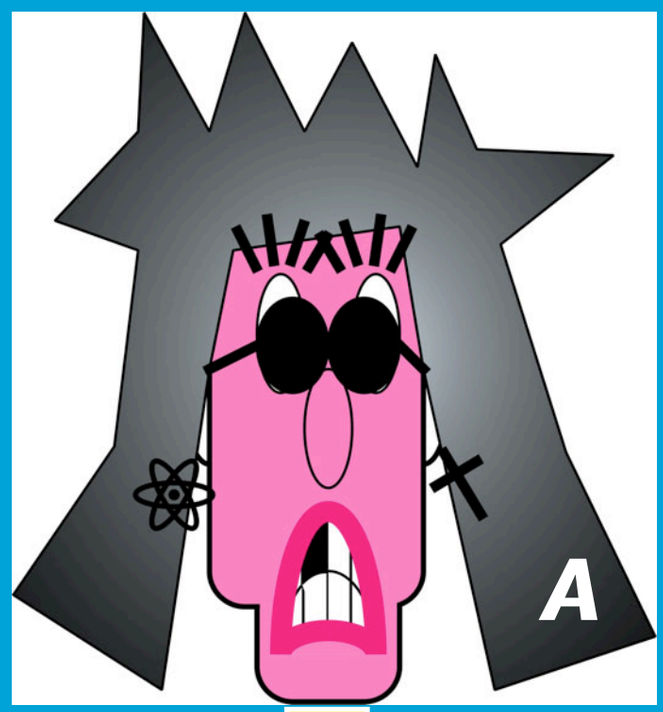
$k \leftarrow \mathrm{Gen}$

$b \leftarrow \{0, 1\}$

$c \leftarrow \mathrm{Enc}_k(m_b)$

$m_0, m_1 \in M$

c

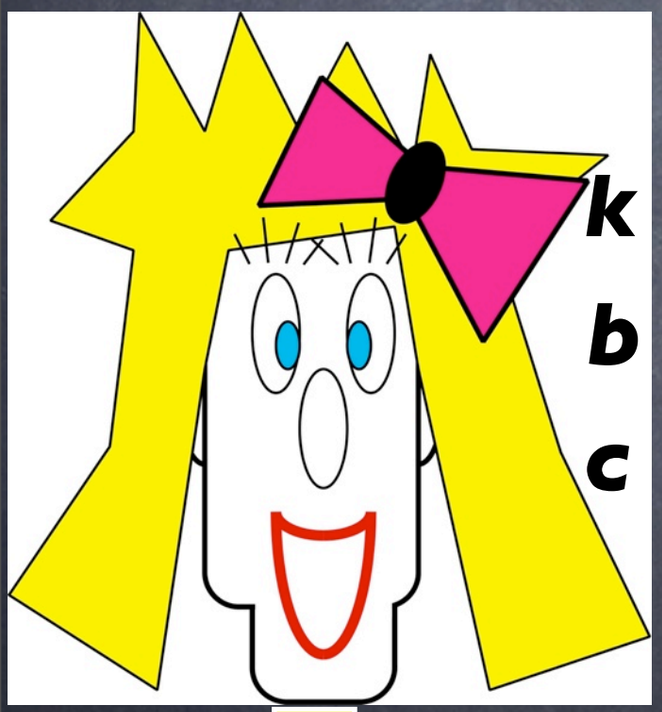


b

b'

$\Pr[\, b = b' \,] = 1/2$

$\mathbf{PrivK}^{eav}_{A,\Pi}$



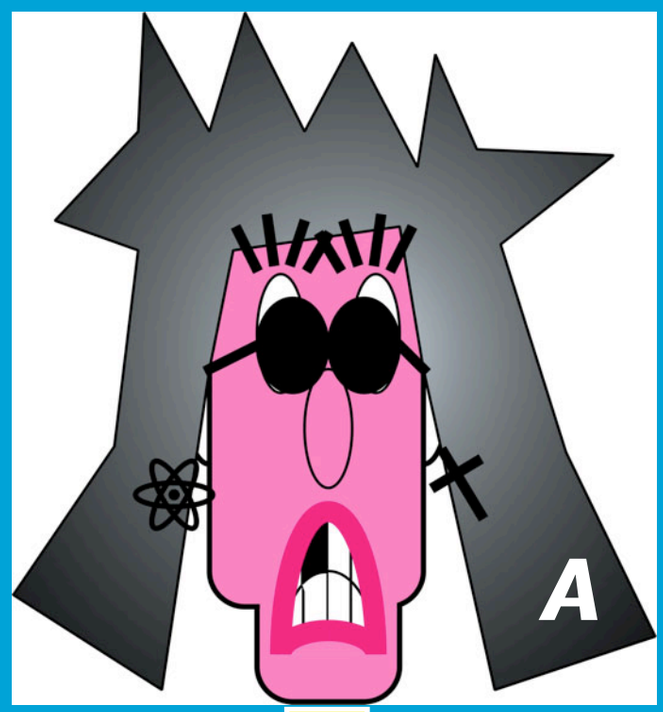
$k \leftarrow \mathbf{Gen}$

$b \leftarrow \{0, 1\}$

$c \leftarrow \mathbf{Enc}_k(m_b)$

$m_0, m_1 \in M$

c



b

b'

perfectly secret

$\Pr[\, b = b' \,] = 1/2$

Adversarial indistinguishability.

DEFINITION 2.4 *An encryption scheme* **Π = (Gen, Enc, Dec)** *over a message space* M *is* perfectly secret *if for every adversary* ***A*** *it holds that*

$$\Pr[\,\mathrm{PrivK}^{\mathrm{eav}}_{A,\Pi} = 1\,] = 1/2\,.$$

Adversarial indistinguishability.

PROPOSITION 2.5 *Let* **(Gen, Enc, Dec)** *be an encryption scheme over a message space* $\mathcal{M}$. *Then* **(Gen, Enc, Dec)** *is perfectly secret with respect to* ***Definition 2.1*** *if and only if it is perfectly secret with respect to* ***Definition 2.4***.

4 Equivalent Formulations

DEFINITION 2.1 *An encryption scheme* **(Gen, Enc, Dec)** *over a message space* $\mathcal{M}$ *is* perfectly secret *if for every probability distribution over* $\mathcal{M}$*, every message* $\mathbf{m} \in \mathcal{M}$*, and every ciphertext* $\mathbf{c} \in \mathcal{C}$ *for which* $\mathbf{Pr[C = c] > 0}$:

$$\mathbf{Pr[M = m \mid C = c] = Pr[M = m]}.$$

LEMMA 2.2 *An encryption scheme* **(Gen, Enc, Dec)** *over a message space* $\mathcal{M}$ *is perfectly secret if and only if for every probability distribution over* $\mathcal{M}$*, every message* $\mathbf{m} \in \mathcal{M}$*, and every ciphertext* $\mathbf{c} \in \mathcal{C}$*:*

$$\mathbf{Pr[C = c \mid M = m] = Pr[C = c]}.$$

LEMMA 2.3 *An encryption scheme* **(Gen, Enc, Dec)** *over a message space* $\mathcal{M}$ *is perfectly secret if and only if for every probability distribution over* $\mathcal{M}$*, every* $\mathbf{m_0}, \mathbf{m_1} \in \mathcal{M}$*, and every* $\mathbf{c} \in \mathcal{C}$*:*

$$\mathbf{Pr[C = c \mid M = m_0] = Pr[C = c \mid M = m_1]}.$$

DEFINITION 2.4 *An encryption scheme* Π **= (Gen, Enc, Dec)** *over a message space* $\mathcal{M}$ *is* perfectly secret *if for every adversary* $\mathbf{A}$ *it holds that*

$$\mathbf{Pr[\ PrivK^{eav}_{A,\Pi} = 1\ ] = 1/2}\ .$$

3.2 Defining Computationally-Secure Encryption

DEFINITION 3.7 A private-key encryption scheme is a tuple of probabilistic polynomial-time algorithms **(Gen, Enc, Dec)** such that:

1/3. The key-generation algorithm **Gen** takes as input the security parameter 1^n and outputs a key k; we write this as $k \leftarrow \mathbf{Gen}(1^n)$ (thus emphasizing the fact that **Gen** is a randomized algorithm). We will assume without loss of generality that any key $k \leftarrow \mathbf{Gen}(1^n)$ satisfies $|k| \leq n$.

Defining Computationally-Secure Encryption

DEFINITION 3.7 A private-key encryption scheme is a tuple of probabilistic polynomial-time algorithms **(Gen, Enc, Dec)** such that:

2/3. The encryption algorithm **Enc** takes as input a key k and a plaintext message $m \in \{0,1\}^*$, and outputs a ciphertext c. Since **Enc** may be randomized, we write $c \leftarrow \mathbf{Enc}_k(m)$.

Defining Computationally-Secure Encryption

DEFINITION 3.7 A private-key encryption scheme is a tuple of probabilistic polynomial-time algorithms **(Gen, Enc, Dec)** such that:

3/3. The decryption algorithm **Dec** takes as input a key k and a ciphertext c, and outputs a message m. We assume that **Dec** is deterministic, and so write this as $m := \mathbf{Dec}_k(c)$.

Defining Computationally-Secure Encryption

- It is required that for every n, every key k output by $\mathbf{Gen}(1^n)$, and every $m \in \{0,1\}^*$, it holds that $\mathbf{Dec}_k(\mathbf{Enc}_k(m)) = m$.

- If **(Gen, Enc, Dec)** is such that for k output by $\mathbf{Gen}(1^n)$, algorithm $\mathbf{Enc}_k$ is only defined for $m \in \{0,1\}^{\ell(n)}$, then we say that **(Gen, Enc, Dec)**

  is a fixed-length private-key encryption scheme for messages of length $\ell(n)$.

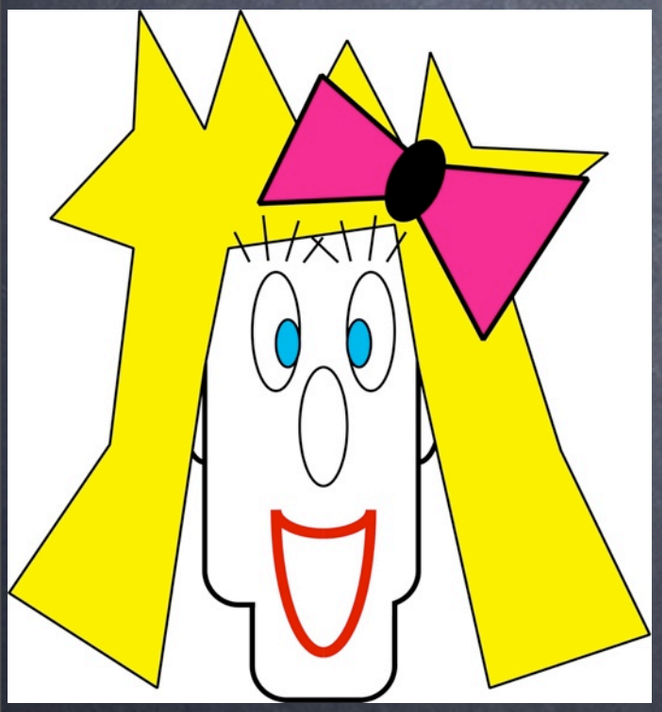
Indistinguishability in the presence of an eavesdropper

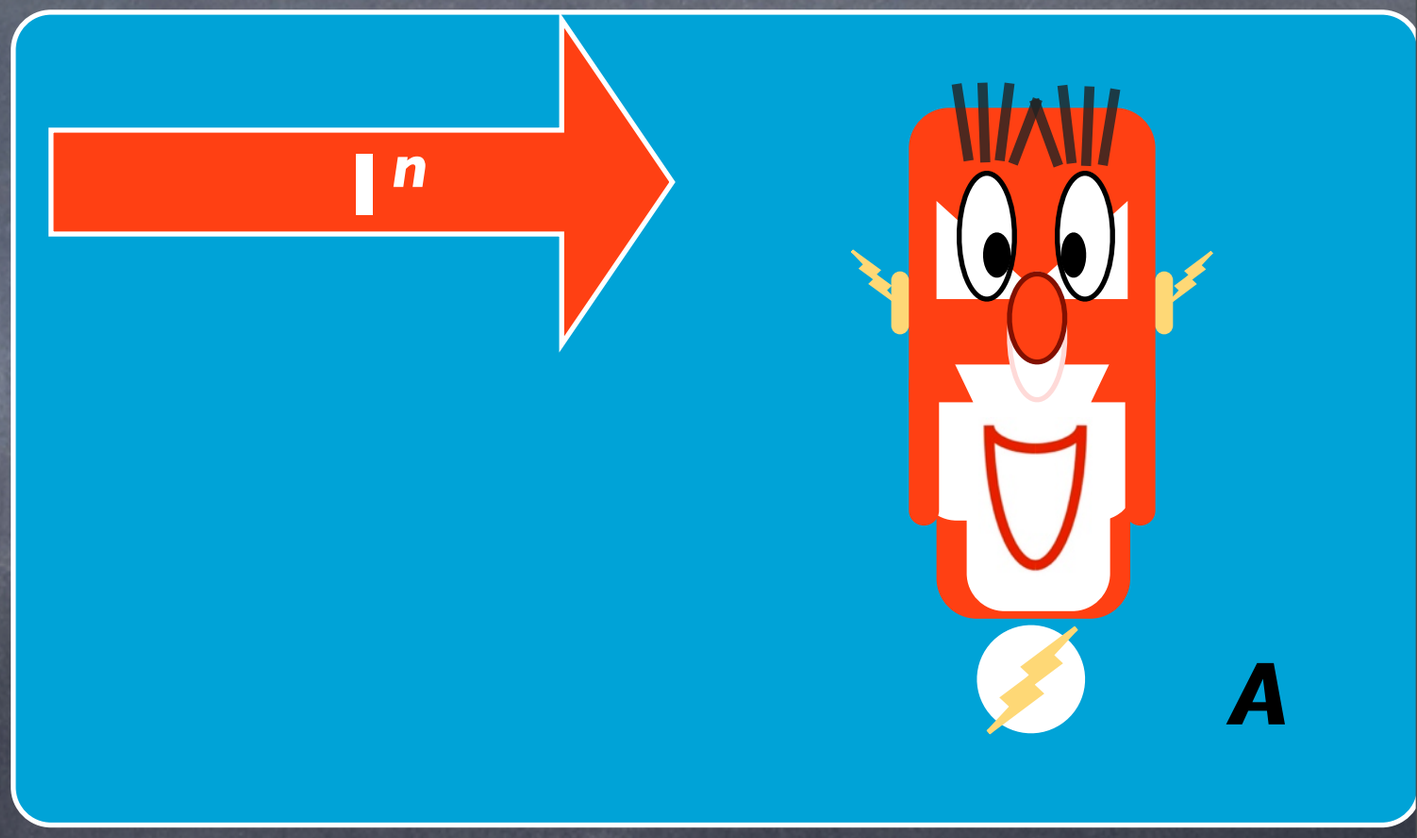
An experiment is defined for any private-key encryption scheme **Π = (Gen, Enc, Dec)**, any **PPT** adversary **A** and any value ***n*** for the security parameter.

The eavesdropping indistinguishability experiment

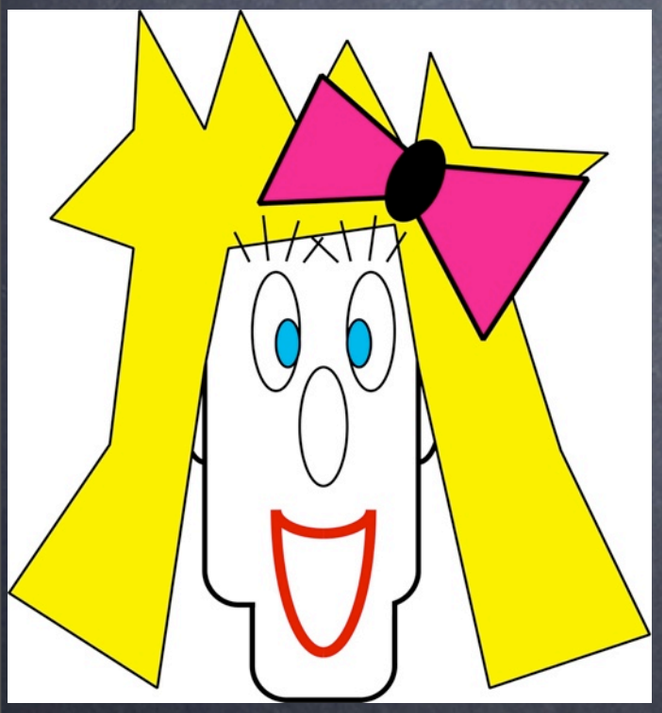
$$\mathbf{PrivK}^{\mathbf{eav}}_{A,\Pi}(n) :$$

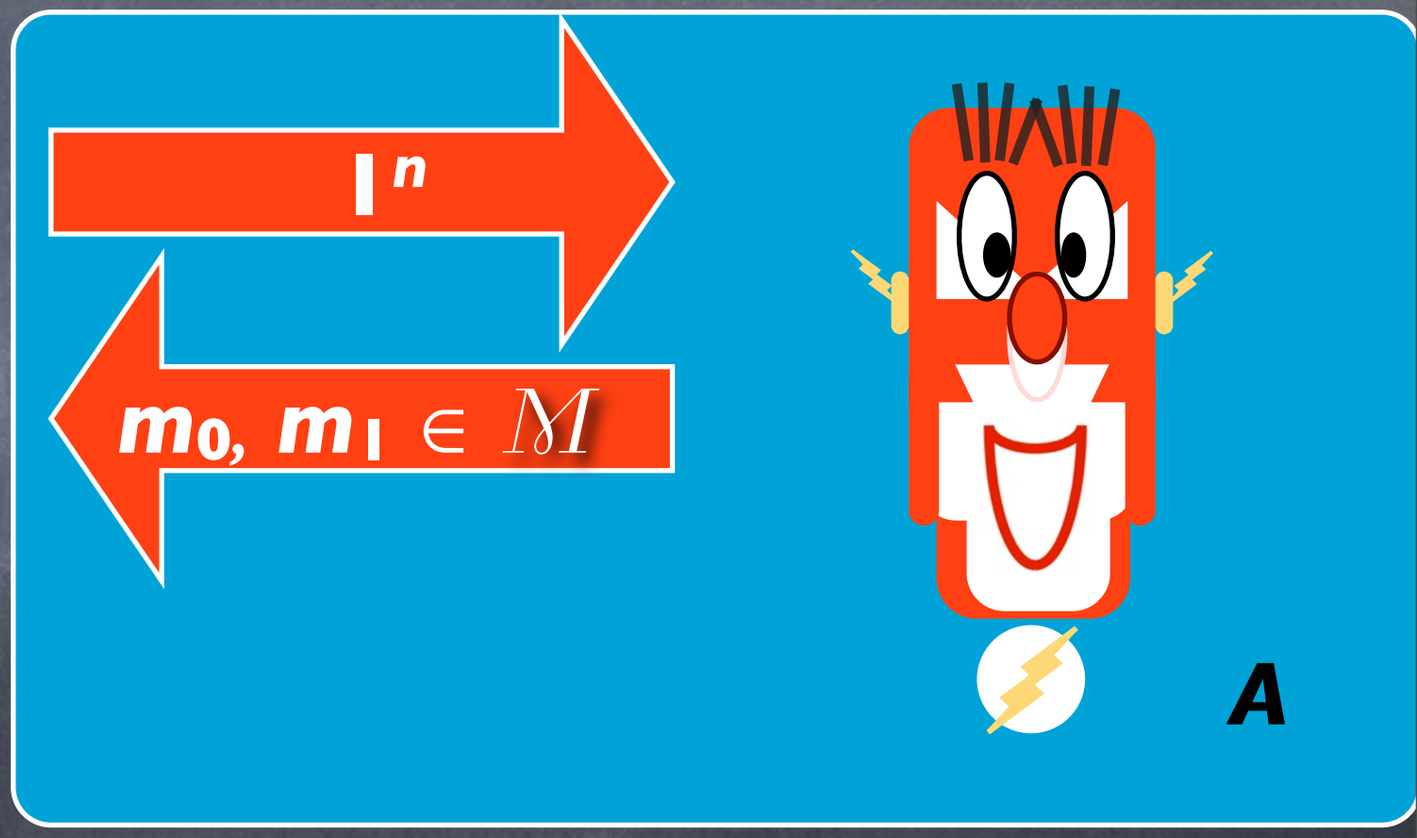
$\mathbf{PrivK}^{eav}_{A,\Pi}$



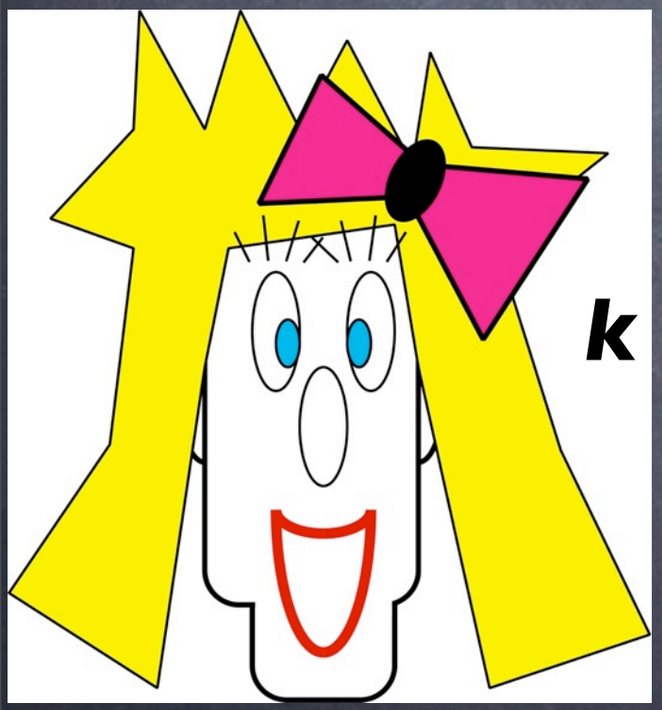


$\mathbf{PrivK}^{eav}_{A,\Pi}$





$\mathbf{PrivK}^{eav}_{A,\Pi}$



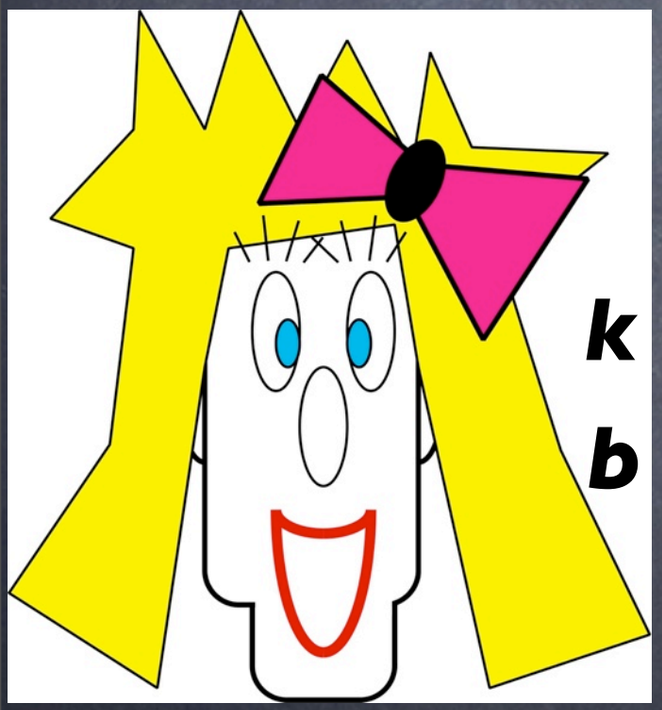
$k \leftarrow \mathbf{Gen}(1^n)$

1^n

$m_0, m_1 \in \mathcal{M}$

A

$\mathbf{PrivK}^{eav}_{A,\Pi}$



$k \leftarrow \mathbf{Gen}(1^n)$

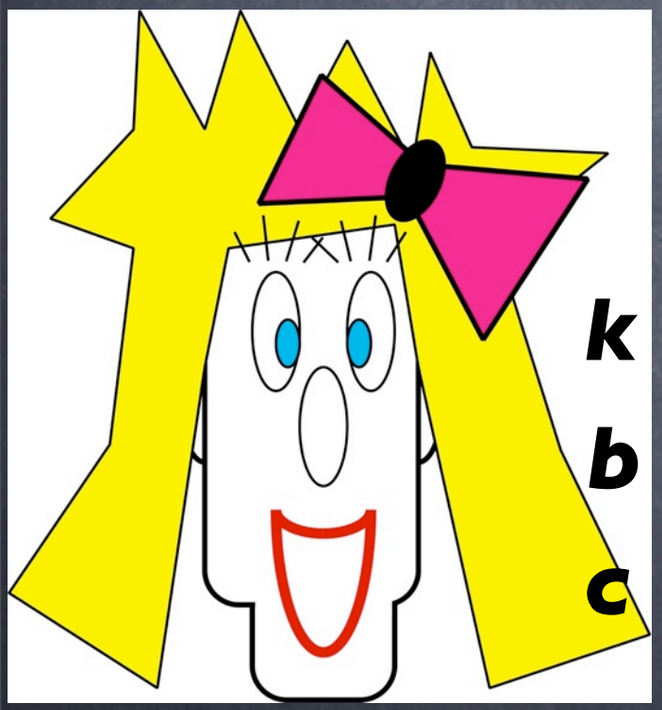
$b \leftarrow \{0, 1\}$

1^n

$m_0, m_1 \in \mathcal{M}$

A

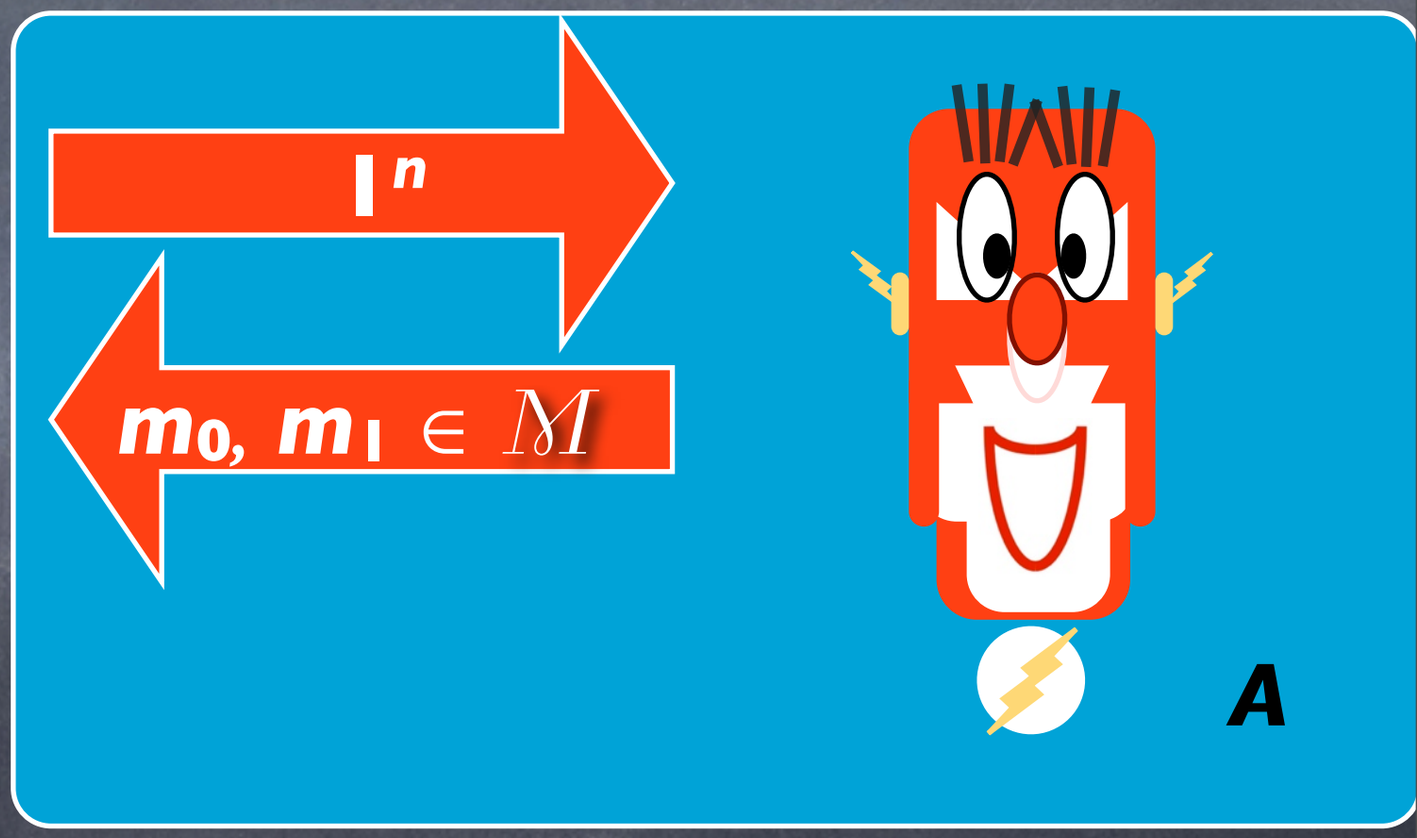
$\mathbf{PrivK}^{eav}_{A,\Pi}$



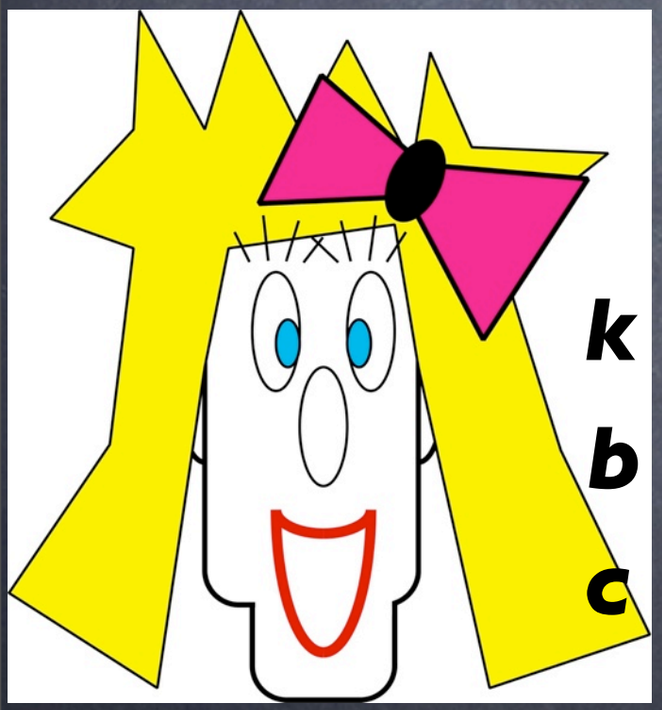
$k \leftarrow \mathbf{Gen}(1^n)$

$b \leftarrow \{0, 1\}$

$c \leftarrow \mathbf{Enc}_k(m_b)$



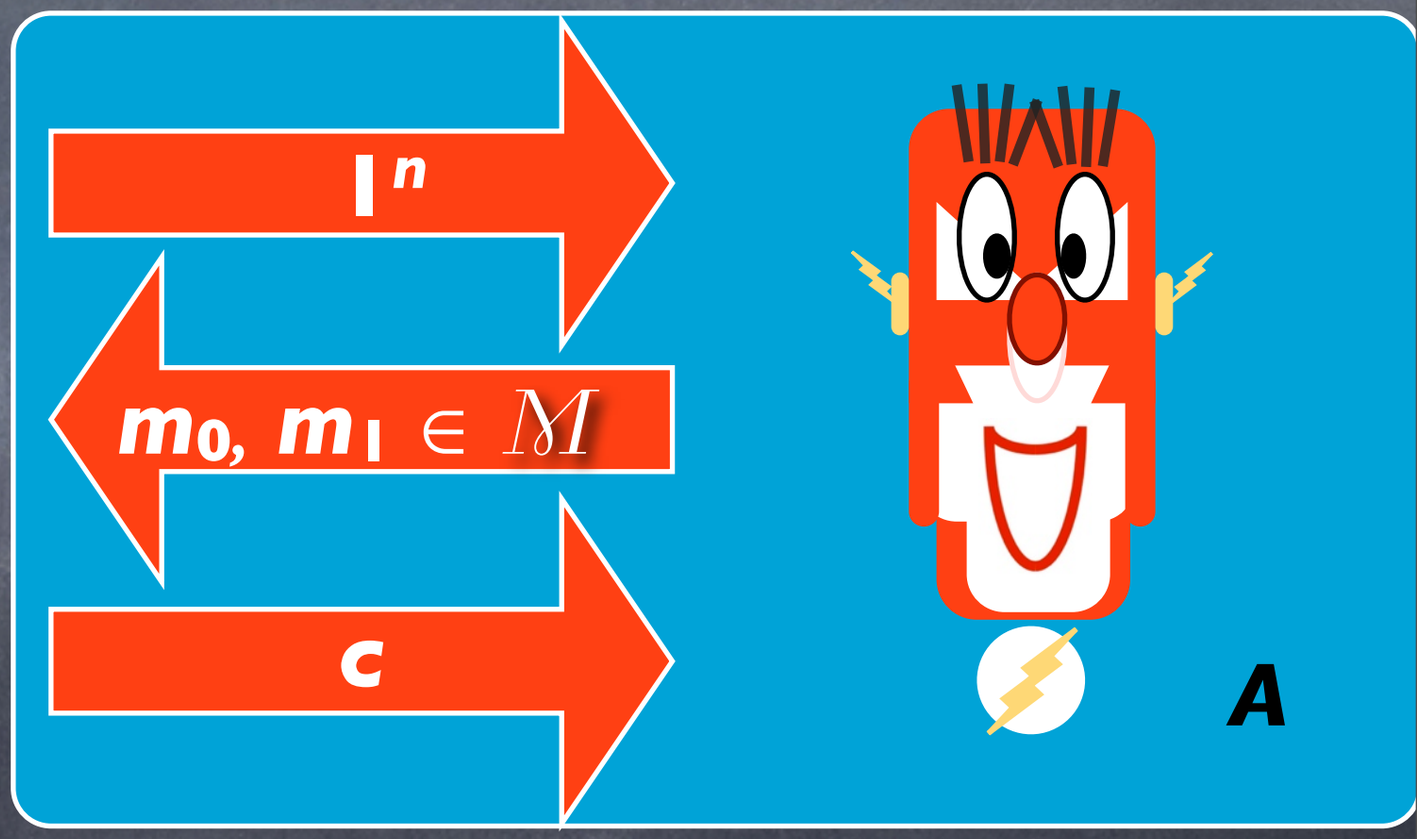
$\mathbf{PrivK}^{eav}_{A,\Pi}$



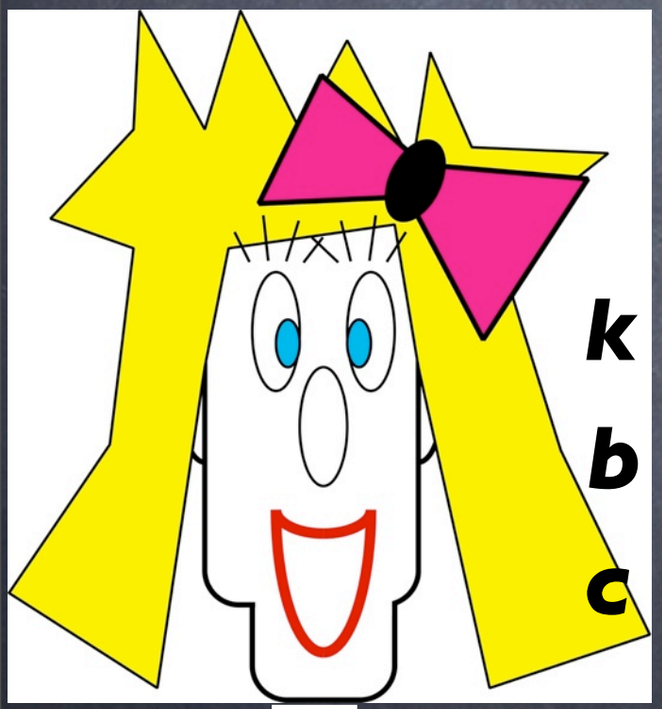
$k \leftarrow \mathbf{Gen}(1^n)$

$b \leftarrow \{0, 1\}$

$c \leftarrow \mathbf{Enc}_k(m_b)$



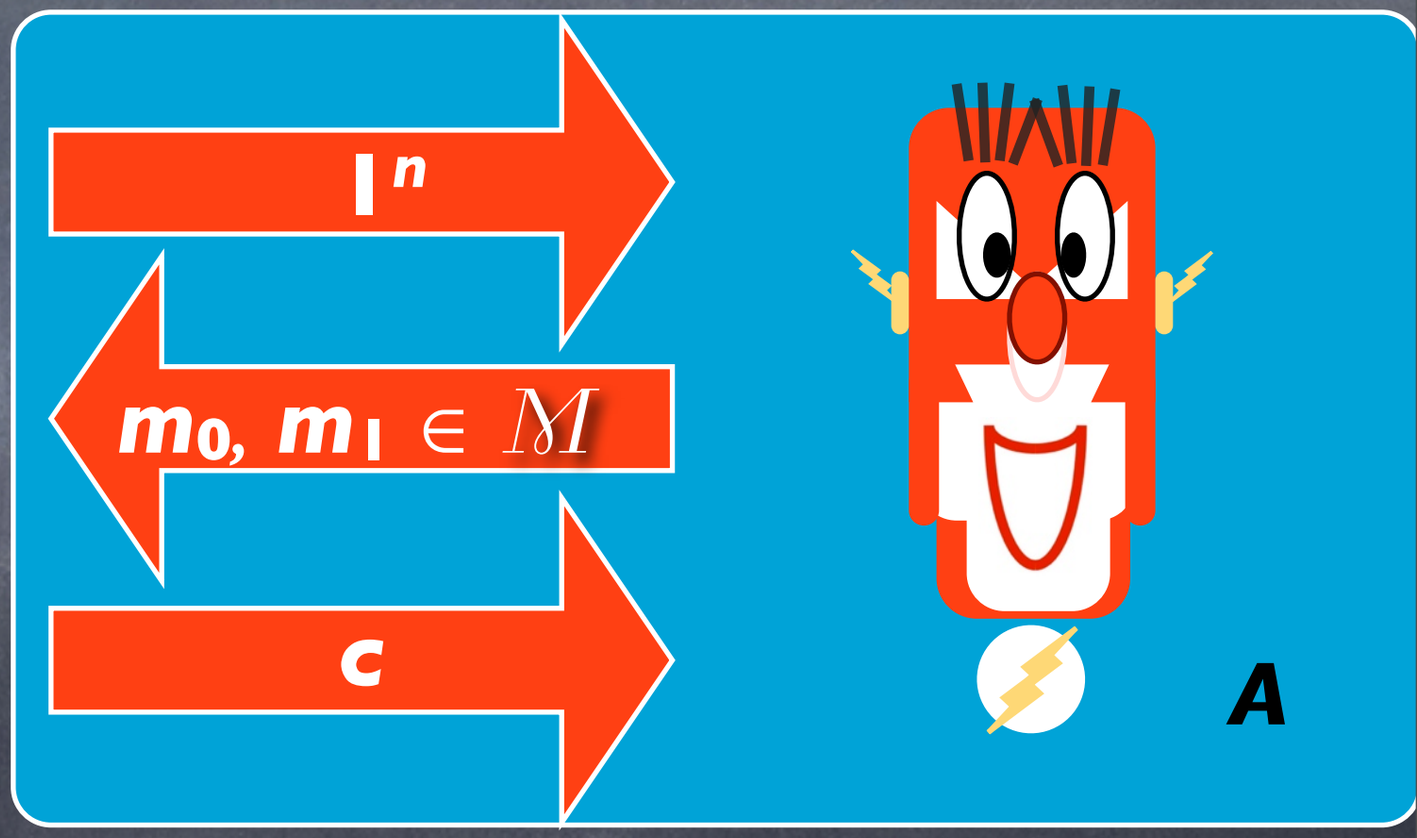
$\mathbf{PrivK}^{eav}_{A,\Pi}$



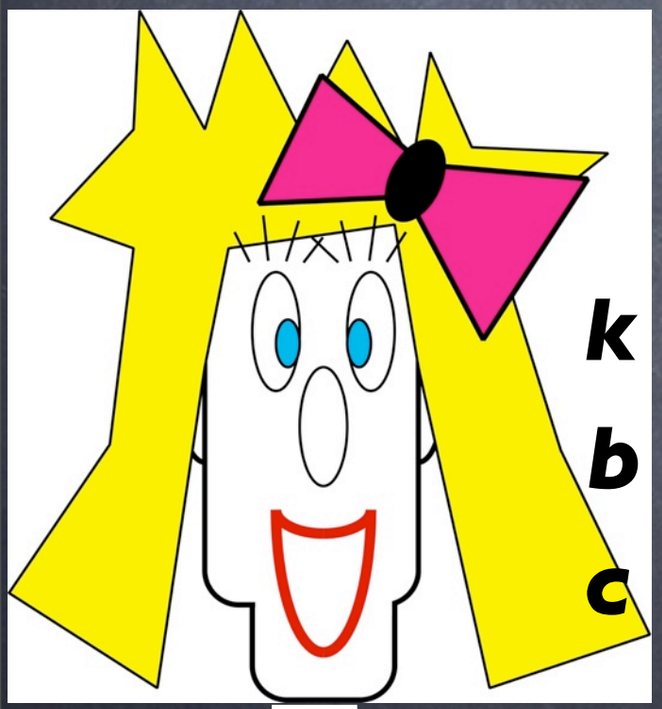
$k \leftarrow \mathbf{Gen}(1^n)$

$b \leftarrow \{0, 1\}$

$c \leftarrow \mathbf{Enc}_k(m_b)$



$\mathbf{PrivK}^{\mathrm{eav}}_{A,\Pi}$



$k \leftarrow \mathrm{Gen}(1^n)$

$b \leftarrow \{0, 1\}$

$c \leftarrow \mathrm{Enc}_k(m_b)$

b

n

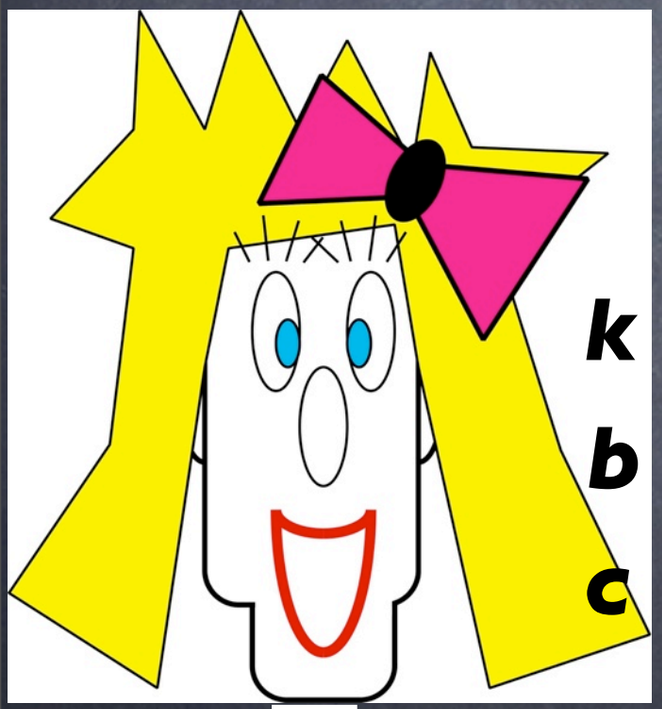
$m_0, m_1 \in \mathcal{M}$

c

A

b'

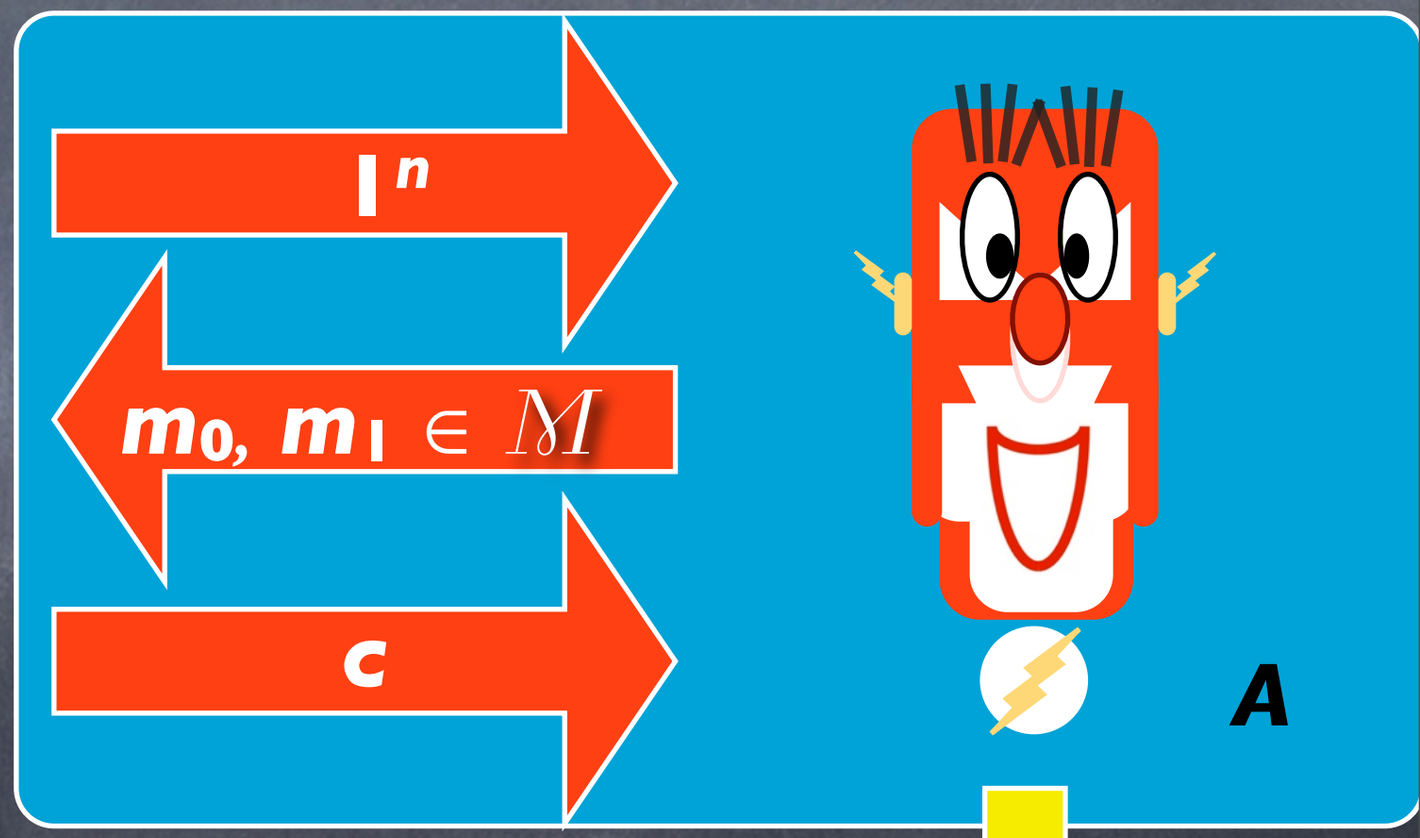
$\mathbf{PrivK}^{eav}_{A,\Pi}$



$k \leftarrow \mathbf{Gen}(1^n)$

$b \leftarrow \{0, 1\}$

$c \leftarrow \mathbf{Enc}_k(m_b)$

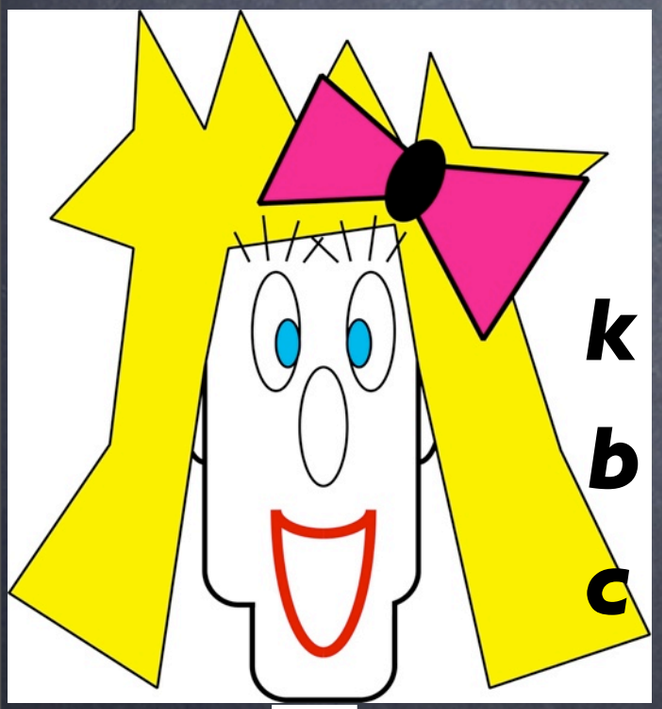


b

b'

$\Pr[\, b = b' \,] \leq \frac{1}{2} + \mathrm{negl}(n)$

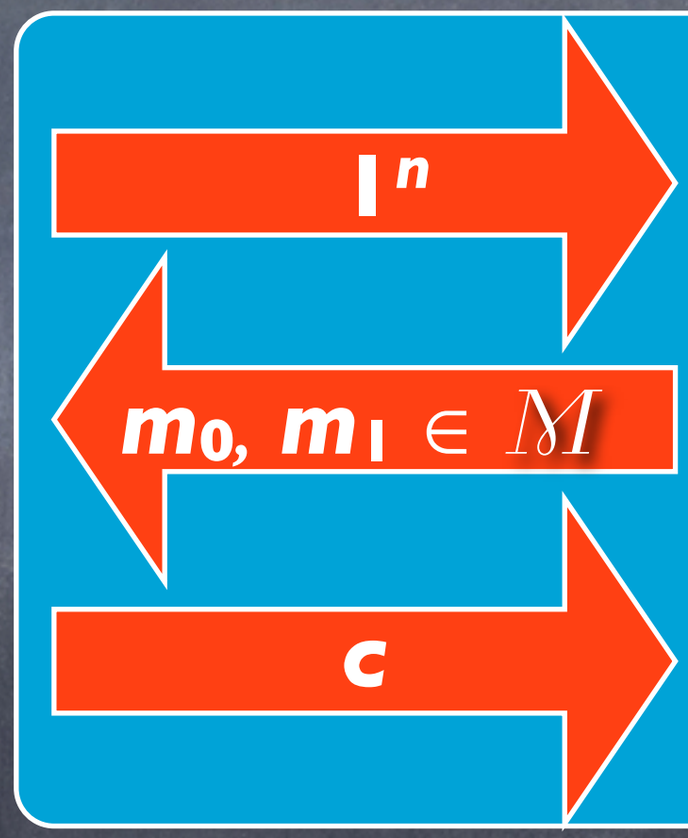
$\mathbf{PrivK}^{eav}_{A,\Pi}$

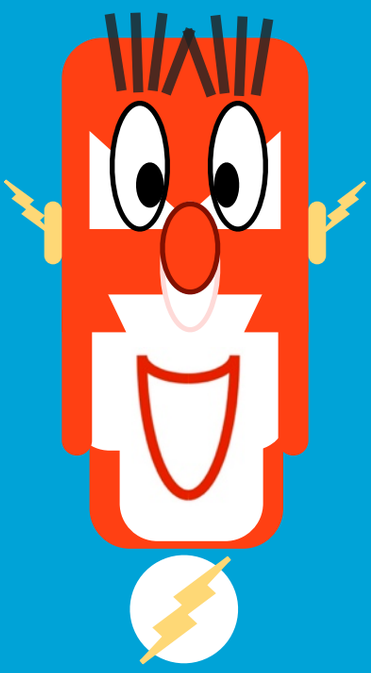


$k \leftarrow \mathbf{Gen}(1^n)$

$b \leftarrow \{0, 1\}$

$c \leftarrow \mathbf{Enc}_k(m_b)$





A

b

b′

computationally secret

$\Pr[\, b = b' \,] \leq \frac{1}{2} + \mathrm{negl}(n)$

$\mathbf{PrivK}^{eav}_{A,\Pi}(n)$

1. The adversary **A** is given input 1^n, and outputs a pair of messages m_0 , m_1 of the same length.

2. A key k is generated by running $\mathbf{Gen}(1^n)$, and a random bit $b \leftarrow \{0,1\}$ is chosen. A (challenge) ciphertext $c \leftarrow \mathbf{Enc}_k(m_b)$ is computed and given to **A**.

3. **A** outputs a bit b'.

4. The output of the experiment is defined to be **1** if $b' = b$, and **0** otherwise.
(If $\mathbf{PrivK}^{eav}_{A,\Pi}(n) = 1$, we say that **A** succeeded.)

$\mathbf{PrivK}^{\mathbf{eav}}_{A,\Pi}(n)$

- If Π is a fixed-length scheme for messages of length $\ell(n)$, the previous experiment is modified by requiring $m_0, m_1 \in \{0,1\}^{\ell(n)}$.

Defining Computationally-Secure Encryption

DEFINITION 3.8 A private-key encryption scheme **Π = (Gen, Enc, Dec)** has indistinguishable encryptions in the presence of an eavesdropper if for all **PPT** adversaries **A** there exists a negligible function **negl** such that

$$\mathbf{Pr}[\,\mathbf{PrivK}^{\mathbf{eav}}_{\mathbf{A},\Pi}(n) = 1\,] \leq \tfrac{1}{2} + \mathbf{negl}(n),$$

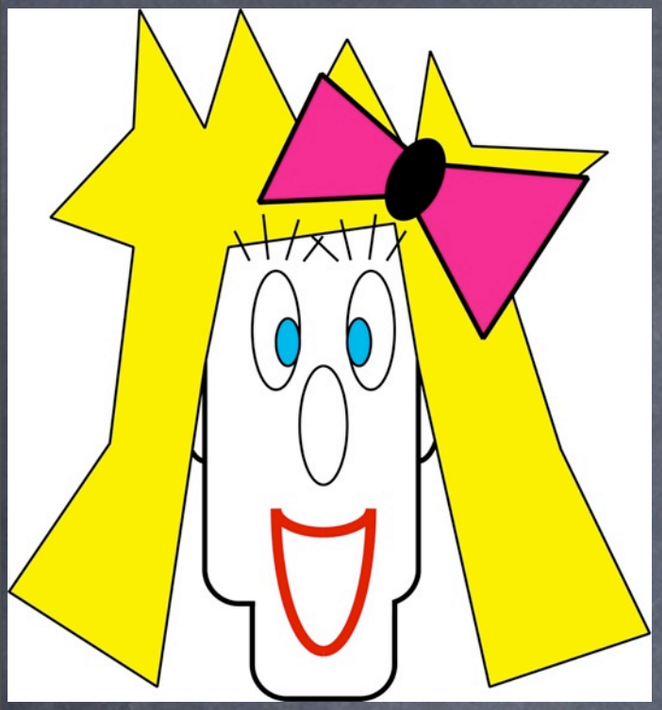
where the probability is taken over the random coins used by **A**, as well as the random coins used in the experiment (for choosing the key, the random bit **b**, and any random coins used in the encryption process).

3.2.2* Properties of the Definition

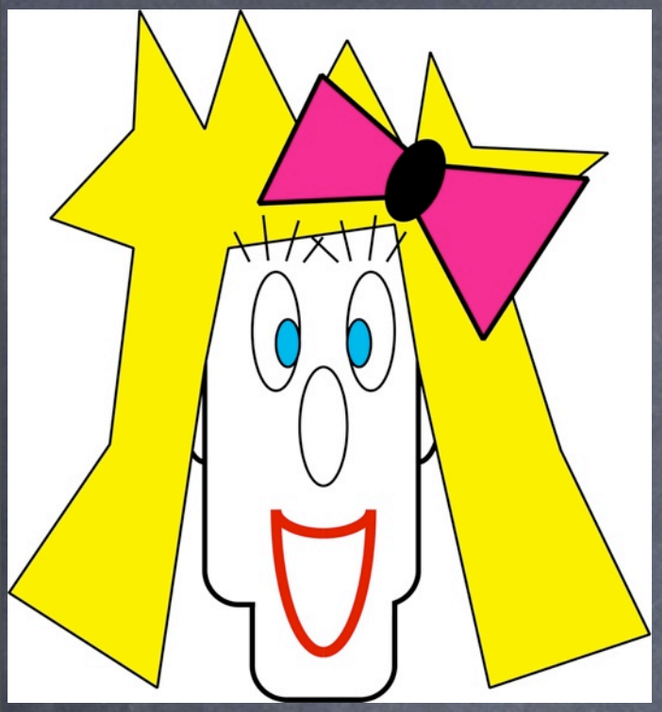
DEFINITION 3.12 A private-key encryption scheme **(Gen, Enc, Dec)** is *semantically* secure in the presence of an eavesdropper if for every **PPT** algorithm **A** there exists a **PPT** algorithm **A′** such that for all efficiently-sampleable distributions $\boldsymbol{X} = (\boldsymbol{X}_1,\ldots)$ and all polynomial-time computable functions **f** and **h**, there exists a negligible function **negl** s.t.

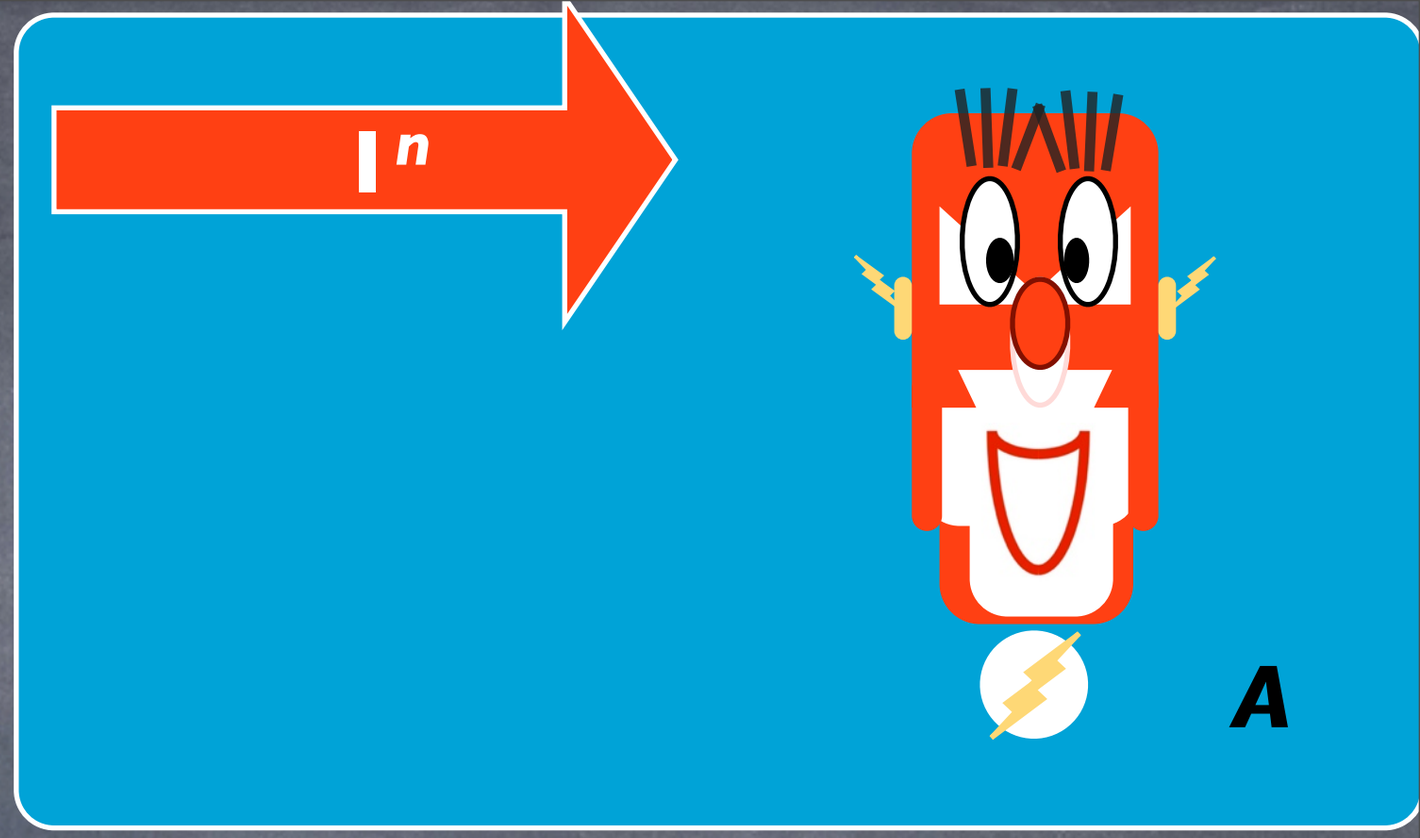
$$|\Pr[\mathbf{A}(\mathbf{1}^n, \mathbf{Enc}_k(m), h(m)) = f(m)] - \Pr[\mathbf{A}'(\mathbf{1}^n, h(m)) = f(m)]| \leq \mathbf{negl}(n),$$

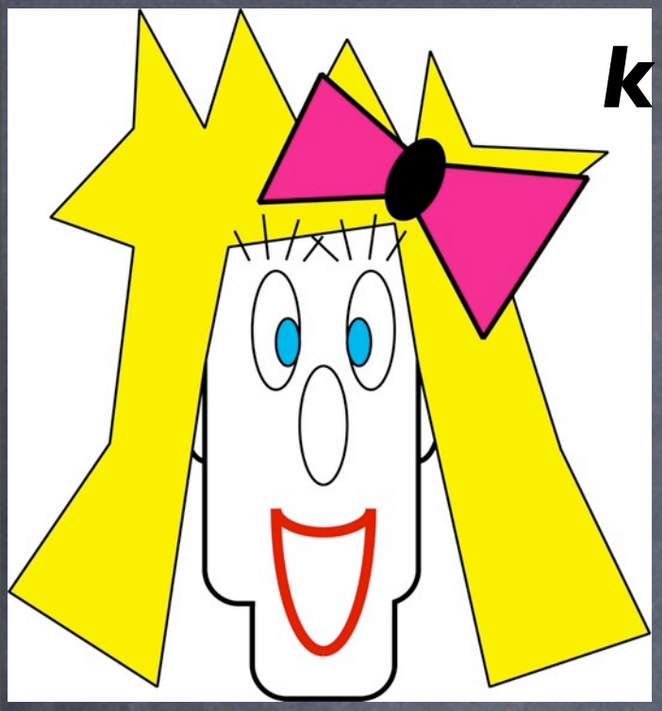
where m is chosen according to distribution $\boldsymbol{X}_n$, and the probabilities are taken over the choice of m and the key k, and any random coins used by **A, A′**, and the encryption process.



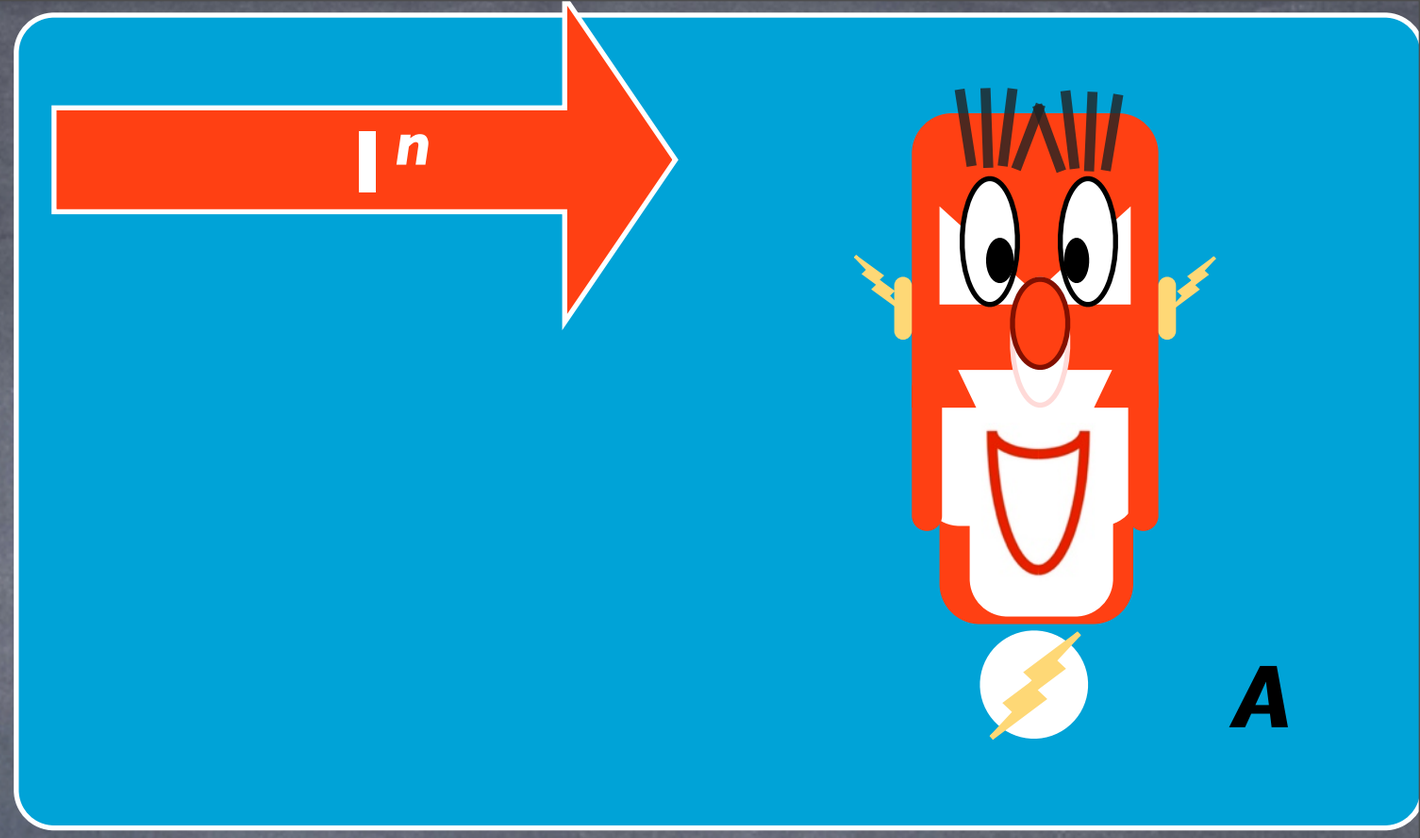
A

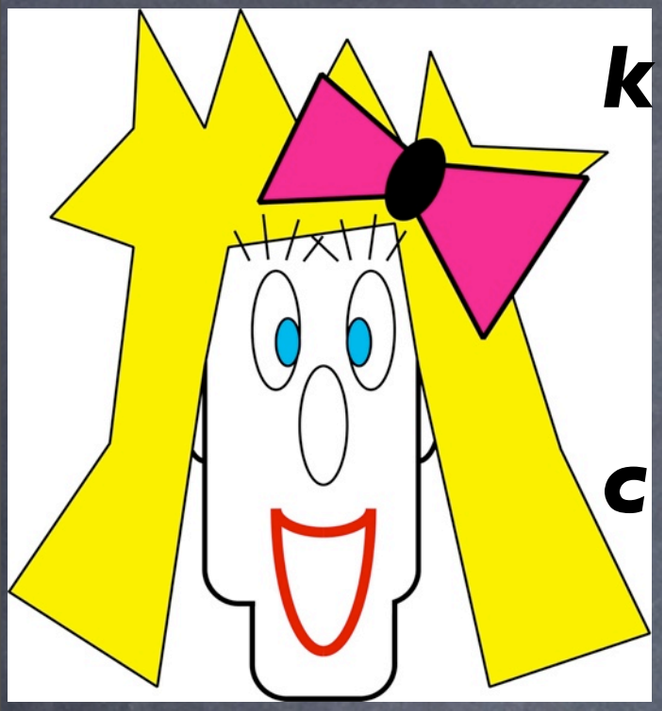



n
A



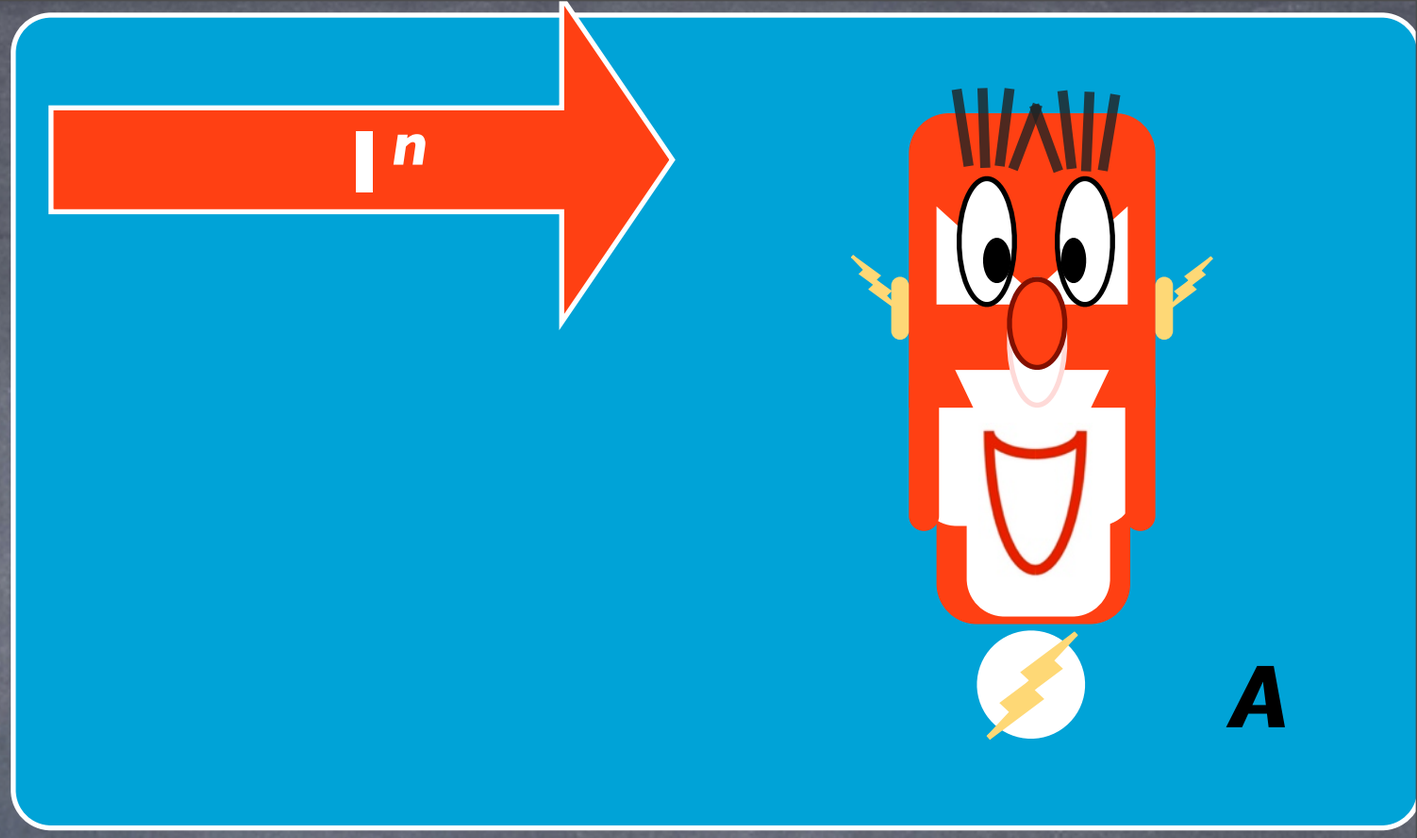
$k \leftarrow \mathrm{Gen}(1^n)$

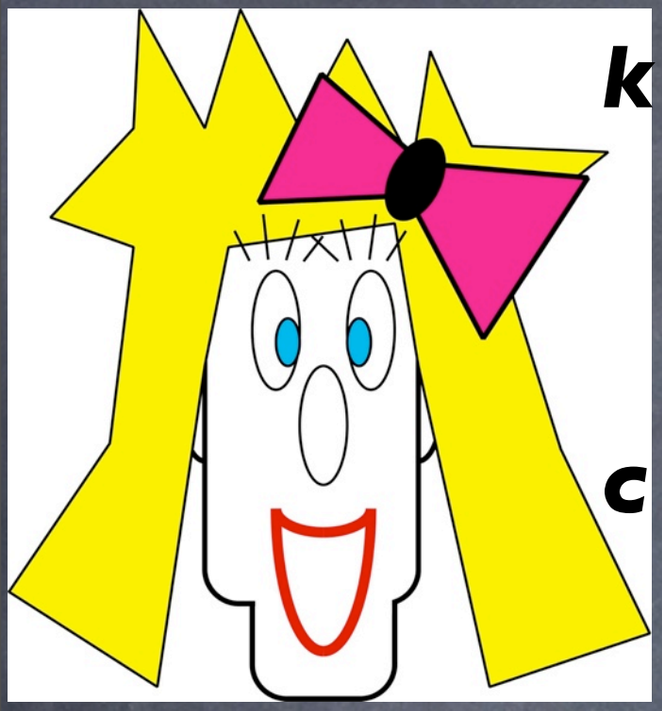




$k \leftarrow \mathrm{Gen}(1^n)$

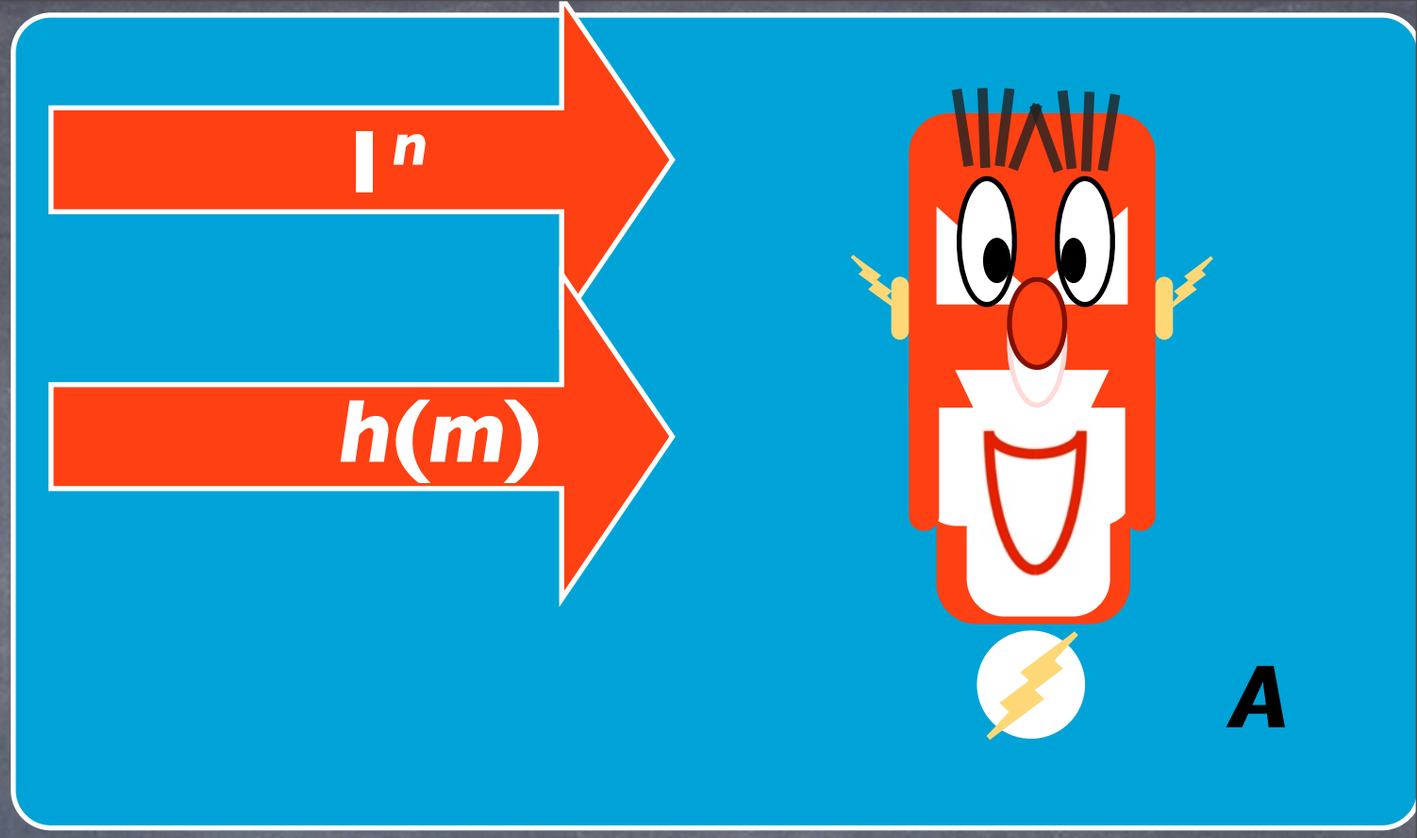
$c \leftarrow \mathrm{Enc}_k(m)$

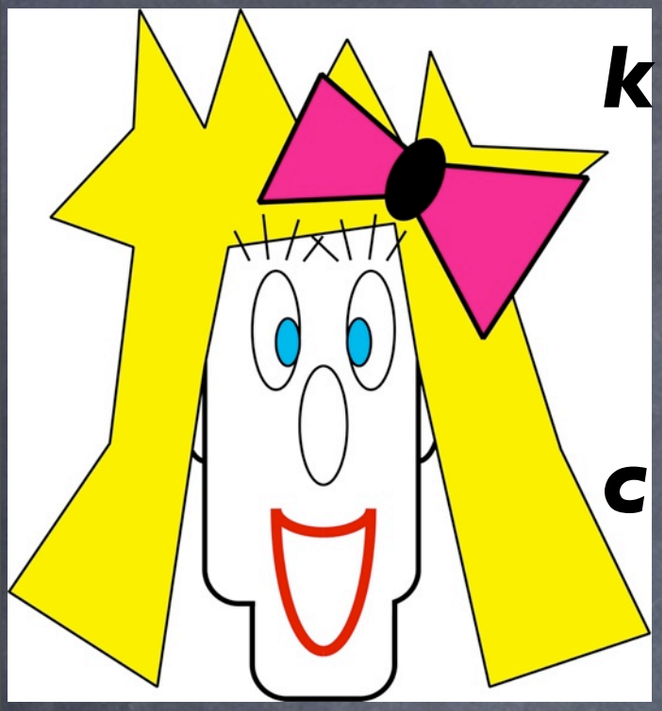




$k \leftarrow \mathsf{Gen}(1^n)$

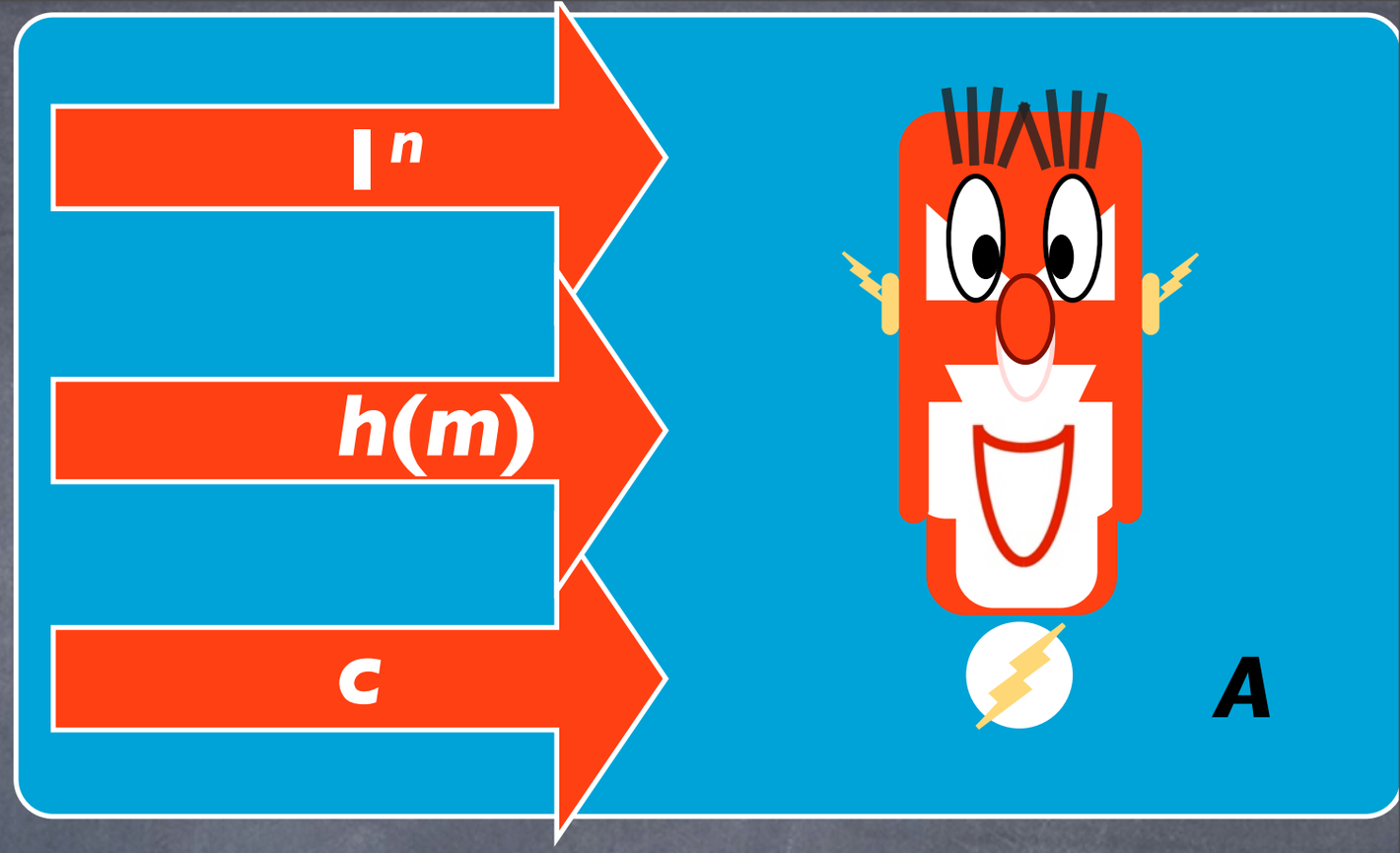
$c \leftarrow \mathsf{Enc}_k(m)$

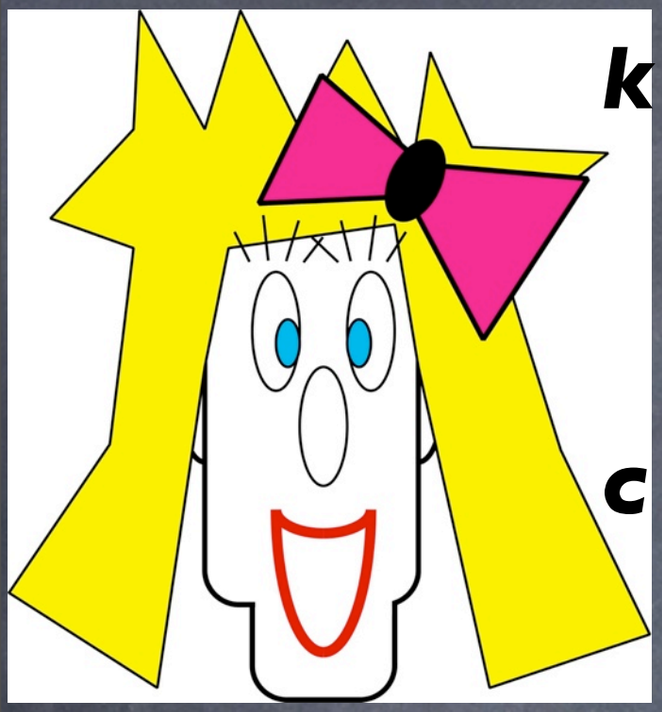




$k \leftarrow \mathrm{Gen}(1^n)$

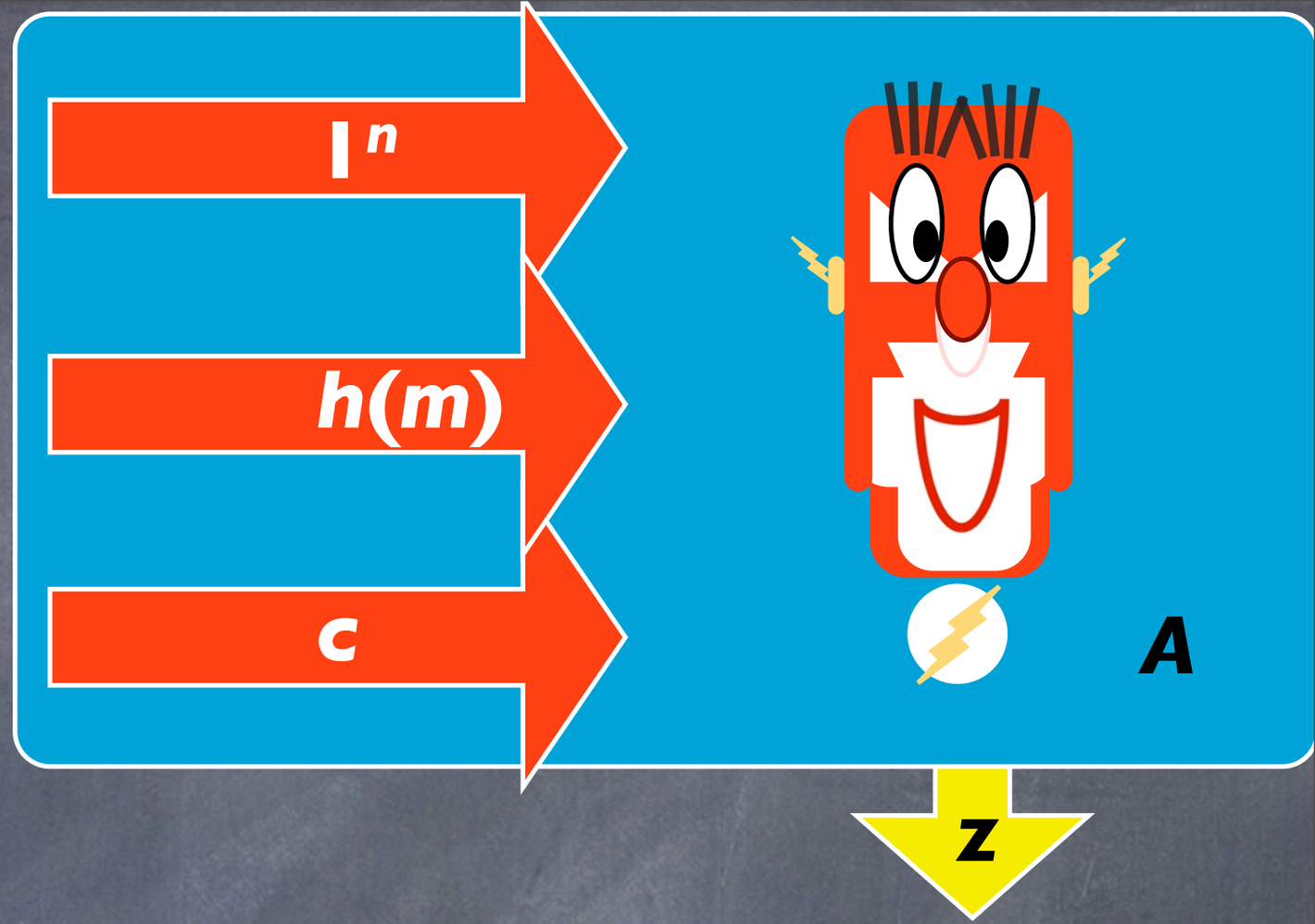
$c \leftarrow \mathrm{Enc}_k(m)$

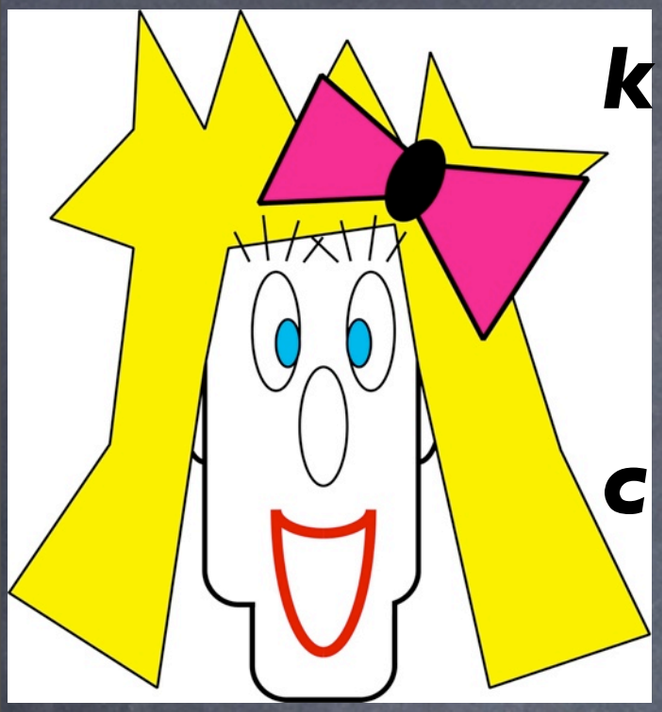




$k \leftarrow \mathrm{Gen}(1^n)$

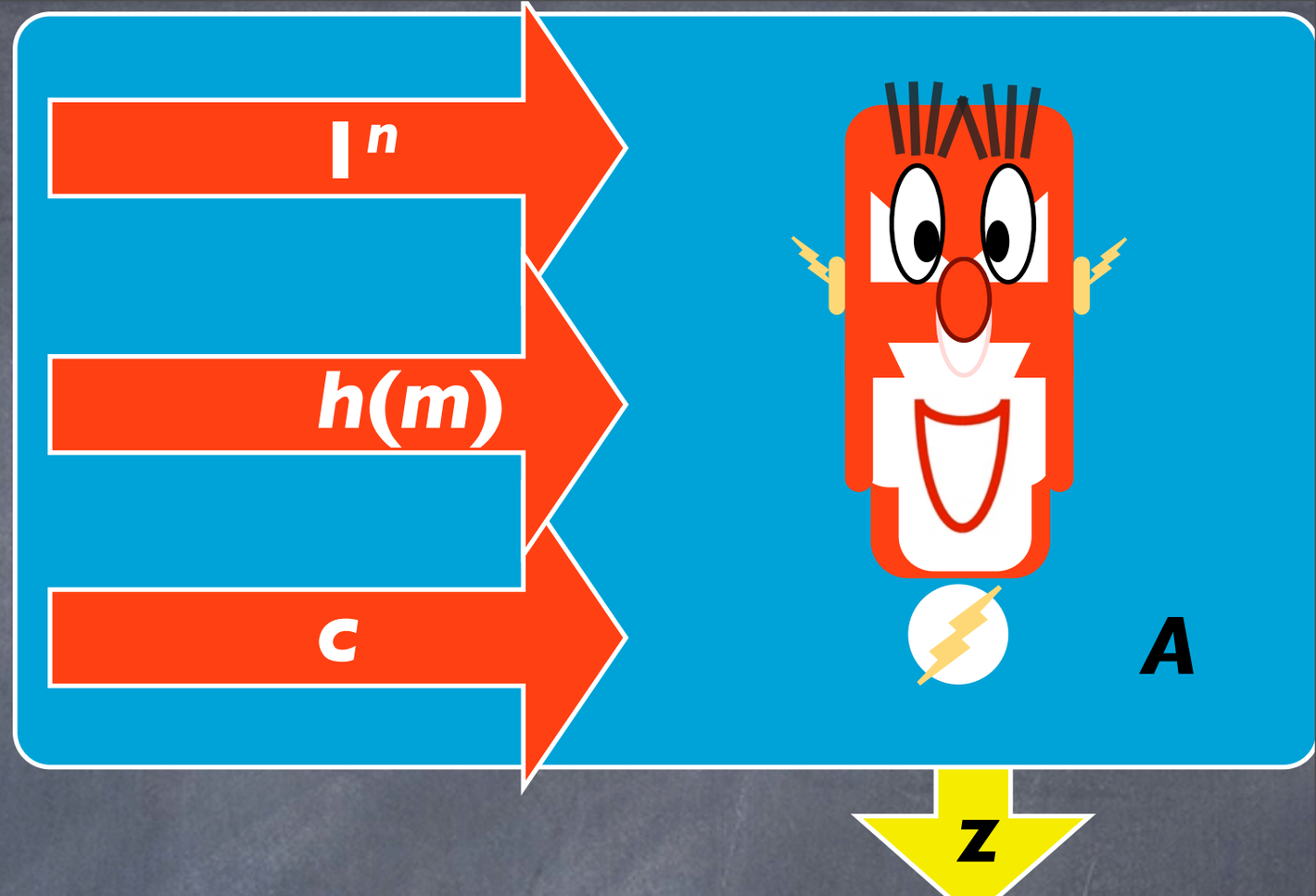
$c \leftarrow \mathrm{Enc}_k(m)$

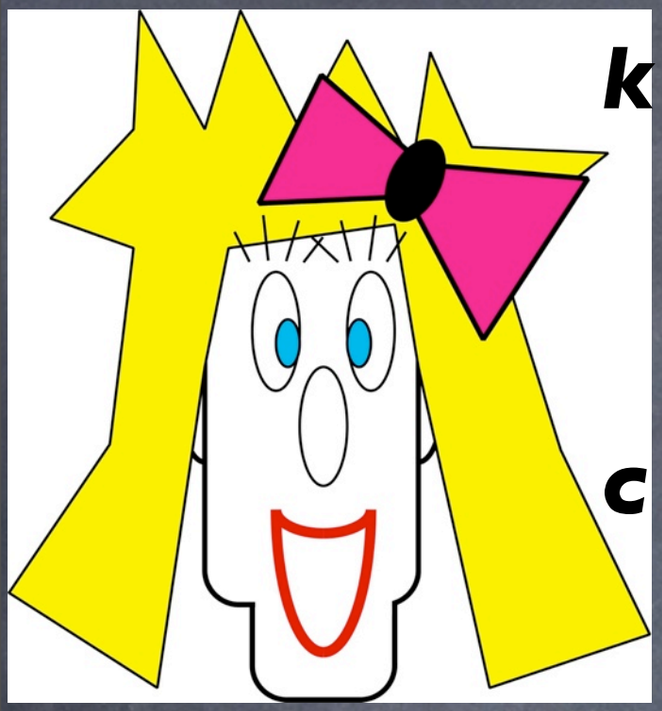




$k \leftarrow \mathrm{Gen}(1^n)$

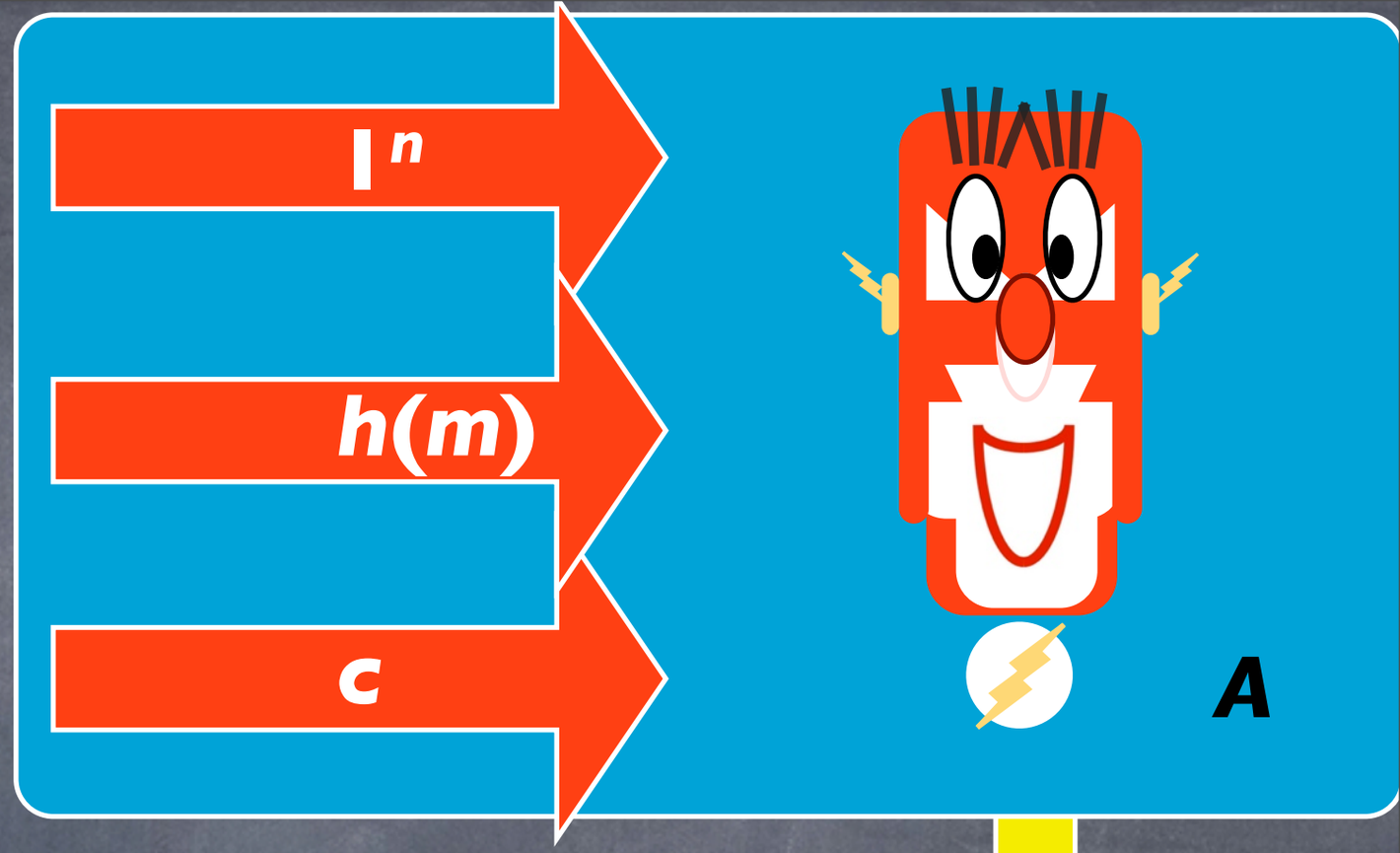
$c \leftarrow \mathrm{Enc}_k(m)$

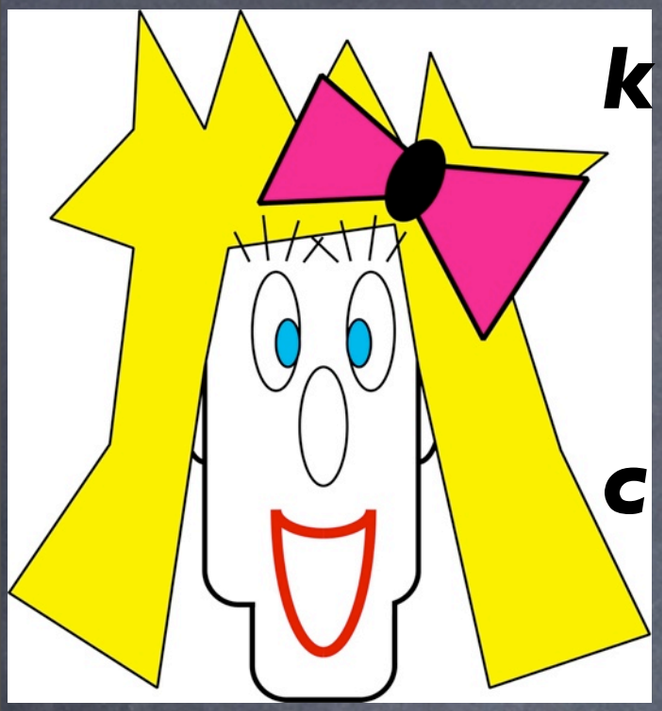




$k \leftarrow \mathsf{Gen}(1^n)$

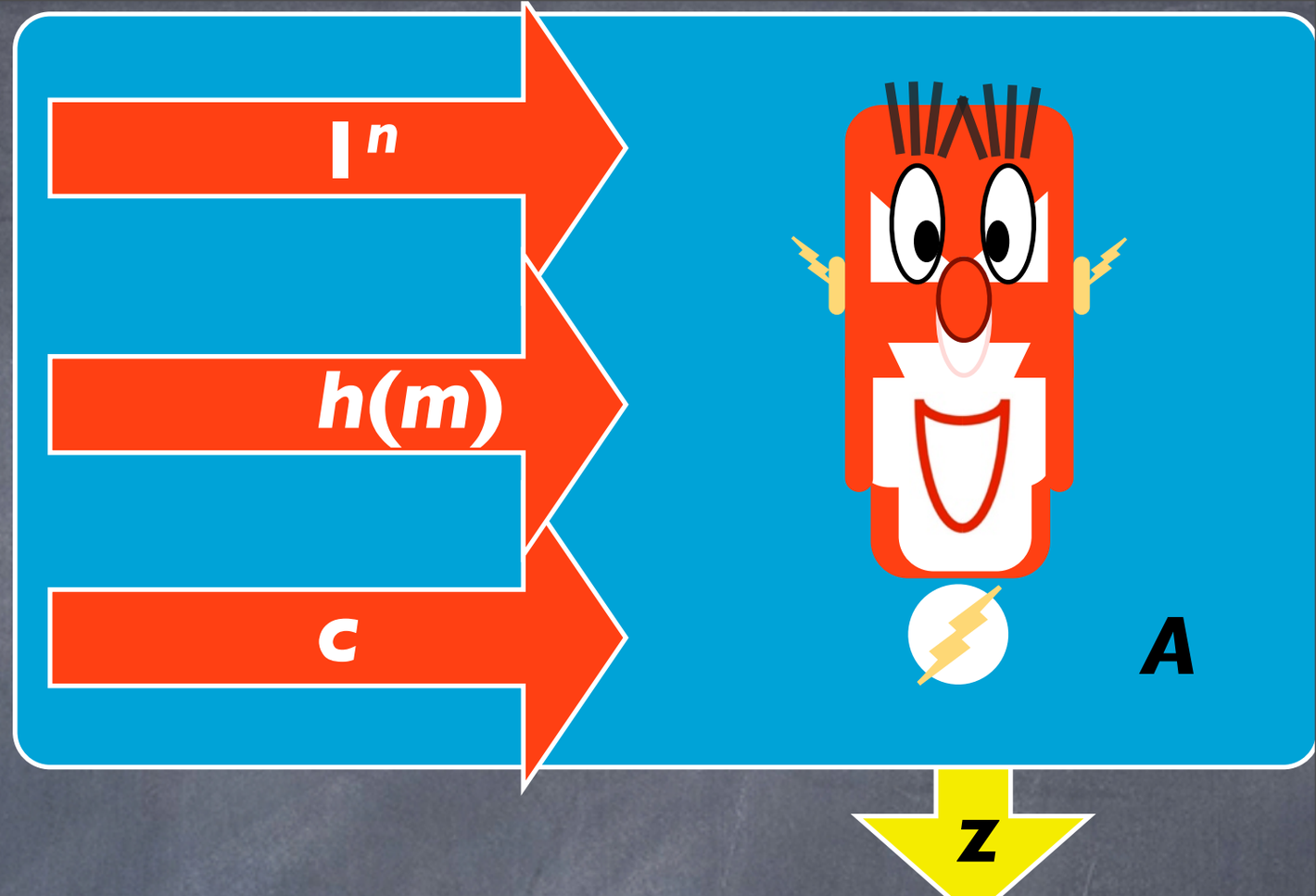
$c \leftarrow \mathsf{Enc}_k(m)$

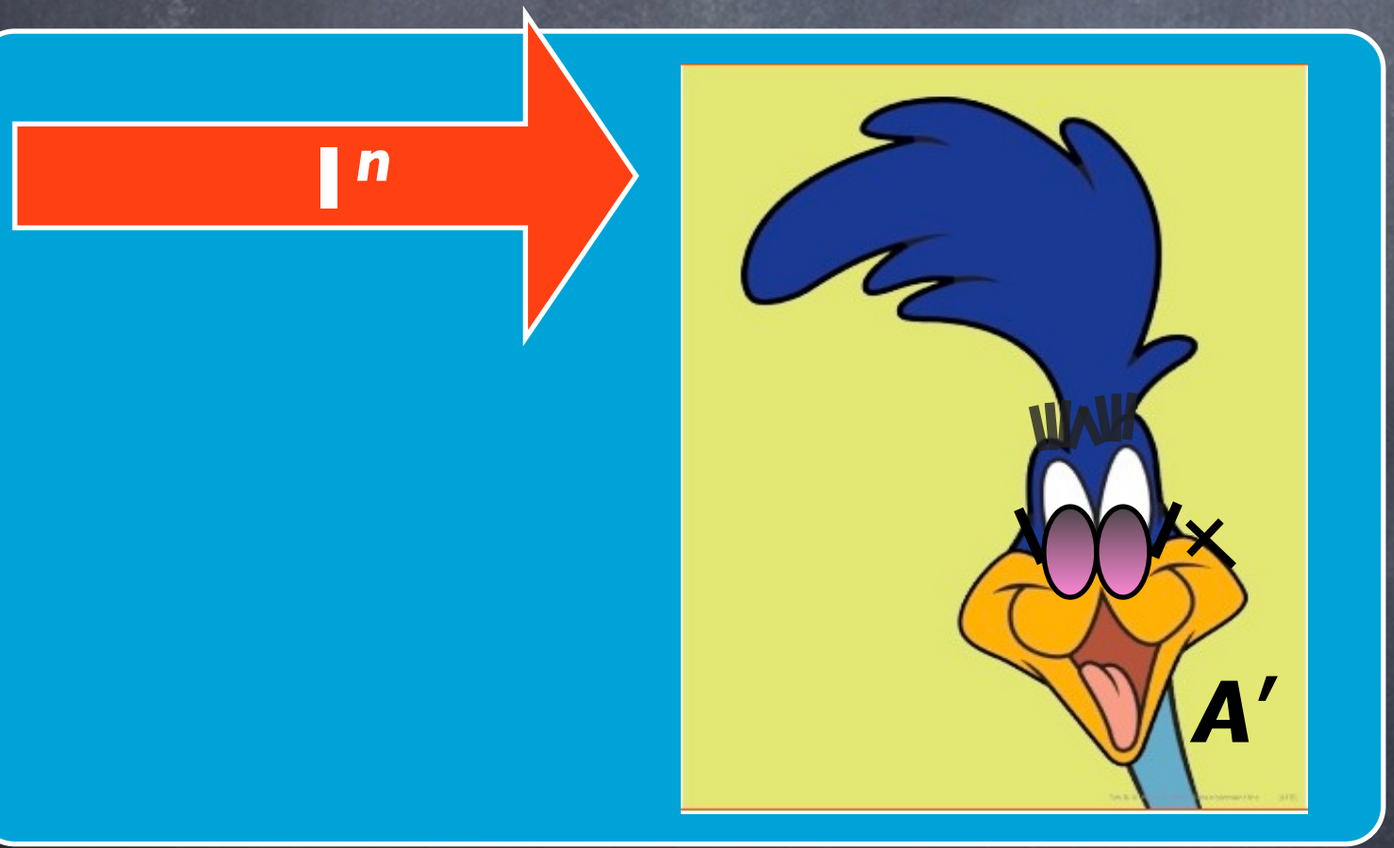


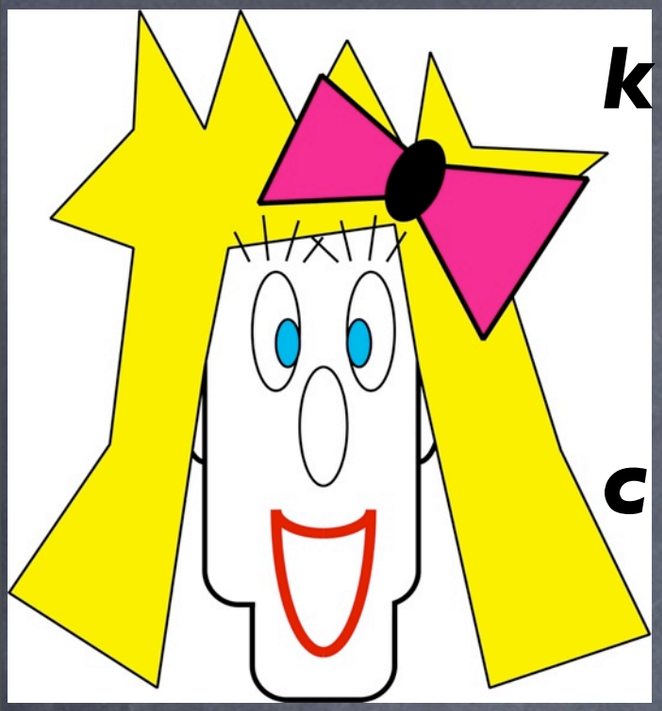


$k \leftarrow \mathrm{Gen}(1^n)$

$c \leftarrow \mathrm{Enc}_k(m)$

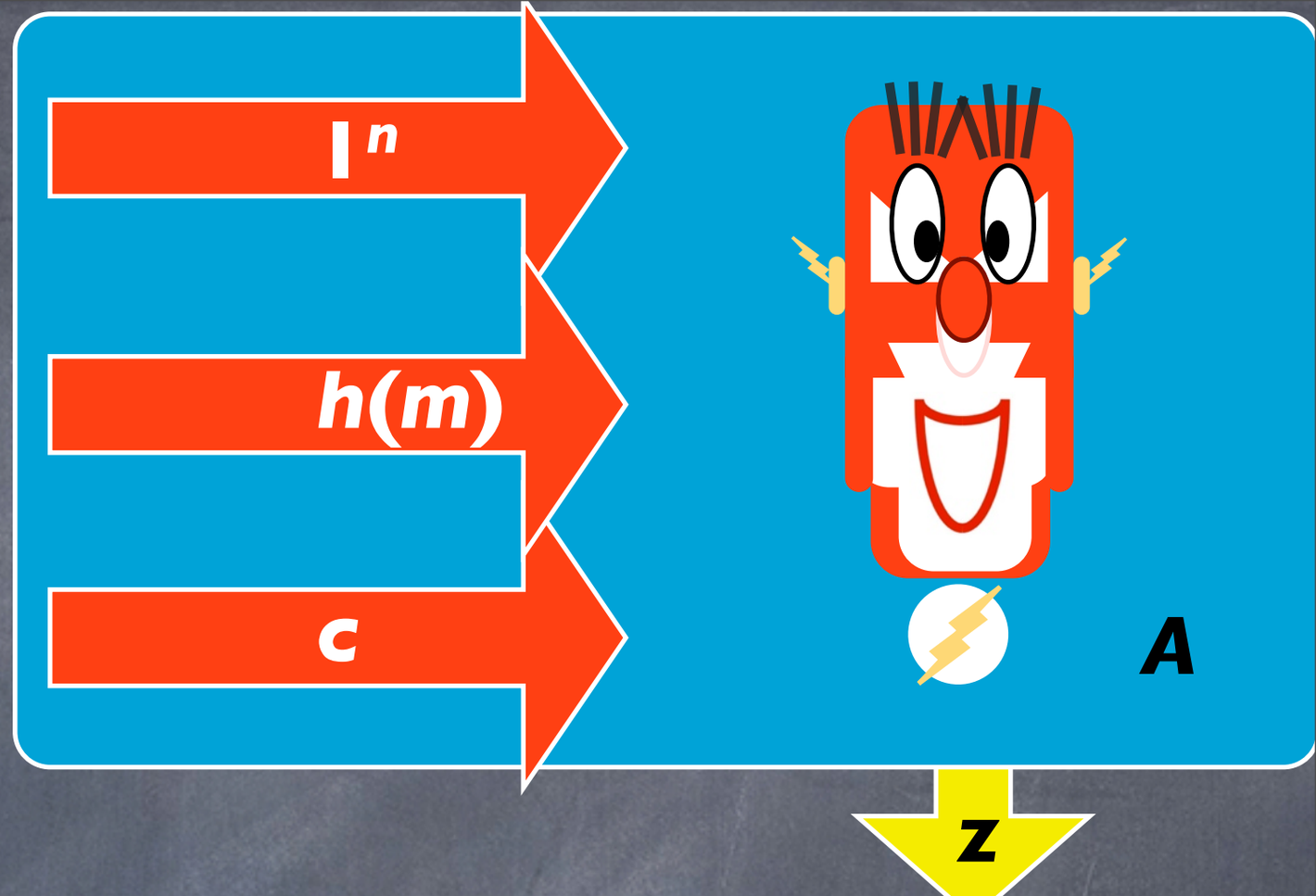


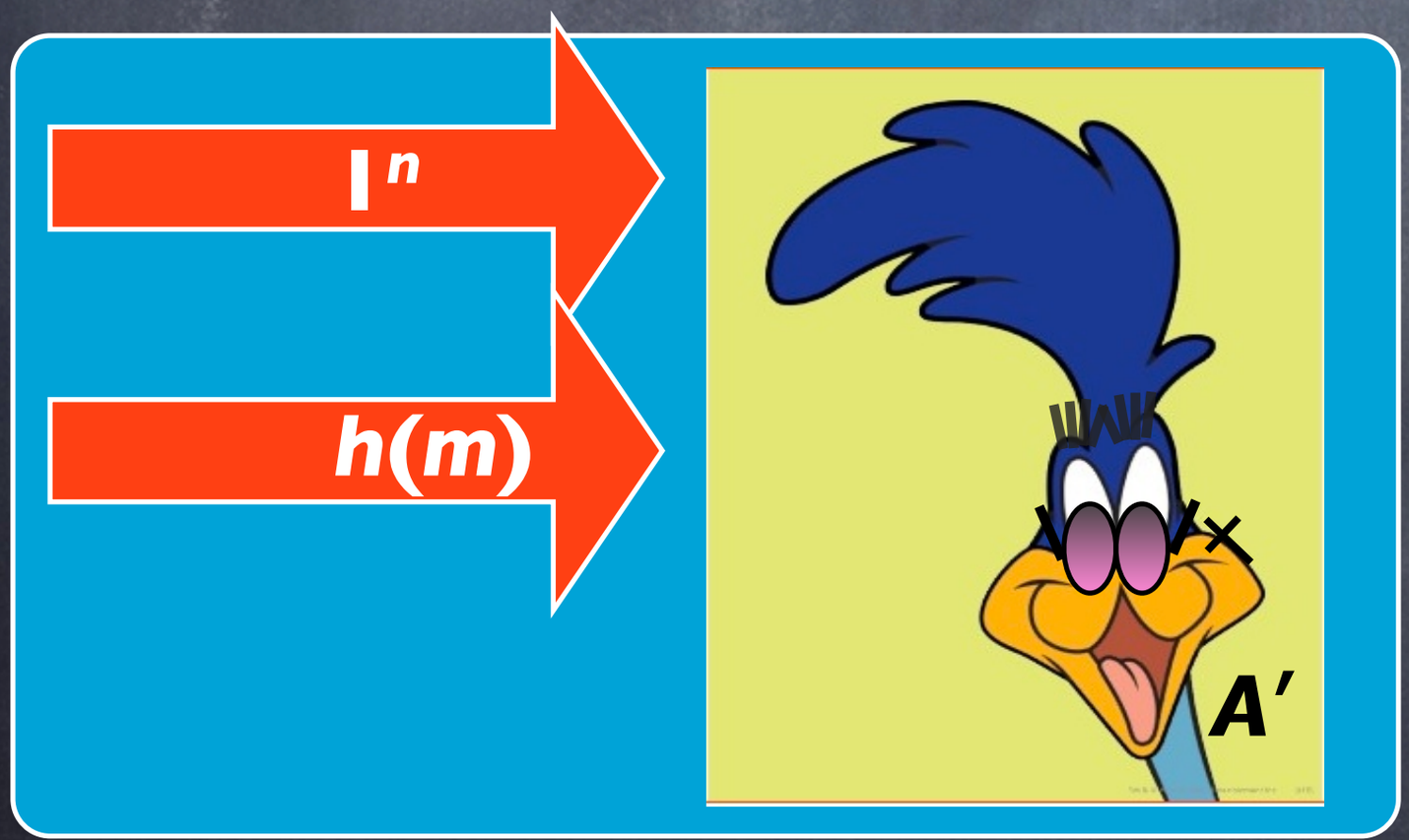


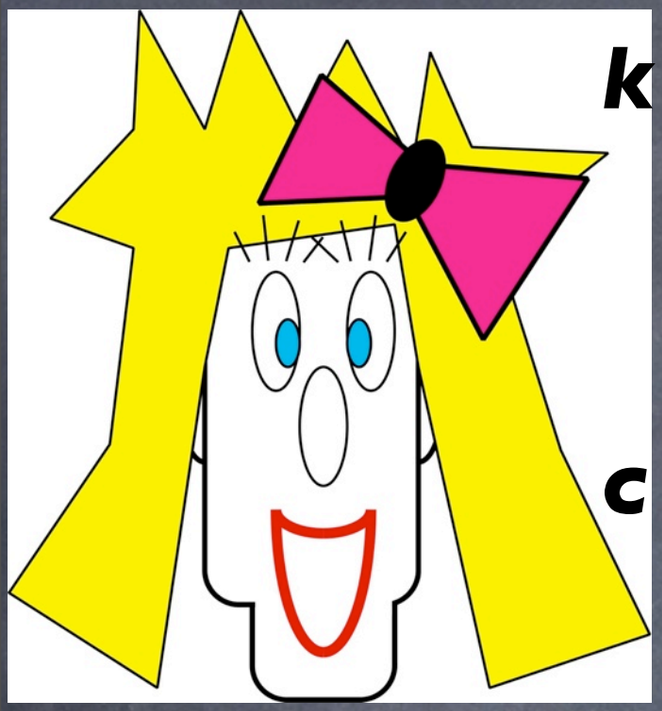


$k \leftarrow \mathrm{Gen}(1^n)$

$c \leftarrow \mathrm{Enc}_k(m)$

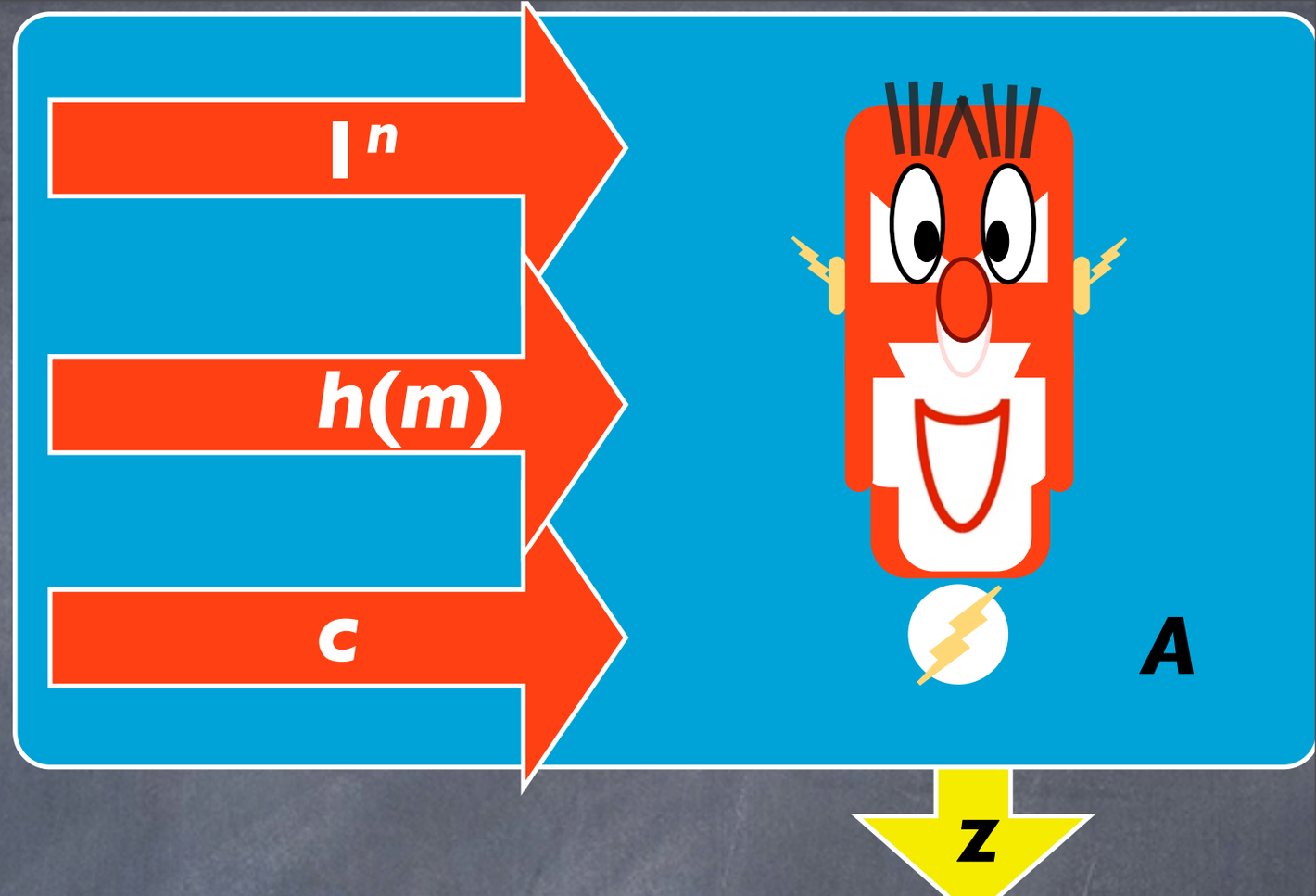


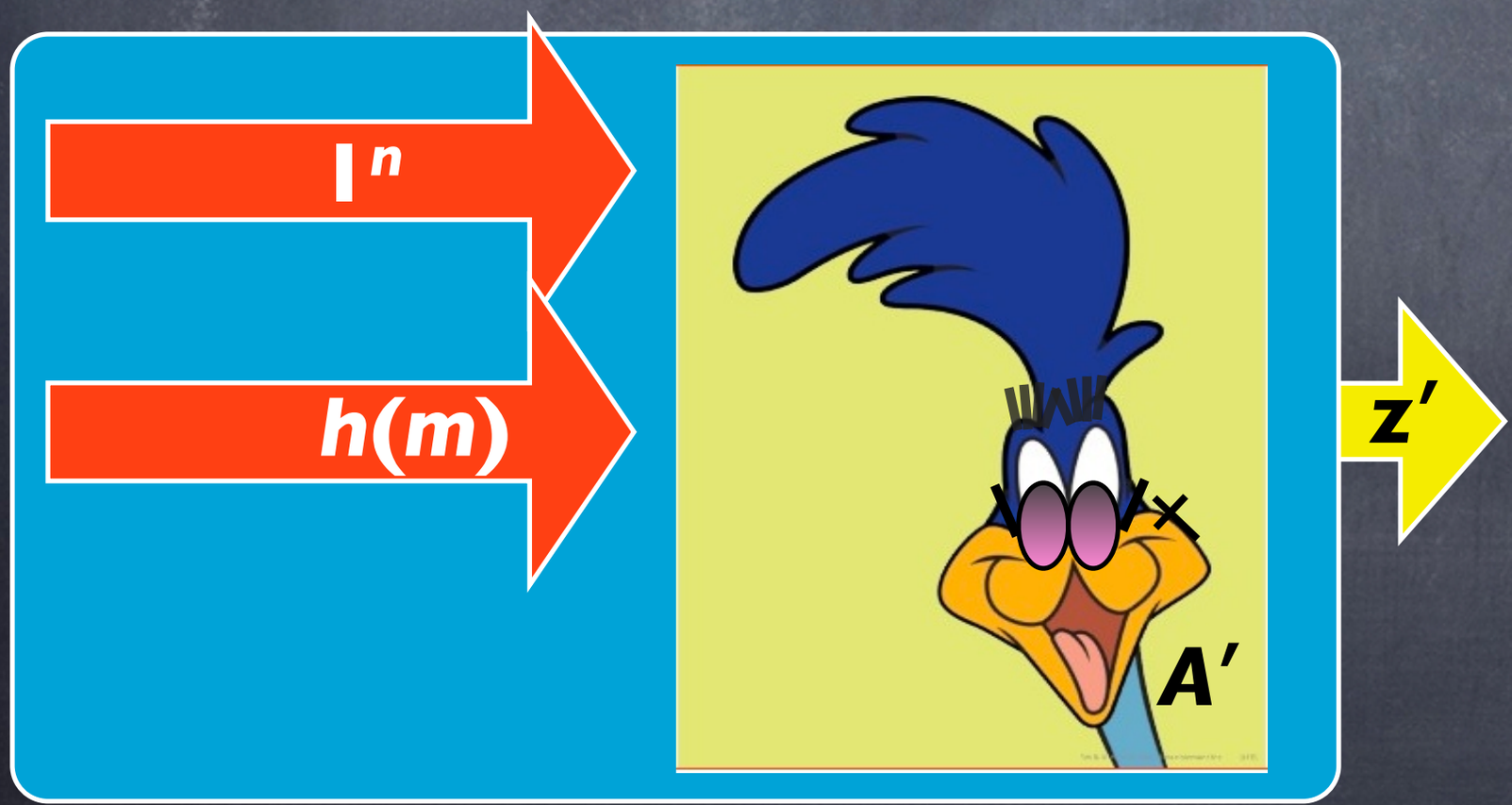


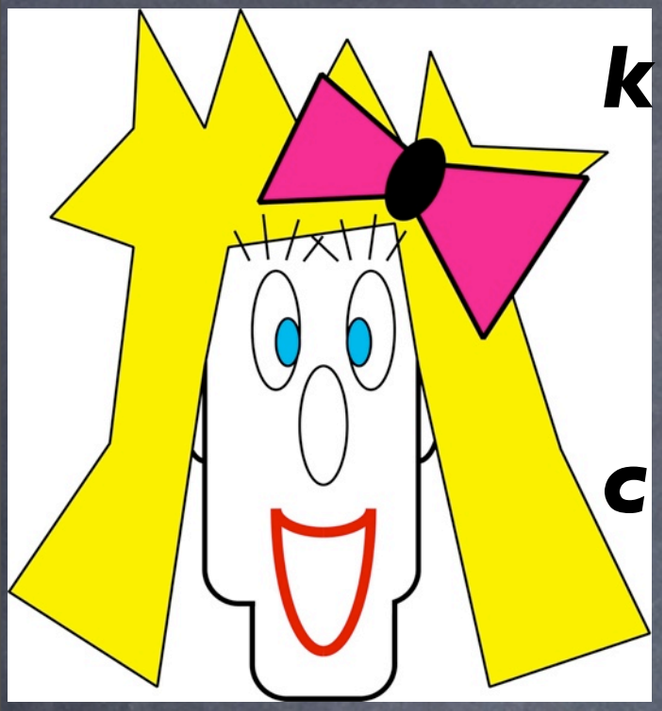


$k \leftarrow \mathrm{Gen}(1^n)$

$c \leftarrow \mathrm{Enc}_k(m)$

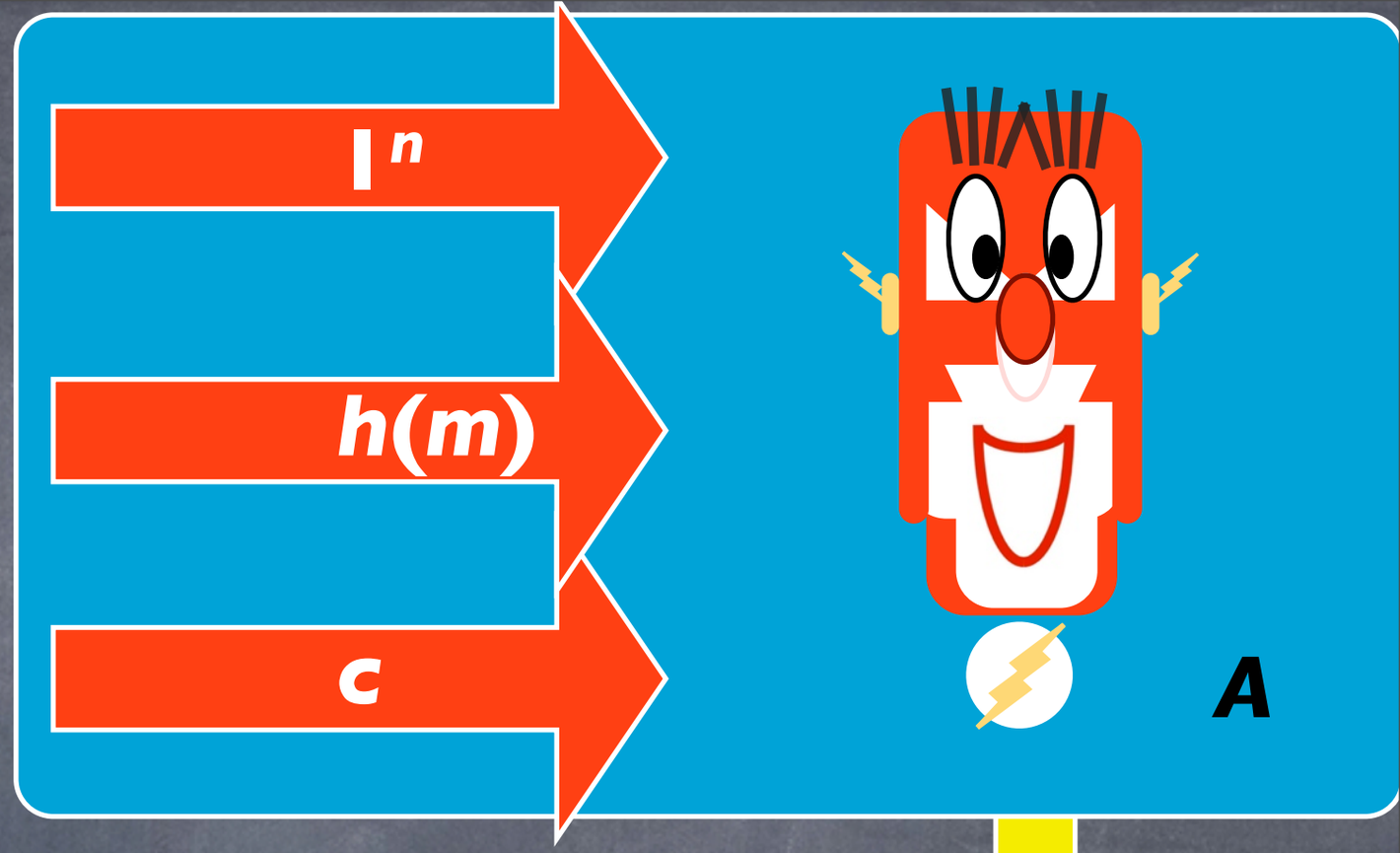




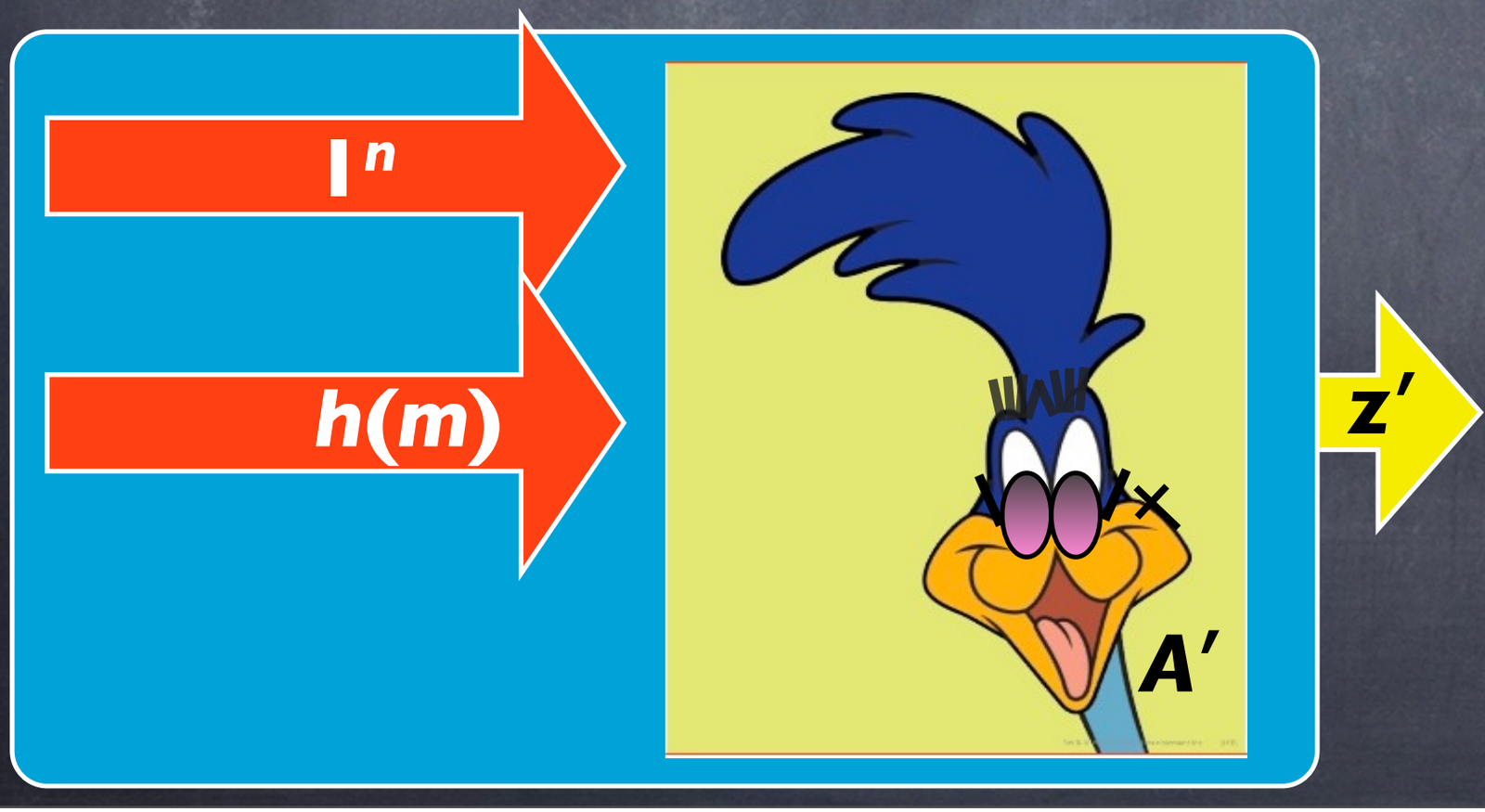


$k \leftarrow \mathrm{Gen}(1^n)$

$c \leftarrow \mathrm{Enc}_k(m)$



$$|\Pr[z = f(m)] - \Pr[z' = f(m)]| \leq \mathrm{negl}(n),$$



Semantic Security

THEOREM 3.13 A private-key encryption scheme has indistinguishable encryptions in the presence of an eavesdropper

if and only if

it is semantically secure in the presence of an eavesdropper.

Shafi Goldwasser

Silvio Micali

Post-Quantum Cryptography

- ~~Finite Fields~~ based cryptography
  - ~~Codes~~
  - ~~Multi-variate Polynomials~~
- Integers based cryptography
  - ~~Approximate Integer GCD~~
  - Lattices

Lattice based cryptography

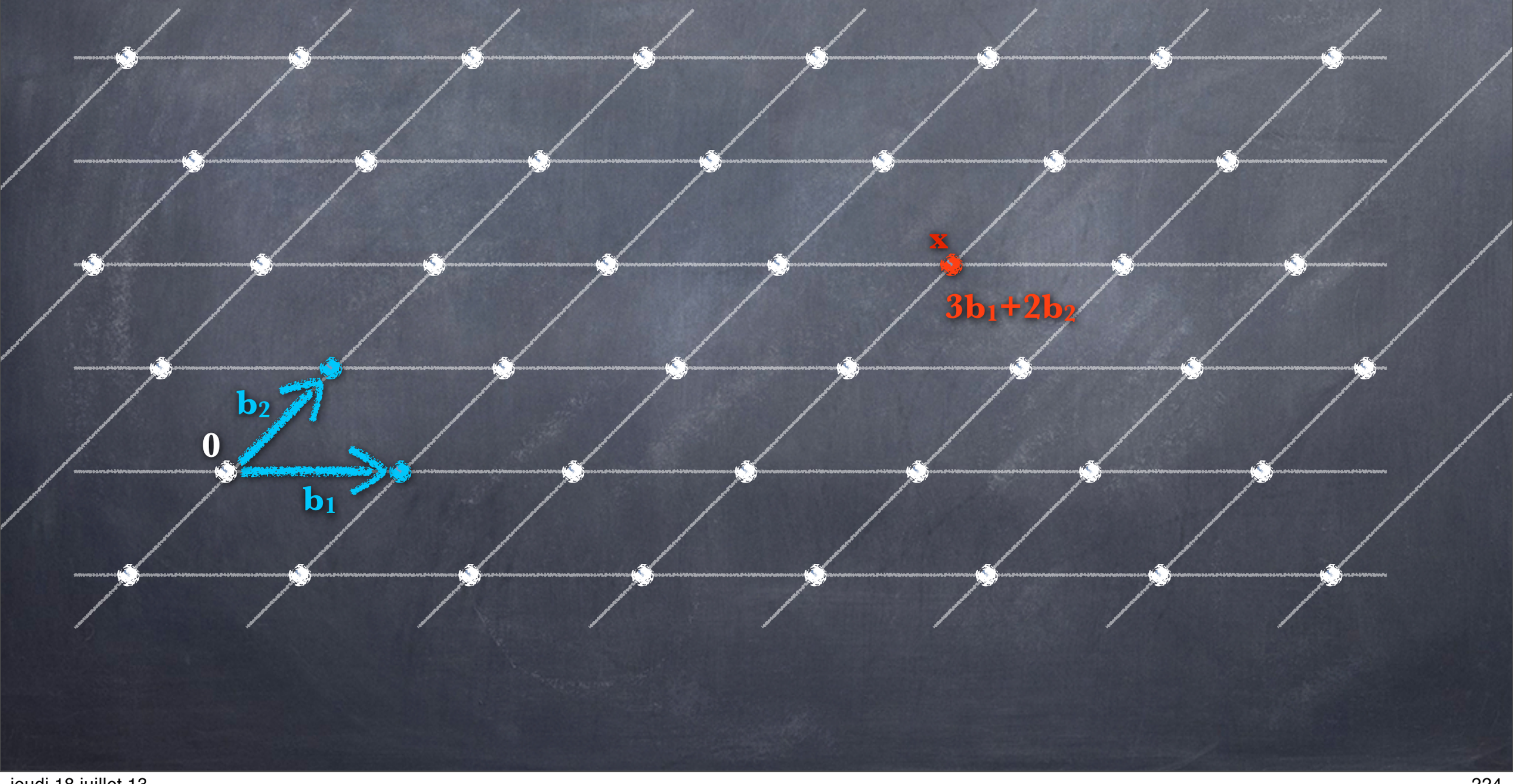
Lattices

- Given n-linearly independent vectors $b_1,\ldots,b_n \in \mathbb{R}^n$, the lattice they generate is the set of vectors

$$\mathcal{L}(b_1,\ldots,b_n) = \sum_{i=1}^{n} x_i b_i : x_i \in \mathbb{Z}.$$

- The vectors $b_1,\ldots,b_n$ are known as a basis of the lattice.

Lattices



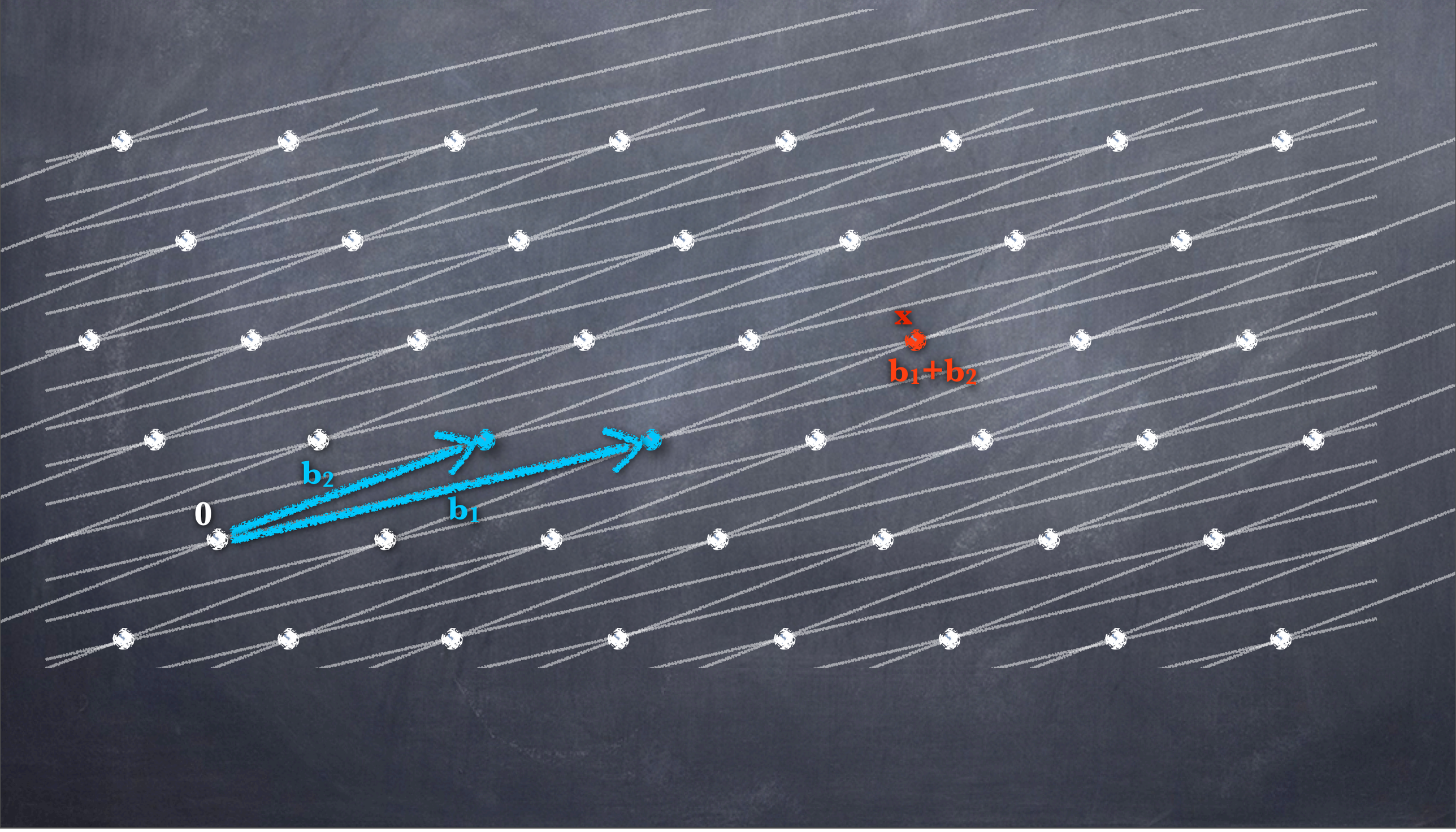
Integer Lattices

- Given n-linearly independent vectors $b_1,\ldots,b_n \in \mathbb{Z}^n$, the lattice they generate is the set of vectors

$$\mathcal{L}(b_1,\ldots,b_n) = \sum_{i=1}^{n} x_i b_i : x_i \in \mathbb{Z}.$$

- The vectors $b_1,\ldots,b_n$ are known as a basis of the lattice.

Lattices



Closest Vector Problem

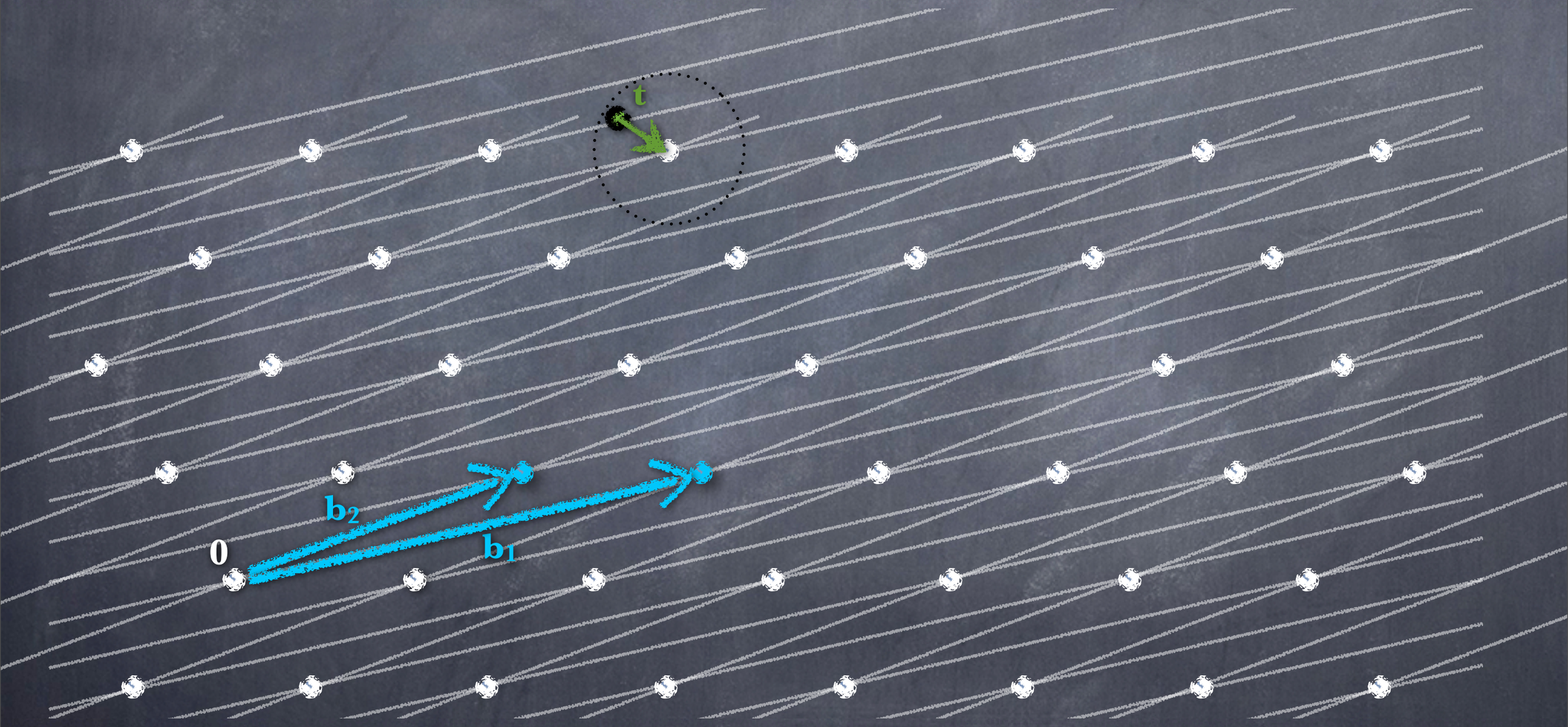
- Given a basis $b_1,\dots,b_n \in \mathbb{R}^n$, and a vector $t \in \mathbb{R}^n$ find the closest vector in the lattice $\mathcal{L}(b_1,\dots,b_n)$

$(x_1,\dots,x_n) \in \mathbb{Z}^n : d(t, \sum_{i=1}^{n} x_i b_i)$ is minimal.

- $d(u,v)$ is Euclidean distance

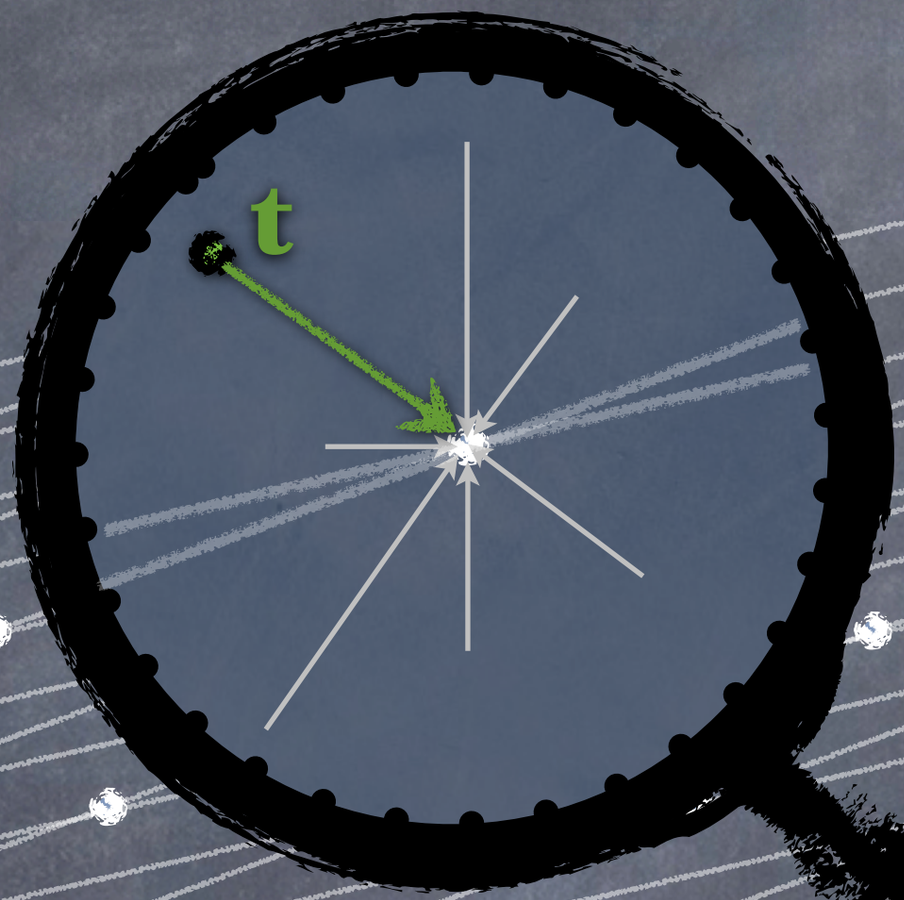
$$\sqrt{\sum_{i=1}^{n} (u_i - v_i)^2}$$

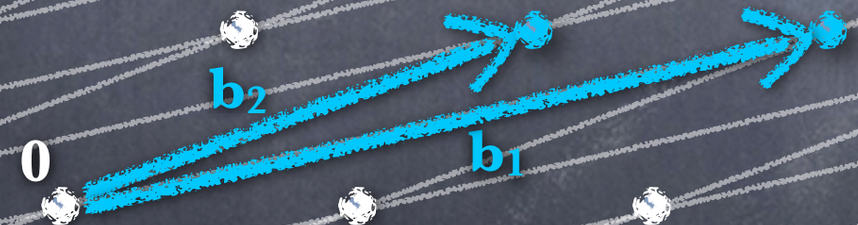
CVP



Analoguous to correcting errors in codes

CVP





Analoguous to correcting errors in codes

Shortest Vector Problem

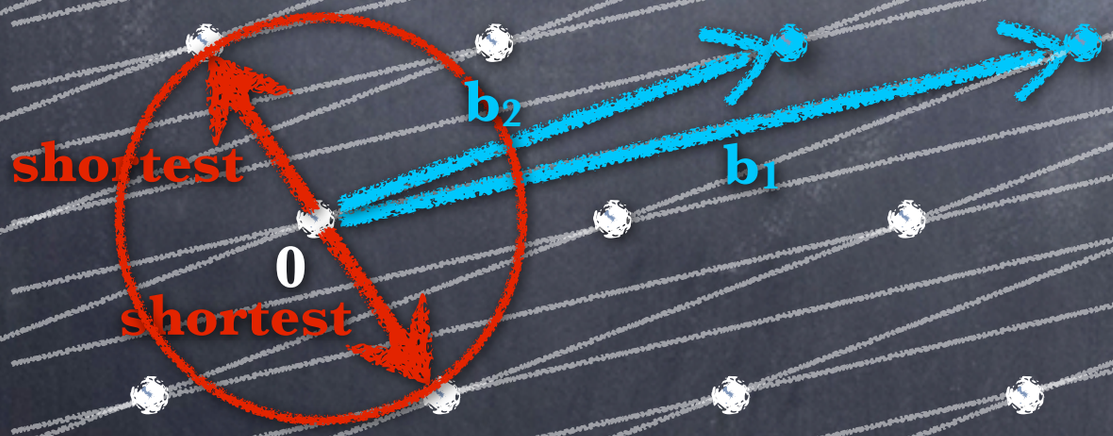
- Given a basis $b_1,\dots,b_n \in \mathbb{R}^n$ find the shortest vector in the lattice $\mathcal{L}(b_1,\dots,b_n)$

$(x_1,\dots,x_n) \in \mathbb{Z}^n \setminus 0 : d(0, \sum_{i=1}^{n} x_i b_i)$ is minimal.

- $d(u,v)$ is Euclidean distance

$$\sqrt{\sum_{i=1}^{n} (u_i - v_i)^2}$$

SVP



Analoguous to finding min distance in code

GGH

GGH

- The GGH cryptosystem, proposed by Goldreich, Goldwasser, and Halevi is essentially a lattice analogue of the McEliece/Niederreiter cryptosystem

GGH

- The GGH cryptosystem, proposed by Goldreich, Goldwasser, and Halevi is essentially a lattice analogue of the McEliece/Niederreiter cryptosystem

- The private key is a "good" lattice basis B.

GGH

- The GGH cryptosystem, proposed by Goldreich, Goldwasser, and Halevi is essentially a lattice analogue of the McEliece/Niederreiter cryptosystem

- The private key is a "good" lattice basis B.

- Typically, a good basis consists of short, almost orthogonal vectors.

GGH

- The GGH cryptosystem, proposed by Goldreich, Goldwasser, and Halevi is essentially a lattice analogue of the McEliece/Niederreiter cryptosystem

- The private key is a "good" lattice basis B.

- Typically, a good basis consists of short, almost orthogonal vectors.

- Algorithmically, good bases allow to efficiently solve certain instances of the closest vector problem in $\mathcal{L}(B)$, e.g., instances where the target is very close to the lattice.

GGH/HNF

GGH/HNF

- The public key H is a "bad" basis for the same lattice $\mathcal{L}(H) = \mathcal{L}(B)$.

GGH/HNF

- The public key H is a "bad" basis for the same lattice $\mathcal{L}(H) = \mathcal{L}(B)$.
- Micciancio proposed to use the Hermite Normal Form (HNF) of B. This normal form gives a lower triangular basis for $\mathcal{L}(B)$.

GGH/HNF

- The public key H is a "bad" basis for the same lattice $\mathcal{L}(H) = \mathcal{L}(B)$.
- Micciancio proposed to use the Hermite Normal Form (HNF) of B. This normal form gives a lower triangular basis for $\mathcal{L}(B)$.
- Notice that any attack on the HNF public key can be easily adapted to work with any other basis B′ of $\mathcal{L}(B)$ by first computing H from B′.

GGH/HNF

GGH/HNF

- The encryption process consists of adding a short noise vector r (somehow encoding the message to be encrypted) to a properly chosen lattice point v.

GGH/HNF

- The encryption process consists of adding a short noise vector r (somehow encoding the message to be encrypted) to a properly chosen lattice point v.
- It was proposed to select the vector v such that all the coordinates of (r + v) are reduced modulo the corresponding element along the diagonal of the HNF public basis H.

GGH/HNF

- The encryption process consists of adding a short noise vector r (somehow encoding the message to be encrypted) to a properly chosen lattice point v.
- It was proposed to select the vector v such that all the coordinates of (r + v) are reduced modulo the corresponding element along the diagonal of the HNF public basis H.
- The resulting vector is denoted r mod H, and it provably makes cryptanalysis hardest because r mod H can be efficiently computed from any vector of the form (r + v) with $v \in \mathcal{L}(B)$.

GGH/HNF

GGH/HNF

- The decryption problem corresponds to finding the lattice point v closest to the target ciphertext c = (r mod H) = v+r, and the error vector r = c−v.

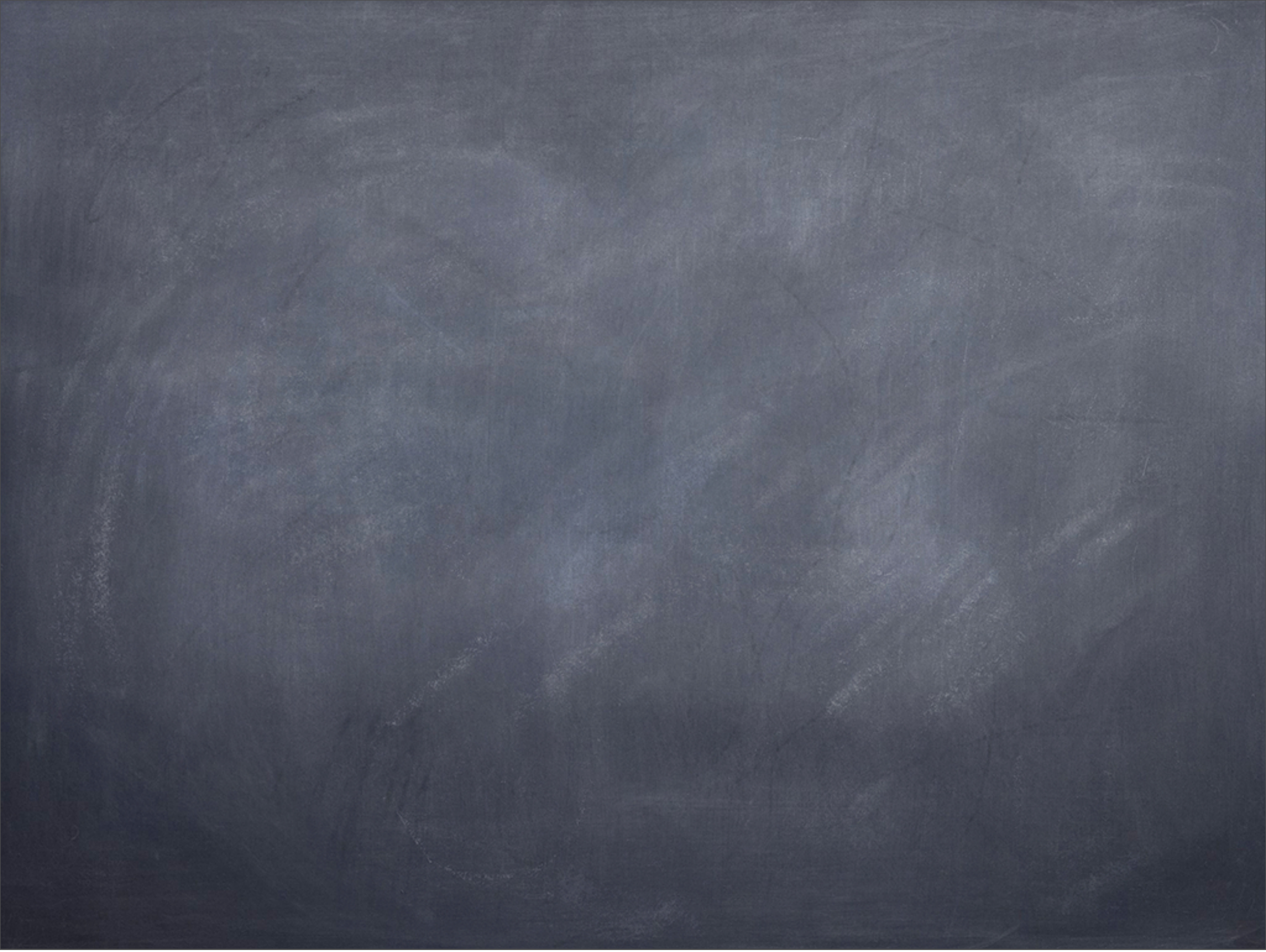
GGH/HNF

- The decryption problem corresponds to finding the lattice point v closest to the target ciphertext c = (r mod H) = v+r, and the error vector r = c−v.

- The correctness of the GGH/HNF cryptosystem rests on the fact that the error vector r is short enough so that the lattice point v can be recovered from the ciphertext v+r using the private basis B, e.g., by using Babai's rounding procedure, which gives

$$v = B\lfloor B^{-1}(v + r)\rceil$$

where $\lfloor x \rceil$ stands for the nearest integer to x



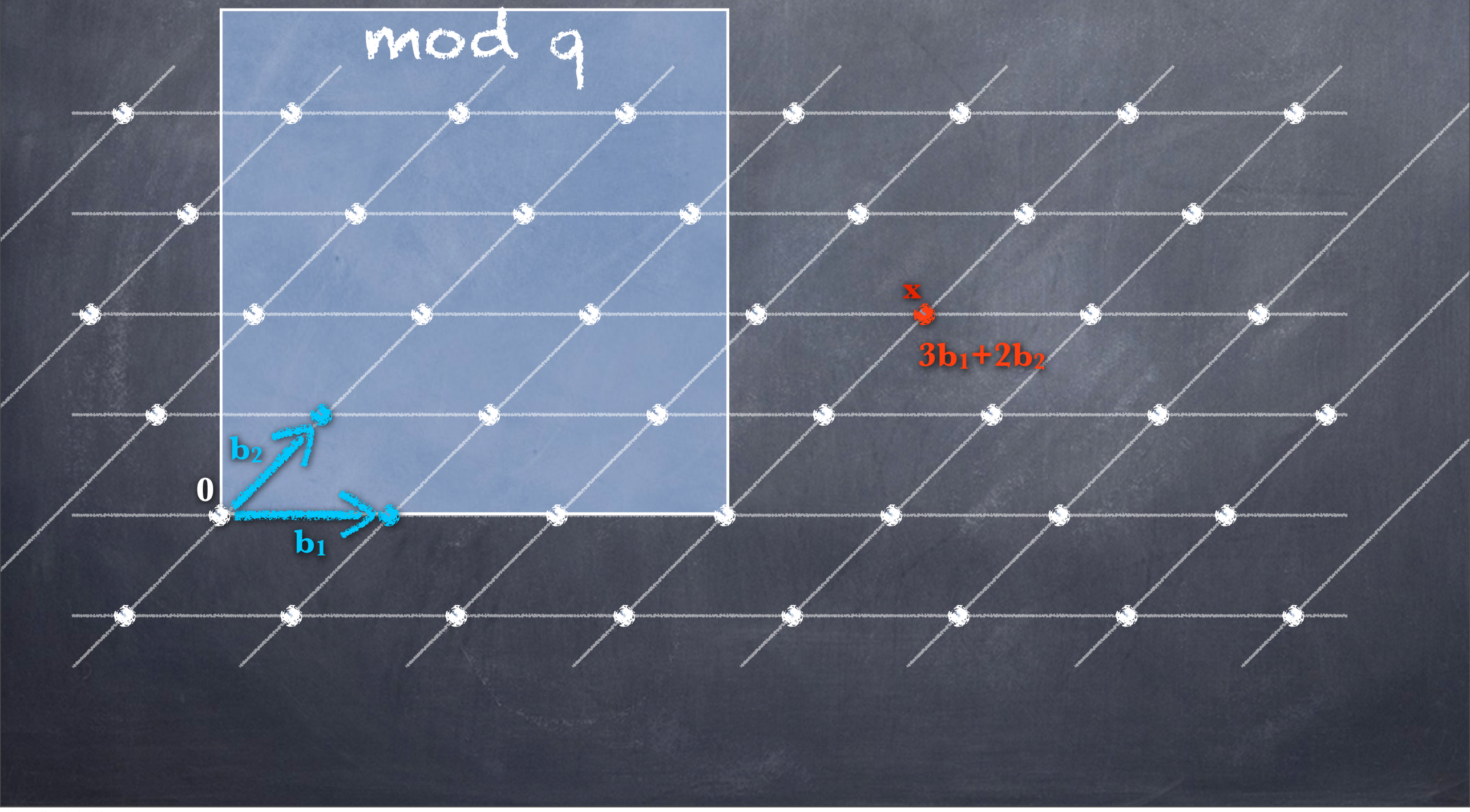
q-ary Lattices

- Given n-linearly independent vectors $b_1,\dots,b_n \in \mathbb{Z}^n$, the q-ary lattice they generate is the set of vectors

$$\mathcal{L}(b_1,\dots,b_n,q_1,\dots,q_n) = \sum_{i=1}^{n} x_i b_i \bmod q : x_i \in \mathbb{Z}$$

where each vector q_i is of the form $(0,\dots,0,q,0,\dots,0)$

q-ary Lattices



q-ary Lattices

q-ary Lattices

- Structure very similar to linear codes

q-ary Lattices

- Structure very similar to linear codes
- We define two types of q-ary lattices from a matrix $A \in \mathbb{Z}_q^{n \times m}$

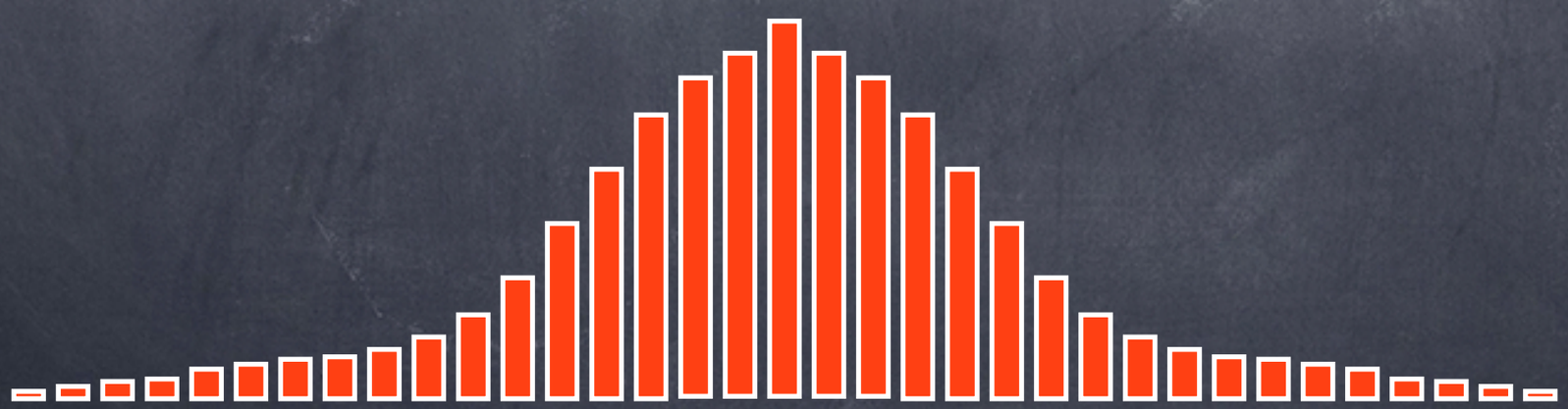
$$\Lambda_q(A) = \{y \in \mathbb{Z}_q^m : y = A^T s \bmod q,\ s \in \mathbb{Z}_q^n\}$$

$$\Lambda_q^\perp(A) = \{y \in \mathbb{Z}_q^m : Ay = 0 \bmod q\}$$

Learning With Errors

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

- LWE uses a discrete normal distribution $\overline{\Psi_\alpha}$ with mean 0 and standard deviation $q\alpha/\sqrt{2\pi}$ defined as $[\Psi_\alpha] \bmod q$

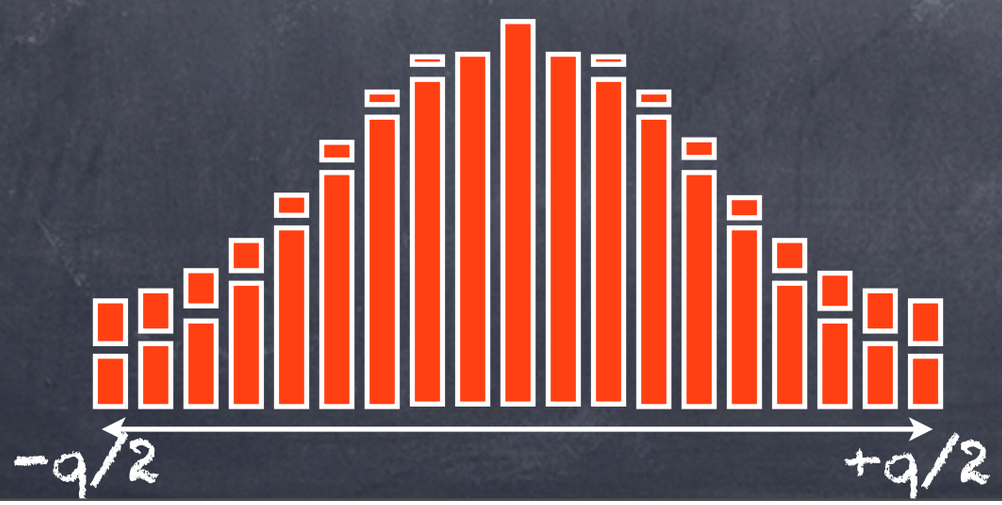


Learning With Errors

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

- LWE uses a discrete normal distribution $\overline{\Psi_\alpha}$ with mean 0 and standard deviation $q\alpha/\sqrt{2\pi}$ defined as

$$[\ \Psi_\alpha\ ] \bmod q$$



Learning With Errors

- A generalization of Learning Parity with Noise where q=2 and Bernouilli errors.

Learning With Errors

- A generalization of Learning Parity with Noise where q=2 and Bernouilli errors.
  - LWE is parametrized by n and q=poly(n)

Learning With Errors

- A generalization of Learning Parity with Noise where q=2 and Bernouilli errors.
  - LWE is parametrized by n and q=poly(n)
  - A: $\mathbb{Z}_q^{m \times n}$, a uniform public matrix

Learning With Errors

- A generalization of Learning Parity with Noise where q=2 and Bernouilli errors.
  - LWE is parametrized by n and q=poly(n)
  - A: $\mathbb{Z}_q^{m \times n}$, a uniform public matrix
  - S: $\mathbb{Z}_q^n$, a uniform secret (trapdoor) vector

Learning With Errors

- A generalization of Learning Parity with Noise where q=2 and Bernouilli errors.
  - LWE is parametrized by n and q=poly(n)
  - A: $\mathbb{Z}_q^{m \times n}$, a uniform public matrix
  - S: $\mathbb{Z}_q^n$, a uniform secret (trapdoor) vector
  - E: $\mathbb{Z}_q^m$, a secret vector where each entry has distribution $\overline{\Psi_\alpha}$ with α s.t. $\alpha q \approx \sqrt{n}$

    (reductions & there is an $\exp((\alpha q)^2)$-time attack)

Learning With Errors

- A generalization of Learning Parity with Noise where q=2 and Bernouilli errors.
  - LWE is parametrized by n and q=poly(n)
  - A: $\mathbb{Z}_q^{m \times n}$, a uniform public matrix
  - S: $\mathbb{Z}_q^n$, a uniform secret (trapdoor) vector
  - E: $\mathbb{Z}_q^m$, a secret vector where each entry has distribution $\overline{\Psi_\alpha}$ with α s.t. $\alpha q \approx \sqrt{n}$

    (reductions & there is an $\exp((\alpha q)^2)$-time attack)
  - (search-)LWE: Given A and P=AS+E find S.

Learning With Errors

Learning With Errors

- Decision-LWE is made of

Learning With Errors

- Decision-LWE is made of
- A: $\mathbb{Z}_q^{m \times n}$, a uniform public matrix

Learning With Errors

- Decision-LWE is made of
- A: $\mathbb{Z}_q^{m \times n}$, a uniform public matrix
- S: $\mathbb{Z}_q^n$, a uniform secret (trapdoor) vector

Learning With Errors

- Decision-LWE is made of
- A: $\mathbb{Z}_q^{m \times n}$, a uniform public matrix
- S: $\mathbb{Z}_q^n$, a uniform secret (trapdoor) vector
- E: $\mathbb{Z}_q^m$, a secret vector where each entry has distribution $\overline{\Psi_\alpha}$.

Learning With Errors

- Decision-LWE is made of
- A: $\mathbb{Z}_q^{m \times n}$, a uniform public matrix
- S: $\mathbb{Z}_q^n$, a uniform secret (trapdoor) vector
- E: $\mathbb{Z}_q^m$, a secret vector where each entry has distribution $\overline{\Psi_\alpha}$.
- Decision LWE : Given either A and P=AS+E or A,P for unfiorm P, identify which is the case.

Learning With Errors

- Decision-LWE is made of
- A: $\mathbb{Z}_q^{m \times n}$, a uniform public matrix
- S: $\mathbb{Z}_q^n$, a uniform secret (trapdoor) vector
- E: $\mathbb{Z}_q^m$, a secret vector where each entry has distribution $\overline{\Psi_\alpha}$.
- Decision LWE : Given either A and P=AS+E or A,P for unfiorm P, identify which is the case.
- Equivalent to the search problem.

LWE hardness

$$\begin{matrix}\text{GapSVP}\\ \text{SIVP}\end{matrix} \leq \text{search-LWE} \leq \text{decision-LWE} \leq \text{crypto}$$

LWE hardness

Quantum!!!

GapSVP / SIVP $\leq$ search-LWE $\leq$ decision-LWE $\leq$ crypto

LWE based cryptography

LWE based cryptography

- Private key: $S: \mathbb{Z}_q^n$, $E: \mathbb{Z}_q^m$ sampled using $\overline{\Psi_\alpha}$

LWE based cryptography

- Private key: S: $\mathbb{Z}_q^n$, E: $\mathbb{Z}_q^m$ sampled using $\overline{\Psi_\alpha}$
- Public Key: A: $\mathbb{Z}_q^{m \times n}$, P=AS+E

LWE based cryptography

- Private key: S: $\mathbb{Z}_q^n$, E: $\mathbb{Z}_q^m$ sampled using $\overline{\Psi_\alpha}$
- Public Key: A: $\mathbb{Z}_q^{m \times n}$, P=AS+E
- Input message: b: {0,1}

LWE based cryptography

- Private key: S: $\mathbb{Z}_q^n$, E: $\mathbb{Z}_q^m$ sampled using $\overline{\Psi_\alpha}$
- Public Key: A: $\mathbb{Z}_q^{m \times n}$, P=AS+E
- Input message: b: {0,1}
- $Enc_{AP}(v) := (A^T a, P^T a + bq/2)$ where a: $\{0,1\}^m$

LWE based cryptography

- Private key: S: $\mathbb{Z}_q^n$, E: $\mathbb{Z}_q^m$ sampled using $\overline{\Psi_\alpha}$
- Public Key: A: $\mathbb{Z}_q^{m \times n}$, P=AS+E
- Input message: b: {0,1}
- $Enc_{AP}(v) := (A^T a, P^T a + bq/2)$ where a: $\{0,1\}^m$
- $Dec_S(u,c) := 1$ (0) iff $c - S^T u$ is closer to q/2 (0)

$$\begin{aligned} c - S^T u &= P^T a + bq/2 - S^T A^T a \\ &= P^T a + bq/2 - P^T a + Ea \\ &= bq/2 + Ea \end{aligned}$$

LWE based cryptography

LWE based cryptography

- In the first part, one shows that distinguishing between public keys (A,P) as generated by the cryptosystem and pairs chosen uniformly at random from $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$ implies a solution to the LWE problem with parameters n,m,q, $\overline{\Psi_\alpha}$.

LWE based cryptography

- In the first part, one shows that distinguishing between public keys (A,P) as generated by the cryptosystem and pairs chosen uniformly at random from $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$ implies a solution to the LWE problem with parameters n,m,q, $\overline{\Psi_\alpha}$.

- The second part consists of showing that if one tries to encrypt with a public key (A,P) chosen at random, then with very high probability, the result carries essentially no statistical information about the encrypted message. (m > n log q)

LWE based cryptography

- In the first part, one shows that distinguishing between public keys (A,P) as generated by the cryptosystem and pairs chosen uniformly at random from $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$ implies a solution to the LWE problem with parameters n,m,q, $\overline{\Psi_\alpha}$.

- The second part consists of showing that if one tries to encrypt with a public key (A,P) chosen at random, then with very high probability, the result carries essentially no statistical information about the encrypted message. (m > n log q)

- Together, these two parts establish the security of the cryptosystem (under chosen plaintext attacks).

LWE-2 based cryptography

LWE-2 based cryptography

- Private key: S,E: $\mathbb{Z}_q^n$ both sampled using $\overline{\Psi_\alpha}$,

LWE-2 based cryptography

- Private key: S,E: $\mathbb{Z}_q^n$ both sampled using $\overline{\Psi_\alpha}$,
- Public Key: A: $\mathbb{Z}_q^{nxn}$, P=AS+E

LWE-2 based cryptography

- Private key: S,E: $\mathbb{Z}_q^n$ both sampled using $\overline{\Psi_\alpha}$,
- Public Key: A: $\mathbb{Z}_q^{nxn}$, P=AS+E
- Input message: b: {0,1}

LWE-2 based cryptography

- Private key: S,E: $\mathbb{Z}_q^n$ both sampled using $\overline{\Psi_\alpha}$,
- Public Key: A: $\mathbb{Z}_q^{n \times n}$, P=AS+E
- Input message: b: {0,1}
- $Enc_{AP}(v) := (A^T a + x, P^T a + bq/2 + E')$, a,x,E': $\mathbb{Z}_q^n$ using $\overline{\Psi_\alpha}$

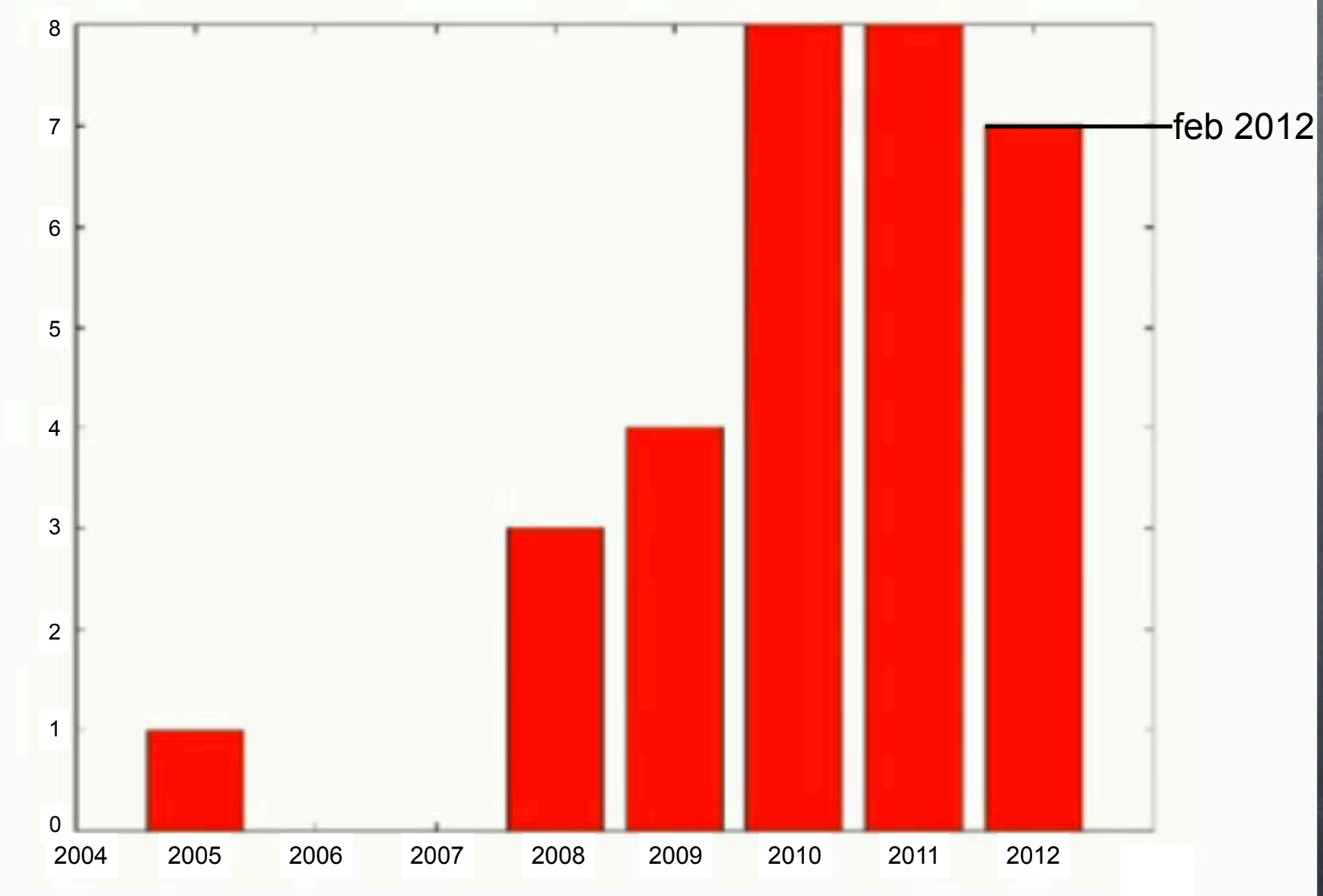
LWE-2 based cryptography

- Private key: S,E: $\mathbb{Z}_q^n$ both sampled using $\overline{\Psi_\alpha}$,
- Public Key: A: $\mathbb{Z}_q^{n \times n}$, P=AS+E
- Input message: b: {0,1}
- $Enc_{AP}(v) := (A^T a+x, P^T a+bq/2+E')$, a,x,E': $\mathbb{Z}_q^n$ using $\overline{\Psi_\alpha}$
- $Dec_S(u,c) := 1$ (0) iff $c-S^T u$ is closer to $q/2$ (0)

$$\begin{aligned} c-S^T u &= P^T a+bq/2+E'-S^T A^T a-S^T x \\ &= P^T a+bq/2+E'-P^T a+Ea-S^T x \\ &= bq/2+Ea+E'-S^T x \end{aligned}$$

LWE based cryptography

Crypto papers with "something new" regarding LWE:



©Peikert

Lattice based cryptography

§

Post-Quantum Cryptography

Prof. Claude Crépeau
McGill University

http://crypto.cs.mcgill.ca/~crepeau/WATERLOO