Cryptography

2017/18 Q2

Jaime Delgado *

DAC – UPC

* Part of the material comes from other sources.

Cryptography

- Private key (symmetric)
- Public key (asymmetric)
- Extended Euclidean algorithm ("magic box")
- Diffie-Hellman
- Encryption/Decryption algorithms for public key
  - RSA
  - ElGamal
- Digital signature
  - RSA

Security - Cryptography

- **Private key (symmetric)**

  Sender: $c = E_k(m)$; Recipient: $m = D_k(c)$

  E: Encryption algorithm; D: Decryption algorithm;

  k: Key; c: cyphered text; m: clear text (message).

  – Historical: Transposition, Substitution, …

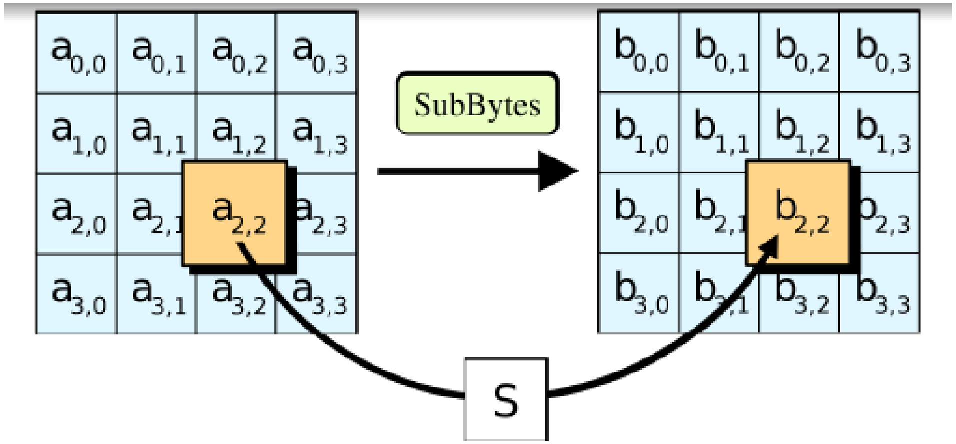
  – Block cipher

    - Principles:
      Confusion (*key* → *cipher text* independence),
      Diffusion (*plain text* → *cipher text* independence)

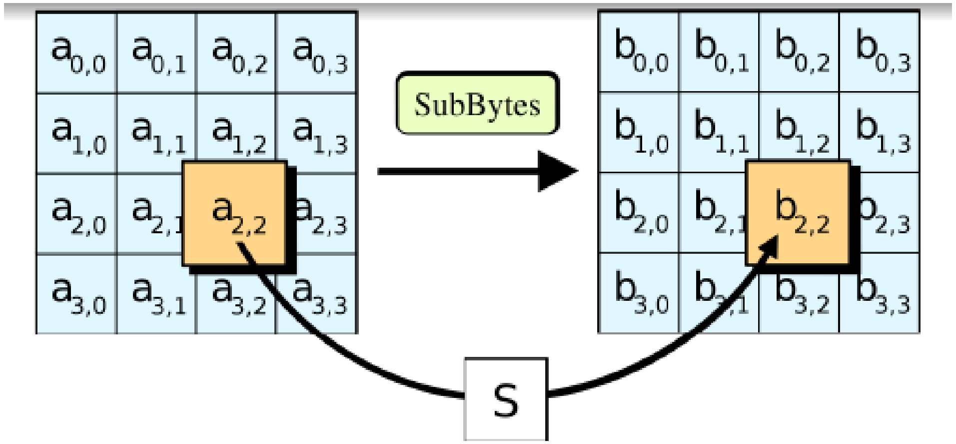
    - DES (Data Encryption Standard), 1976
    - AES (Advanced Encryption Standard), 2001

  (permutations)

AES permutations

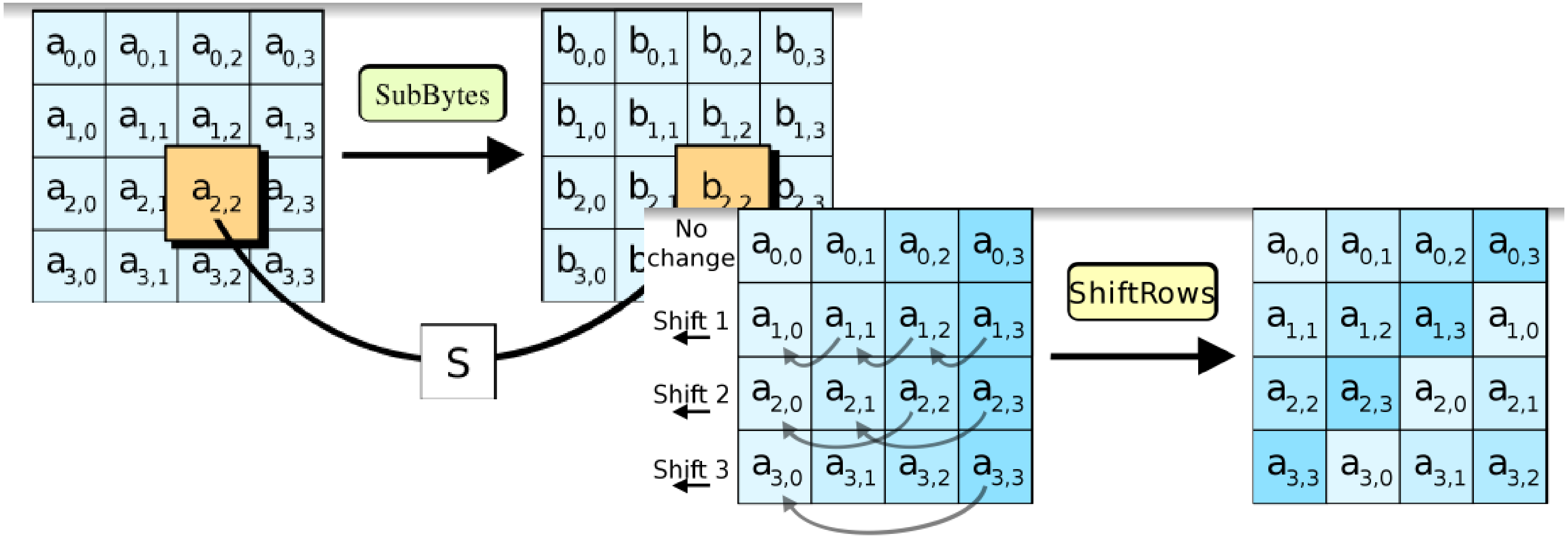


AES permutations

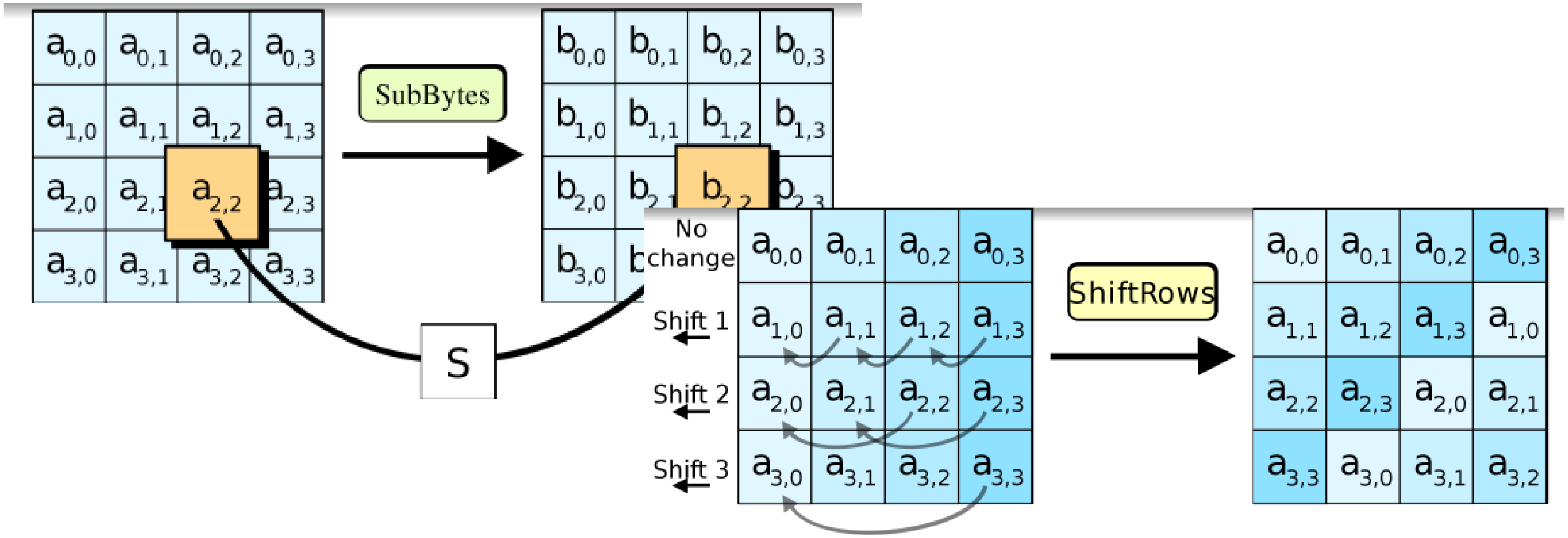


SubBytes: a non-linear substitution step where each byte is replaced with another according to a lookup table

AES permutations

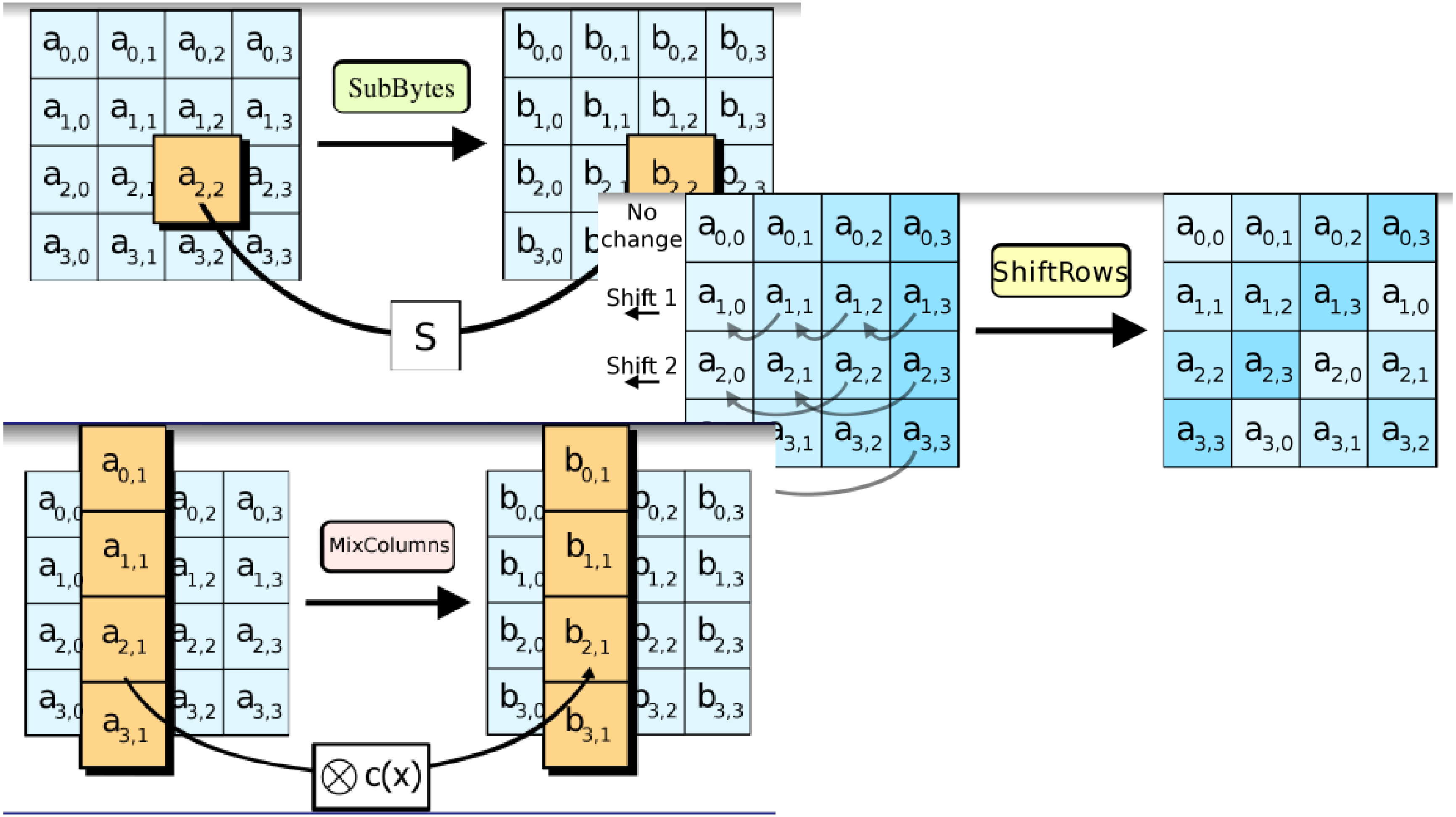


AES permutations

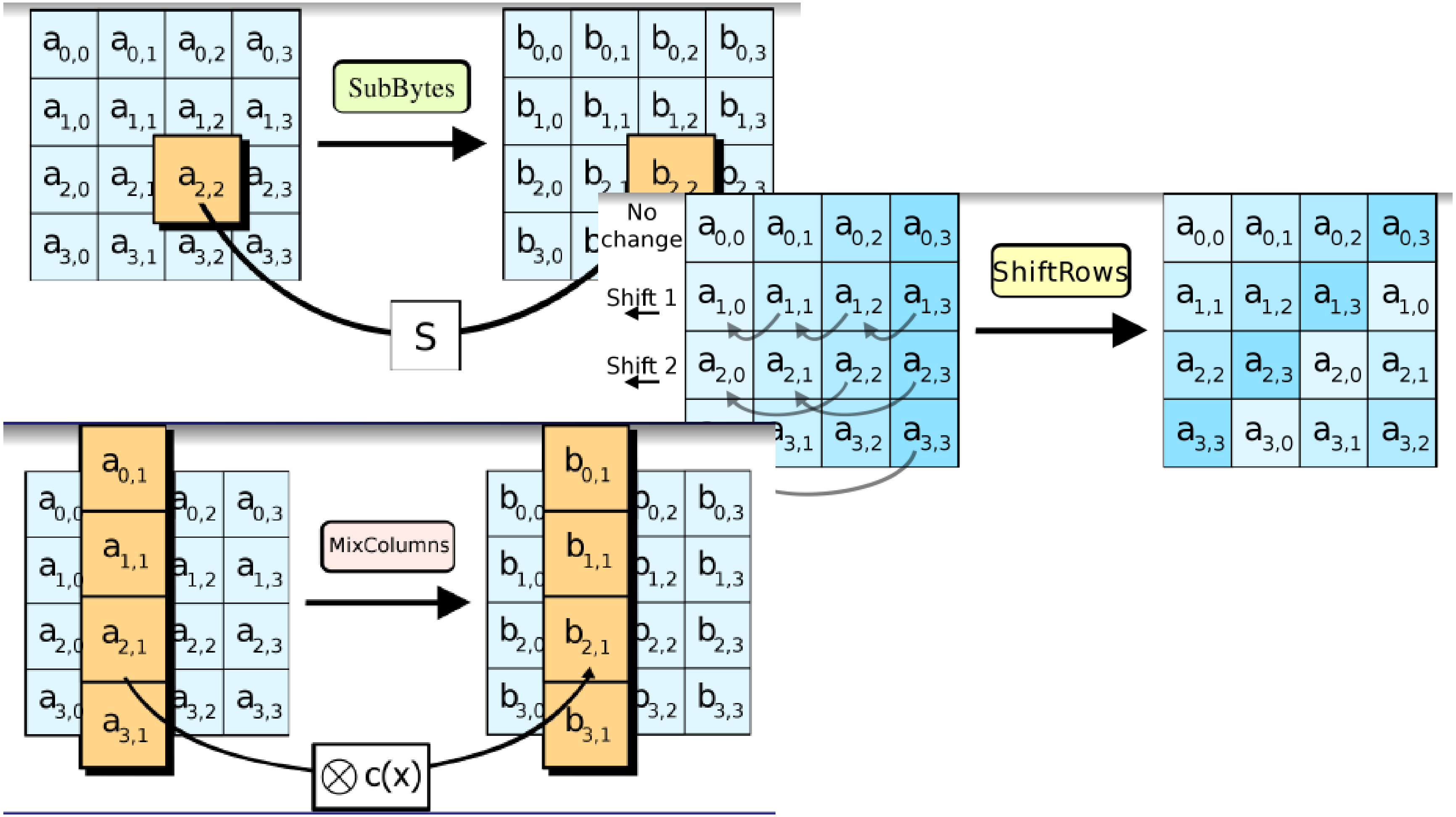


ShiftRows: a transposition step where each row of the state is shifted cyclically a certain number of steps

AES permutations

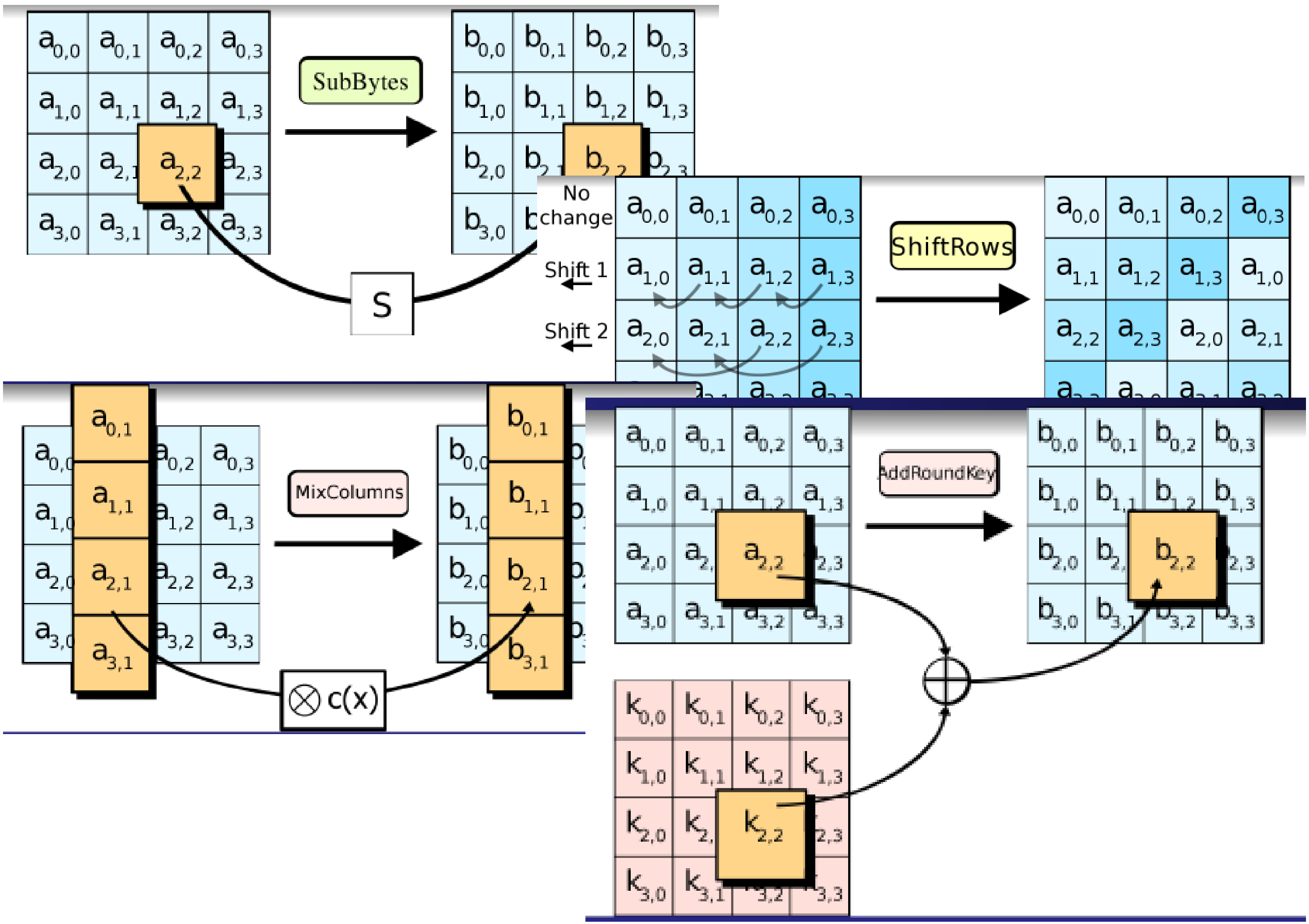


AES permutations

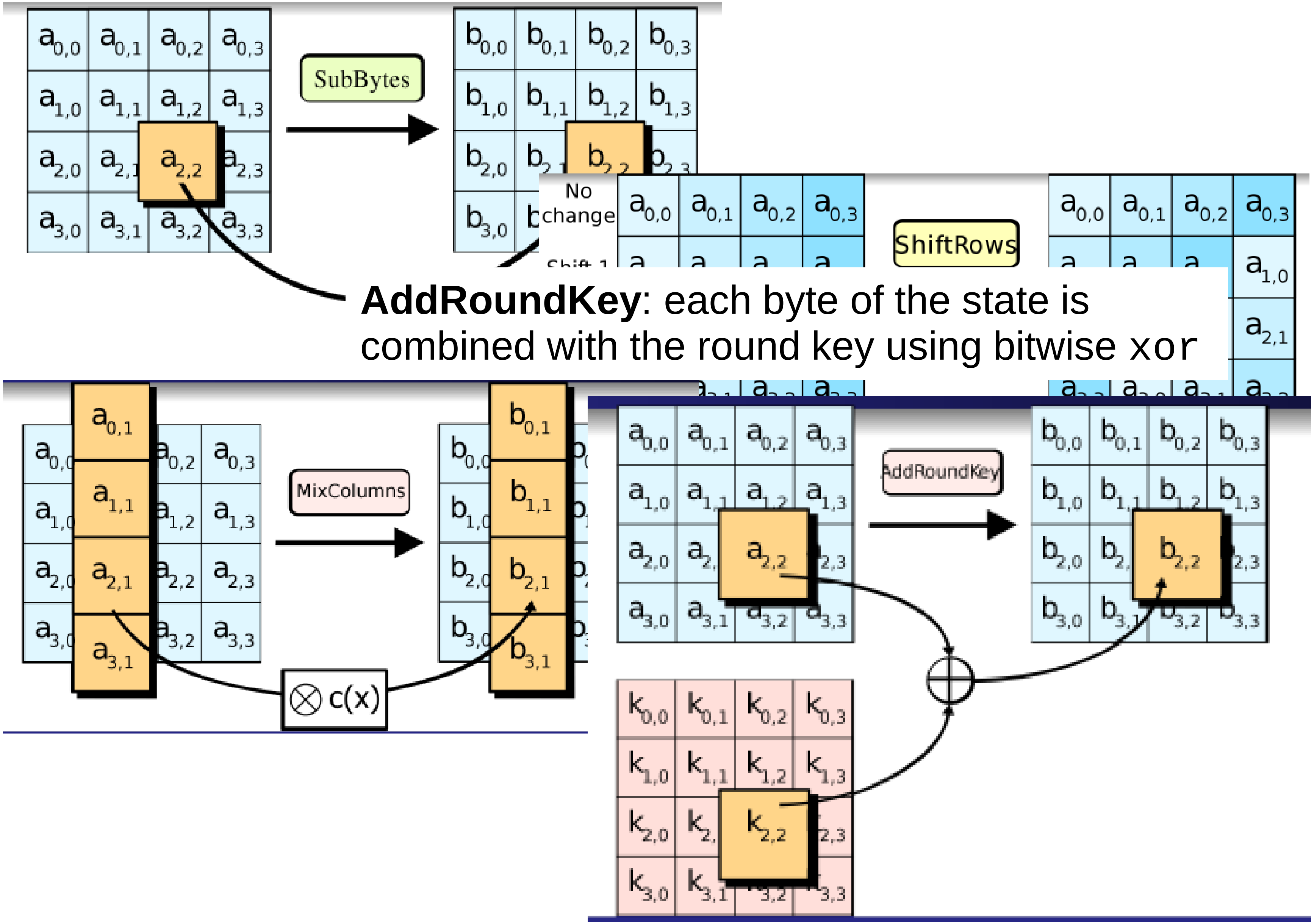


MixColumns: a mixing operation which operates on the columns of the state, combining the four bytes in each column

AES permutations



AES permutations



AddRoundKey: each byte of the state is combined with the round key using bitwise xor

Cryptography

- Private key (symmetric)
- **Public key (asymmetric)**
  - Secret + Public parts of the key (Ks, Kp)
  - No need for keys distribution
  - *Encryption*

    Sender: $c = E_{Kp}(m)$; Recipient: $m = D_{Ks}(c)$

    (Ks,Kp) from the Recipient.
  - *Signature*

    Sender: $s = E_{Ks}(m)$; Recipient: $m = D_{Kp}(s)$

    *"Sender" is the **Signer**. (Ks,Kp) from the Signer.*

Cryptography

- Private key (symmetric)
- Public key (asymmetric)
- **Extended Euclidean algorithm (“magic box”)**
- Diffie-Hellman
- Encryption/Decryption algorithms for public key
  - RSA
  - ElGamal
- Digital signature
  - RSA

Extended Euclidean algorithm

- The extended Euclidean algorithm computes the *greatest common divisor (gcd)* of two integers *a* and *n*
- It is particularly useful when *a* and *n* are *coprime*, (i.e. *gcd (a,n) = 1*), since its output is the modular multiplicative inverse of *a* mod *n*

'The Magic Box' method

a	b	d	k
1	0	n	
0	1	a	

$d_i = d_{i-2} \bmod d_{i-1}$

k_i is the quotient of d_{i-1}/d_i

$a_i = a_{i-2} - k_{i-1} \cdot a_{i-1}$

$b_i = b_{i-2} - k_{i-1} \cdot b_{i-1}$

The procedure finishes when d_i =1 (if a,n are coprime)

Extended Euclidean algorithm

- The extended Euclidean algorithm computes the *greatest common divisor (gcd)* of two integers a and n
- It is particularly useful when a and n are *coprime*, (i.e. $gcd\ (a,n) = 1$), since its output is the modular multiplicative inverse of a mod n

'The Magic Box' method

a	b	d	k
1	0	n	
0	1	a	

Starting point

$$d_i = d_{i-2} \bmod d_{i-1}$$

k_i is the quotient of d_{i-1}/d_i

$$a_i = a_{i-2} - k_{i-1} \cdot a_{i-1}$$

$$b_i = b_{i-2} - k_{i-1} \cdot b_{i-1}$$

The procedure finishes when $d_i = 1$ (if a,n are coprime)

“Magic Box” example

Compute the GCD of 120 and 23

a	b	d	k
1	0	120	
0	1	23	5
1	-5	5	4
-4	21	3	1
5	-26	2	1
-9	47	1	2

$d_i = d_{i-2} \bmod d_{i-1}$

k_i is the quotient of d_{i-1}/d_i

$a_i = a_{i-2} - k_{i-1} \cdot a_{i-1}$

$b_i = b_{i-2} - k_{i-1} \cdot b_{i-1}$

$d_3 = 120 \bmod 23 = 5$ k_2 is the quotient of $120/23$

$a_3 = 1 - 5 \cdot 0 = 1$

$b_3 = 0 - 5 \cdot 1 = -5$

- 47 is the **modular multiplicative inverse** of 23 mod 120.
- Bézout identity in the example: 1 = 23*47 + 120*(-9)

Cryptography

- Private key (symmetric)
- Public key (asymmetric)
- Extended Euclidean algorithm (“magic box”)
- **Diffie-Hellman**
- Encryption/Decryption algorithms for public key
  - RSA
  - ElGamal
- Digital signature
  - RSA

Diffie-Hellman

- Modular arithmetic algorithm with several uses:
  - Part of asymmetric key mechanisms.
  - **Private key generation** in symmetric key mechanisms.

- A: takes random element $a \in G$; computes $\alpha^a \in G$;
  B: id. with $b \in G$; $\alpha^b \in G$

  (A: Sender; B: Recipient; G and α known by both;
  G: multiplicative finite group with generator $\alpha \in G$).
- A & B interchange α^a and α^b
- A computes $(\alpha^b)^a$; B computes $(\alpha^a)^b$.
- $(\alpha^b)^a = (\alpha^a)^b$ is the private key !!! Only A & B know.

Diffie-Hellman example

Example

1. A and B choose publicly $G = \mathbb{Z}_{53}^*$ and the generator $\alpha = 2$
2. A chooses $a = 29$, computes $\alpha^a = 2^{29} \bmod 53 = 45$ and sends 45 to B
3. B chooses $b = 19$, computes $\alpha^b = 2^{19} \bmod 53 = 12$ and sends 12 to A
4. A receives 12 and computes $12^{29} \bmod 53 = 21$
5. B receives 45 and computes $45^{19} \bmod 53 = 21$

The **private key** is 21

Exponentiation by squaring

Modular arithmetic allows us to compute exponentiations without managing very big numbers!

Exponentiation by squaring (a,z,n) $x = a^z \bmod n$

begin
- $x = 1$;
- $z^1 =$ binary representation of z;
- `// starting by the most significant bit`
- **foreach** *bit* $z_i^1 \in z^1$ **do**
  - $x = x^2 \bmod n$;
  - `// multiply` x `by` a `if` z_1 `is equal to one`
  - **if** $z_i^1 == 1$ **then**
    - $x = x \cdot a \bmod n$
- **return** x

Exponentiation by squaring - Example

Example

Compute $5^{27} \bmod 217$

27 is 11011 in binary

$5^{27} \bmod 217 \Rightarrow 1 \rightarrow S \rightarrow 1 \rightarrow M \rightarrow 5 \rightarrow S \rightarrow 25 \rightarrow M \rightarrow 125 \rightarrow S \rightarrow 15625 \equiv 1 \rightarrow S \rightarrow 1 \rightarrow M \rightarrow 5 \rightarrow S \rightarrow 25 \rightarrow M \rightarrow 125$

S: squaring
M: multiply

Cryptography

- Private key (symmetric)
- Public key (asymmetric)
- Extended Euclidean algorithm ("magic box")
- Diffie-Hellman
- **Encryption/Decryption algorithms for public key**
  - RSA
  - ElGamal
- Digital signature
  - RSA

RSA

KEYS GENERATION

- Choose 2 distinct **very big** prime numbers p and q.
- Compute $n=p*q$; n defines the multiplicative group $\mathbb{Z}n$
- Compute $\Phi(n)=(p-1)*(q-1)$
- Choose an integer e such that $1<e<\Phi(n)$, and $GCD(e,\Phi(n))=1$ (i.e., e and $\Phi(n)$ are coprime)
- Determine $d=e^{-1} mod \Phi(n)$ using Modular multiplicative inverse (extended Euclidean algorithm)
- The **public key** is **(n,e)**
- The **secret key** is **d** or **(n,d)**

  p, ***q*** *and* ***Φ(n)*** *are also secret*

RSA

ENCRYPTION

- B has keys:

  PUBLIC (n,e); SECRET (d)

- A wants to sends `m` to B:

$$c = m^e \bmod n$$

DECRYPTION

- B receives c and computes:

$$m = c^d \bmod n$$

RSA

EXAMPLE

Keys generation:

- *p=61, q=53*; n=61*53=3223 $n=p*q$
- Φ(3233)=60*52=3120 $\Phi(n)=(p-1)*(q-1)$
- e=17 (1<17<3120)
  e coprime to 3120 (i.e. *e* not a divisor of 3120)
- d = 17^{-1} mod 3120 = 2753 $d=e^{-1} \bmod \Phi(n)$
  ext. Eucl. alg.

Public key: (n=3233, e=17)

Secret key: (n=3233, **d=2753)**

RSA

EXAMPLE

Keys generation (d calculation):

- $d = 17^{-1} \bmod 3120 = 2753$

$d=e^{-1} \bmod \Phi(n)$

- Extended Euclidean algorithm:

b	d	k
0	**3120**	-
1	**17**	183
-183	9	1
184	8	1
-367	1	

$bi=b_{i-2}-(k_{i-1}*b_{i-1})$

3120-367=**2753**

RSA

EXAMPLE

Encryption / Decryption:

- *m=65*;

$$c = m^e \bmod n$$

- $c = 65^{17} \bmod 3233 = 2790$

$$m = c^d \bmod n$$

- $m = 2790^{2753} \bmod 3233 = 65$

ElGamal

KEYS GENERATION

- Choose a cyclic multiplicative finite group G and one element $\alpha \in G$.
- Users choose a random number a → Secret key *Ks*.
- Also compute $\alpha^a \in G$ → Public key *Kp*.

(Although α, G and α^a (i.e. *Kp*) are publicly known,
a (i.e. *Ks*) is not known)

ElGamal

ENCRYPTION

- A has keys:

  PUBLIC ($Kp=\alpha^a$); SECRET ($Ks=a$)

- B wants to send $m \in G$ to A
- B choses a random number v and computes $\alpha^v \in G$
- B computes

  $c = m * (Kp)^v \in G$; i.e.:

  $c = m * (\alpha^a)^v \bmod G$

- B sends to A:

  (α^v, c)

ElGamal

ENCRYPTION

- A has keys:

  PUBLIC ($Kp=\alpha^a$); SECRET ($Ks=a$)

- B wants to send $m \in G$ to A
- B choses a random number v and computes $\boxed{\alpha^v} \in G$
- B computes

  $c = m * (Kp)^v \in G$; i.e.:

  $\boxed{c} = m * (\alpha^a)^v \bmod G$

- B sends to A:

  $\boxed{(\alpha^v, c)}$

ElGamal

DECRYPTION

- A receives (α^v, c)
- A computes

$$(\alpha^v)^{Ks} \in G\text{; i.e:}$$
$$(\alpha^v)^a \in G = (\alpha^{va}) \bmod G$$

- A computes

$$\mathbf{m = c * (\alpha^{va})^{-1} \in G}\text{; i.e.:}$$
$$\mathbf{m = c * (\alpha^{va})^{-1} \bmod G}$$

$\boxed{\mathbf{Ks = a}}$

$(\alpha^{va})^{-1}$ is the modular multiplicative inverse of $(\alpha^{va}) \bmod G$

ElGamal

EXAMPLE

Keys generation:

- *G=13*; *α=2*; ***a=9***
- Kp = α^a = 2^9 mod 13 = 5 $\boxed{\alpha^a \in G}$

***Secret key: Ks*=9**

Public key: Kp=5

ElGamal

EXAMPLE

Encryption:

- *m=11*; *v=**10***; (*G=13*; *α=2*); (Ks=9, Kp=**5**)

- $\alpha^v = 2^{10} \bmod 13 = 10$

$\alpha^v \in G$

$c = m*(Kp)^v \bmod G$

- $c = 11 * \mathbf{5}^{10} \bmod 13 = 11 * 12 \bmod 13 = 2$

$Kp=\alpha^a$

- Sends **(10, 2)**

ElGamal

EXAMPLE

Encryption:

- $m=11$; $v=\mathbf{10}$; ($G=13$; $\alpha=2$); (Ks=9, Kp=**5**)

- $\alpha^v = 2^{10} \bmod 13 = 10$

$\alpha^v \in G$

$c = m*(Kp)^v \bmod G$

- $c = 11 * \mathbf{5}^{10} \bmod 13 = 11 * 12 \bmod 13 = 2$

$Kp=\alpha^a$

- Sends **(10, 2)**

ElGamal

EXAMPLE

Decryption:

- Receives $(\alpha^v, c) = (10,2)$. Ks=9; ($G=13$; $\alpha=2$)

- m = 2 * $(10^9)^{-1}$ mod 13

$m = c * (\alpha^{va})^{-1} \bmod G$

Ks=a

- m = (2 * ((10^9 mod 13 $)^{-1}$ mod 13)) mod 13

$(\alpha^v)^a = 10^9 \bmod 13 = 12$

- m = (2 * (12^{-1} mod 13)) mod 13

“magic box”

ElGamal

EXAMPLE

Decryption **("magic box" calculation):**

- Ks=9; (*G=13*; α=2). Receives $(\alpha^v, c) = (10, 2)$

- $(\alpha^{va})^{-1}$ mod G = 12^{-1} mod 13 = 12 ("magic box")

```
  b  |   d  |  k
----------|-----------|----------
 0  | 13  |  -
 1  | 12  | 1
-1  |  1  |
```

$b_i = b_{i-2} - (k_{i-1} * b_{i-1})$

13-1=**12**

ElGamal

EXAMPLE

Decryption:

- Receives $(\alpha^v, c) = (10,2)$. Ks=9; ($G=13$; $\alpha=2$)
- $m = 2 * (10^9)^{-1} \bmod 13$

$m = c * (\alpha^{va})^{-1} \bmod G$

Ks=a

- $m = (2 * ((10^9 \bmod 13)^{-1} \bmod 13)) \bmod 13$

$10^9 \bmod 13 = 12$

- $m = (2 * (12^{-1} \bmod 13)) \bmod 13$

12 ("magic box")

$m = (2 * 12) \bmod 13 = 11$

Cryptography

- Private key (symmetric)
- Public key (asymmetric)
- Extended Euclidean algorithm (“magic box”)
- Diffie-Hellman
- Encryption/Decryption algorithms for public key
  - RSA
  - ElGamal
- **Digital signature**
  - RSA

(Public key) Digital signature

– *Encryption*

Sender: $\mathtt{c = E_{Kp}(m)}$; Recipient: $\mathtt{m = D_{Ks}(c)}$

(Ks,Kp) from the Recipient.

– *Signature*

Sender: $\mathtt{s = E_{Ks}(m)}$; Recipient: $\mathtt{m = D_{Kp}(s)}$

*"Sender" is the **Signer**. (Ks,Kp) from the Signer.*

- To reduce computational cost,
  sign `Hash(m)` instead of `m`
  (Hash: unidirectional; large variable-size → small fixed-size)
- Signature is distributed with message (encrypted or not).
- *RSA algorithm. There are many.*

RSA signature

SIGNATURE

- A has keys:

  PUBLIC (n,e); SECRET (d)

- A signs a message (or its Hash) `m`:

$$s = m^d \bmod n$$

VERIFICATION

- B receives `s` and `m` (encrypted or not):

$$m = s^e \bmod n$$

`m` should be equal to