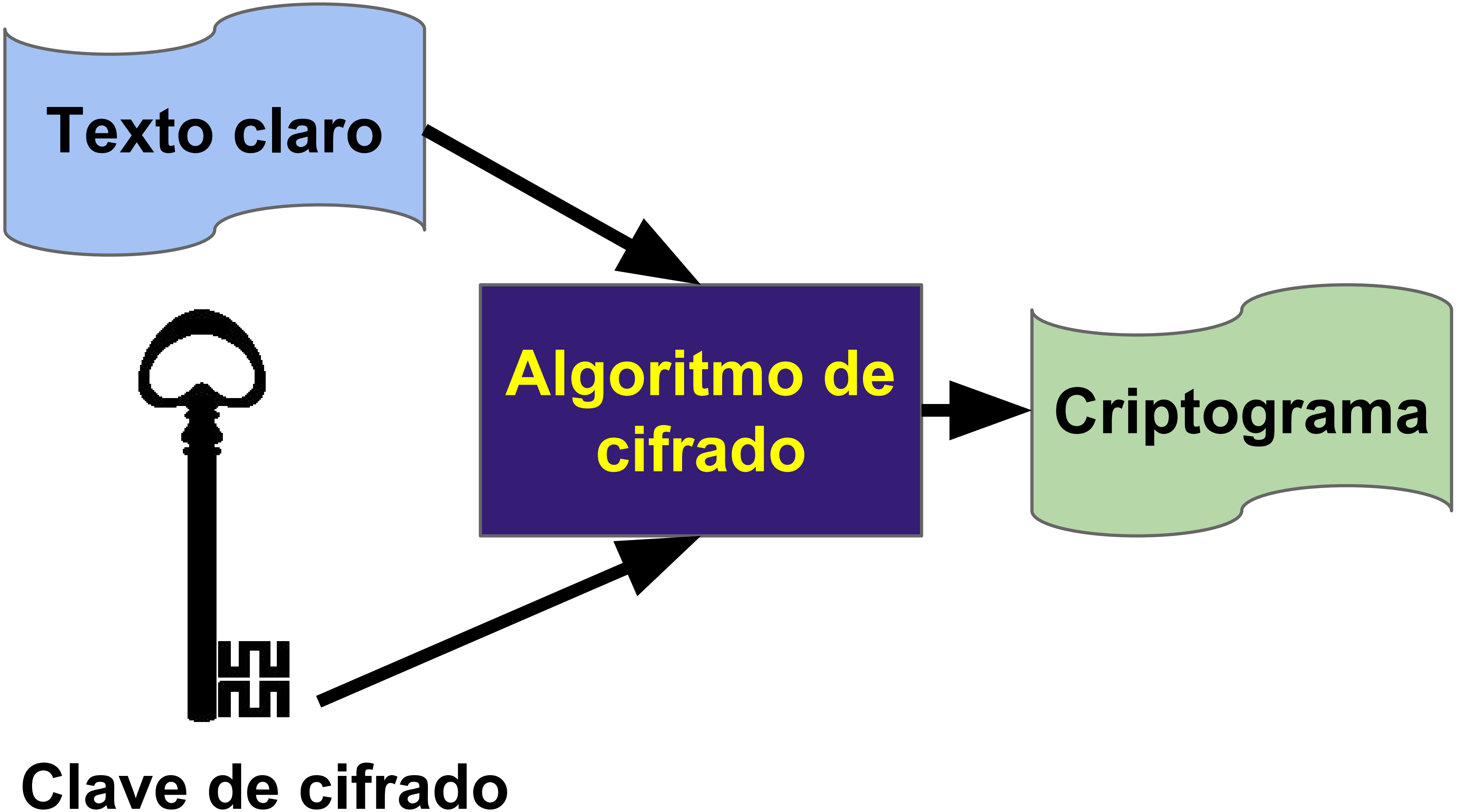
Shamir Secret Sharing Scheme

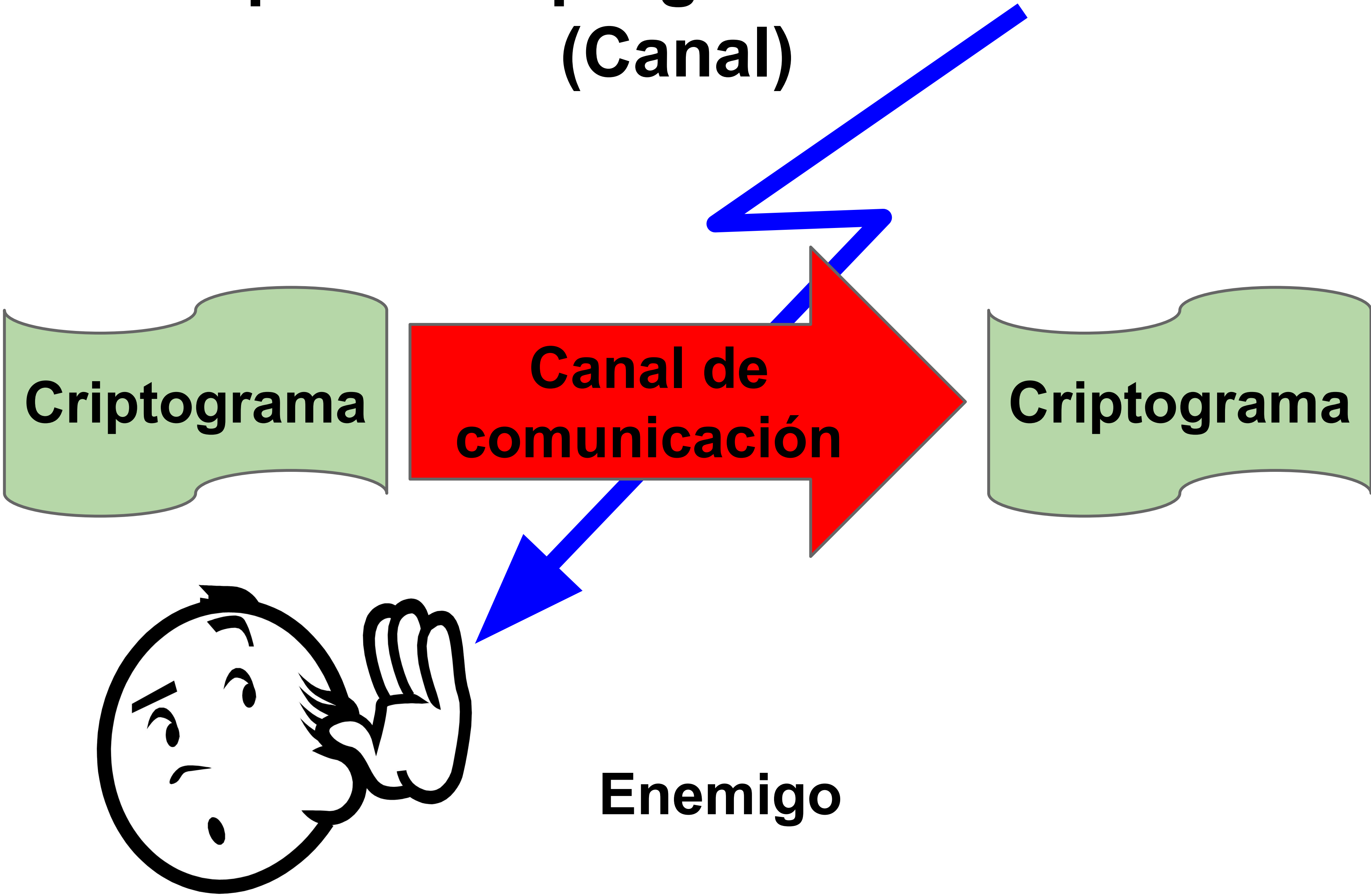
S^4

José Galaviz

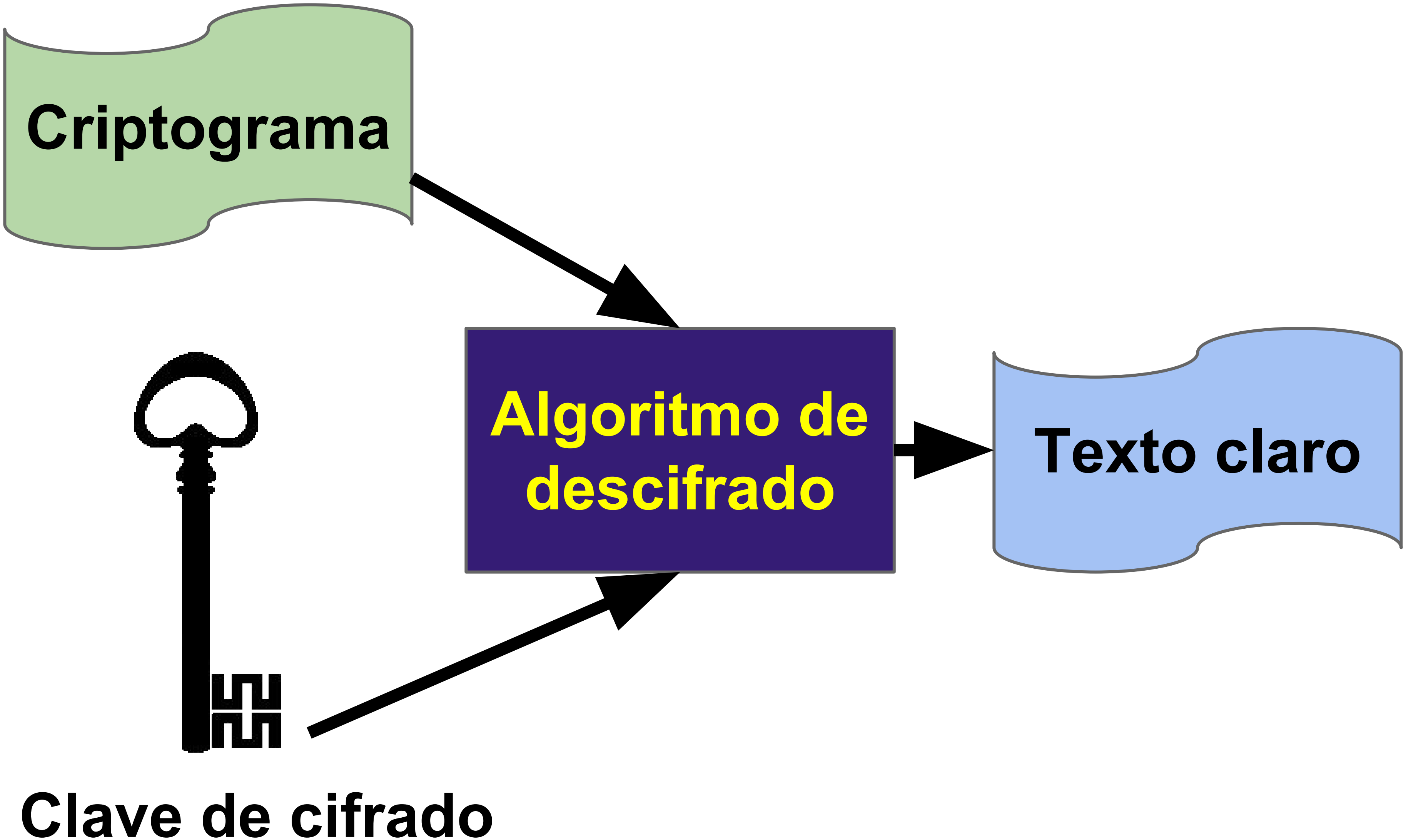
Esquema criptográfico simétrico (Emisor)

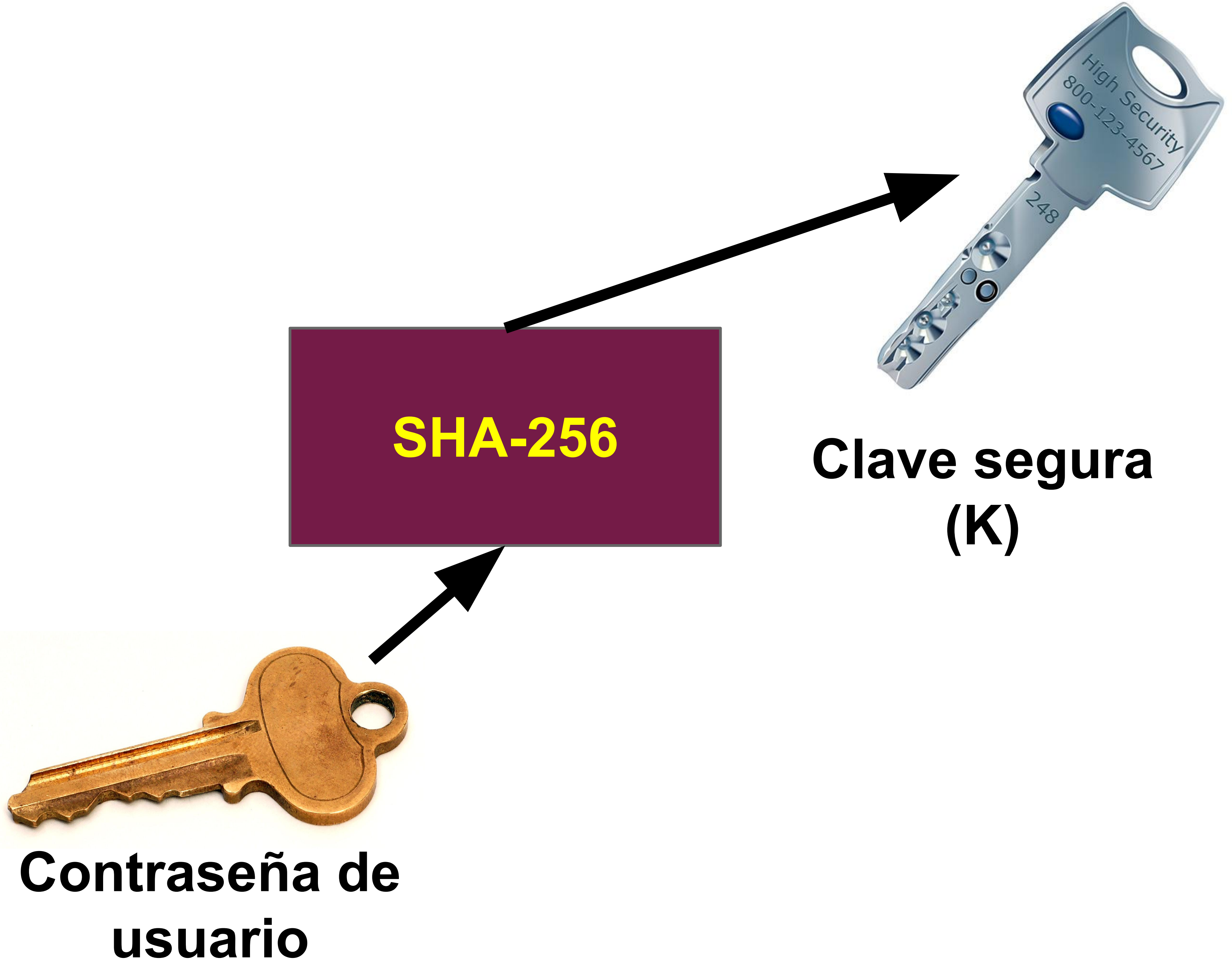


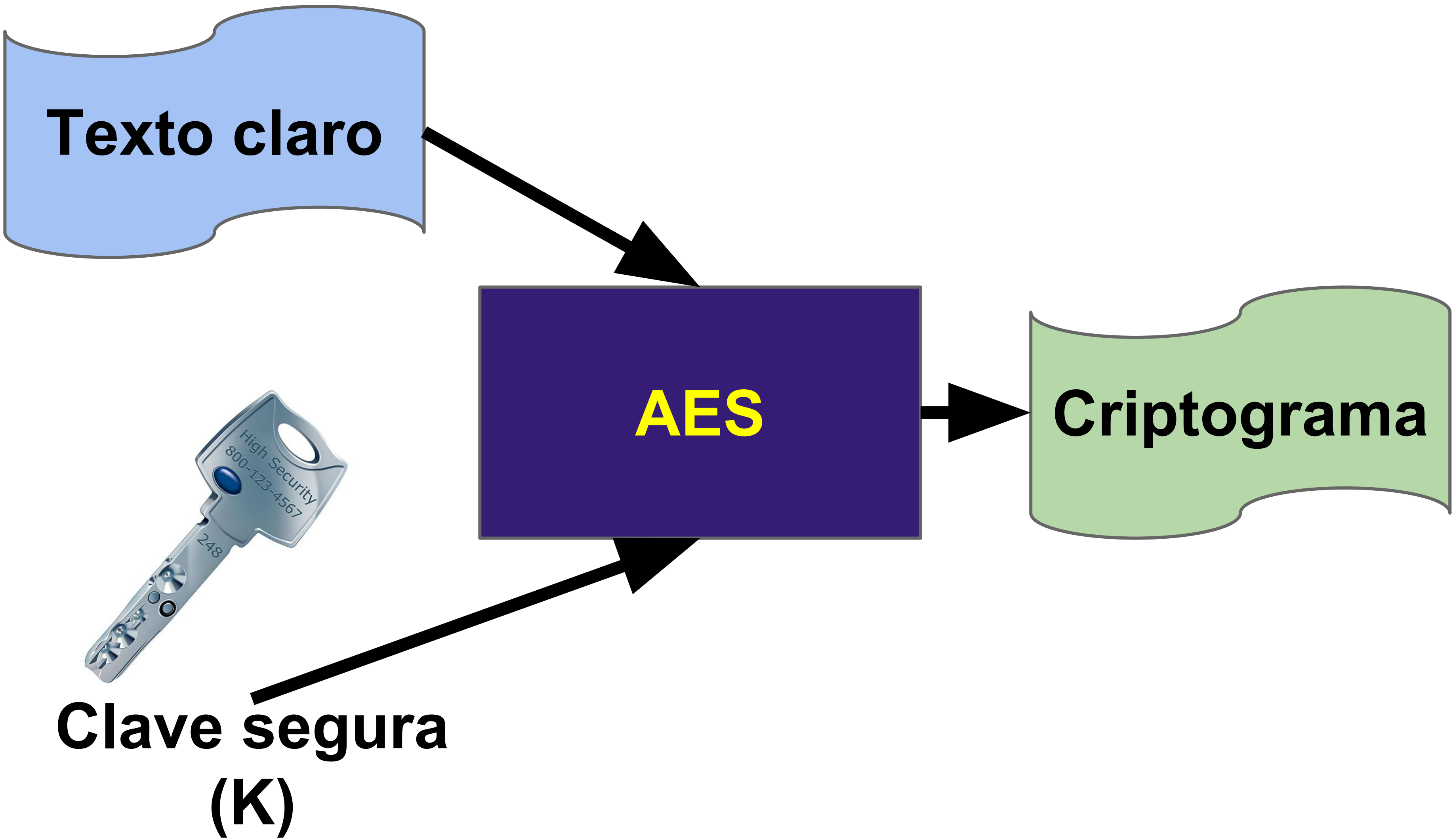
Esquema criptográfico simétrico (Canal)



Esquema criptográfico simétrico (Receptor)




SHA-256
High Security
800-123-4567
248
Clave segura
(K)
Contraseña de
usuario


Texto claro
AES
Criptograma
Clave segura
(K)

Esquema t de n

- Se requiere repartir a n diferentes personas:
  - Una copia del criptograma.
  - Un dato que contribuye a descifrar el criptograma (*share*).
- Se debe poder descifrar el criptograma teniendo al menos $t \leq n$ de los *shares* para descifrar.

Puntos y curvas

- Por un punto del plano pasan una infinidad no numerable de funciones.
- Por dos puntos dados del plano, pasa una y sólo una línea recta. .
- Por tres puntos del plano…

  .
- Por r puntos del plano…

  .

Puntos y curvas

- Por un punto del plano pasan una infinidad no numerable de funciones.
- Por dos puntos dados del plano, pasa una y sólo una línea recta. Polinomio de grado 1.
- Por tres puntos del plano… un único polinomio de grado 2.
- Por r puntos del plano… un único polinomio de grado $r - 1$.

$$P(x) = c_{t-1}x^{t-1} + c_{t-2}x^{t-2} + \cdots c_1 x + K$$

- $t\text{-}1$ coeficientes aleatorios.
- Polinomio de grado $t\text{-}1$.
- Reconstruible a partir de t puntos $\boldsymbol{(x_i, P(x_i))}$.
- Lo evaluamos en $n \geq t$ puntos.

Se reparten

- A la i-ésima persona se le da una copia del criptograma y...
- Una pareja (x_i, y_i) donde $y_i = P(x_i)$.

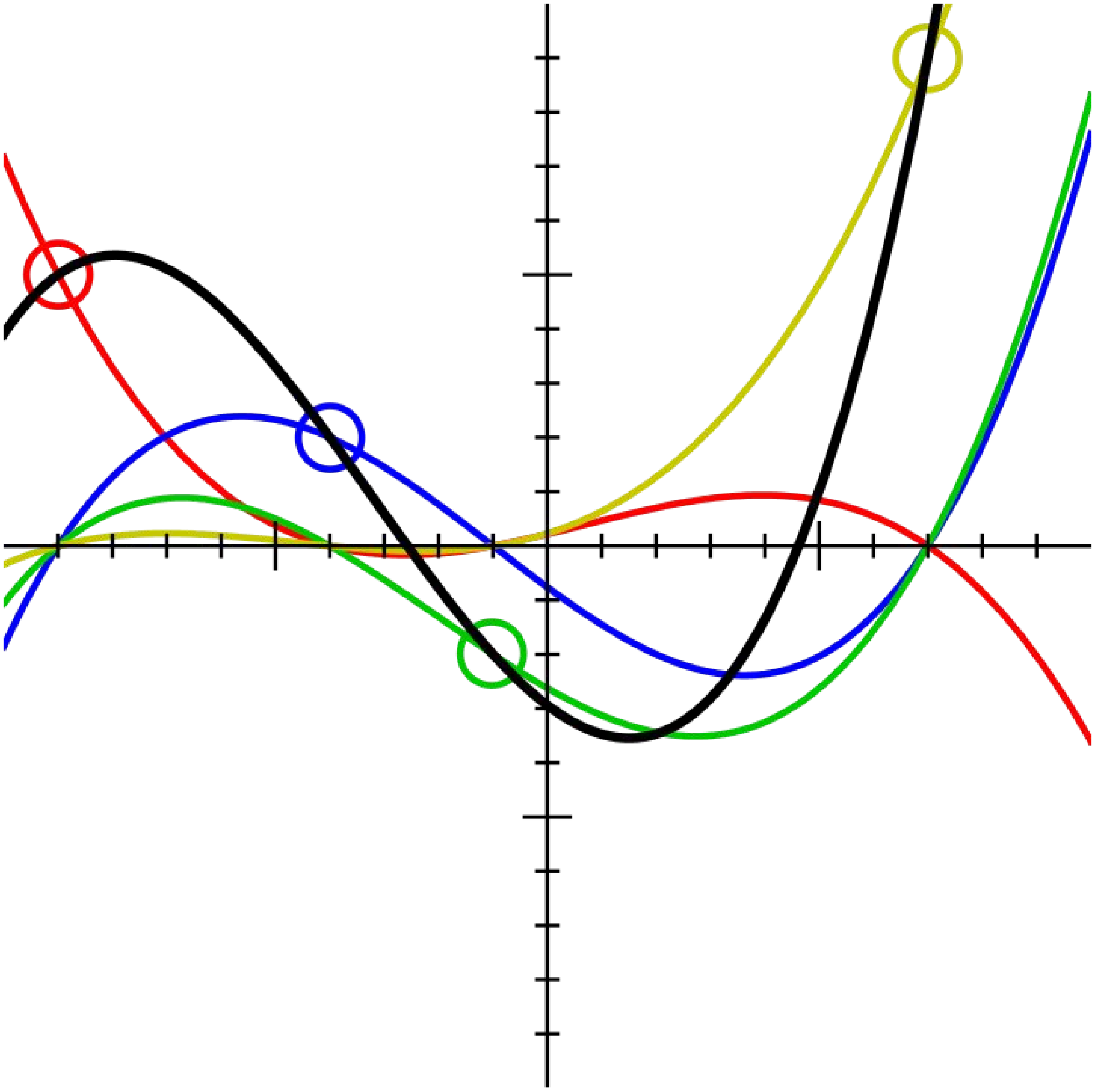
Para descifrar

- Se reúnen, al menos t de las n personas.
- Se toman sus parejas ordenadas $\boldsymbol{(x_i, P(x_i))}$.
- Se evalúa el polinomio de interpolación (de Lagrange) en cero.
- Como $P(0) = K$, esto recupera la clave segura.
- Se usa K para descifrar con AES el criptograma.

Polinomio de interpolación

- Dados los puntos con sus $t \leq r \leq n$ evaluaciones, hay un único polinomio.
- Diferentes maneras de calcularlo (Lagrange, Newton, Hermite).
- La de Lagrange-Euler-Waring es adecuada para nosotros porque queremos evaluarlo, no calcular analíticamente sus coeficientes.

Polinomios de base



Base de Lagrange

Se calculan los polinomios de la base:

$$P_i(x) = \prod_{\substack{j=1 \\ j \neq i}}^{r} \frac{x - x_j}{x_i - x_j}$$

Forma de Lagrange

Se evalúa en x :

$$P(x) = \sum_{i=1}^{r} y_i P_i(x)$$

Ejemplo

i	x	y
1	1	13
2	2	38
3	3	93
4	4	190

$$P(x) = 2x^3 + 3x^2 + 2x + 6$$

Polinomios de base: $P_1(x)$

$$P_1(x) = \frac{(x - x_2)(x - x_3)(x - x_4)}{(x_1 - x_2)(x_1 - x_3)(x_1 - x_4)}$$

$P_1(0)$ = (-2 * -3 * -4) / ((1 - 2)*(1 - 3)*(1 - 4)) = 4

Polinomios de base: $P_2(x)$

$$P_2(x) = \frac{(x-x_1)(x-x_3)(x-x_4)}{(x_2-x_1)(x_2-x_3)(x_2-x_4)}$$

$P_2(0)$ = (-1 * -3 * -4) / ((2 - 1)*(2 - 3)*(2 - 4)) = -6

Polinomios de base: $P_3(x)$

$$P_3(x) = \frac{(x-x_1)(x-x_2)(x-x_4)}{(x_3-x_1)(x_3-x_2)(x_3-x_4)}$$

$P_3(0)$ = (-1 * -2 * -4) / ((3 - 1)*(3 - 2)*(3 - 4)) = 4

Polinomios de base: $P_4(x)$

$$P_4(x) = \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_4-x_1)(x_4-x_2)(x_4-x_3)}$$

$$P_4(0) = (-1 * -2 * -3) \,/\, ((4 - 1)*(4 - 2)*(4 - 3)) = -1$$

El polinomio: *P(x)*

$$P(x) = y_1 P_1(x) + y_2 P_2(x) + y_3 P_3(x) + y_4 P_4(x)$$

P(0) = 13 * 4 + 38 * -6 + 93 * 4 + 190 * -1 =

52 - 228 + 372 - 190 = **6**

Warnings

- Hay que usar números de cientos de bits de tamaño (256), así que hay que usar bibliotecas para operar con ellos.
- Se requiere que los resultados sean números enteros. Así que habrá que trabajar en un campo finito.
- Se requiere entonces de operar siempre módulo algún primo grande, de 256 bits o cerca.
- Se requiere usar bibliotecas probadas para SHA-256 y AES.

```
/**
 * Número primo usado como el tamaño del
módulo en el campo finito que se
 * usará.
 */
#define MODSZ
"208351617316091241234326746312124448251235562226470491514186331217050270460481"
```

Caldwell, Chris K. y G.L. Honaker, "Prime Curios! The dictionary of prime number trivia", CreateSpace, 2009.

Prime Curios Database:

http://primes.utm.edu/curios/page.php?number_id=3746

78 decimal digits

257 binary digits

Evaluar un polinomio

- Hay que evaluar un polinomio (construido aleatoriamente).
- En n puntos elegidos aleatoriamente.
- Método eficiente: **Regla de Horner**.

Ejemplo

Evaluar:

$$P(x) = 2x^3 + 3x^2 + 2x + 6$$

Es lo mismo que hacer:

$$P(x) = [([(0)x + 2]x + 3)x + 2]x + 6$$

Algoritmo de Horner

Se evalúa un polinomio en un punto del dominio.
`x` *es el punto del dominio en el que se ha de evaluar.*
`n` *es el número de coeficientes (incluyendo el término independiente) del polinomio (i.e. su grado es n-1.*
`coefs` *es el arreglo de coeficientes, el índice del arreglo corresponde con la potencia de la variable independiente, así que en* `coefs[0]` *se encuentra el término independiente.*

```
EvaluaHorner(x, n, coefs[])
    result ← 0
    foreach i = {n-1, …, 0}     Cuenta en reversa
        result ← (result * x) + coefs[i]
    return result
```