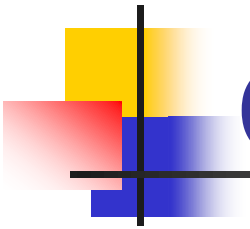
Bloque I
Criptografía

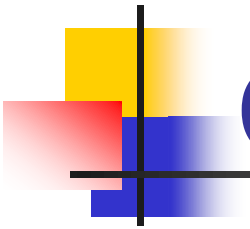
Cifrado en flujo

Seguridad en Redes de Comunicaciones
María Dolores Cano Baños



Contenidos

3.1 Cifrado en Flujo
3.2 RC4
3.3 A5



Cifrado en flujo

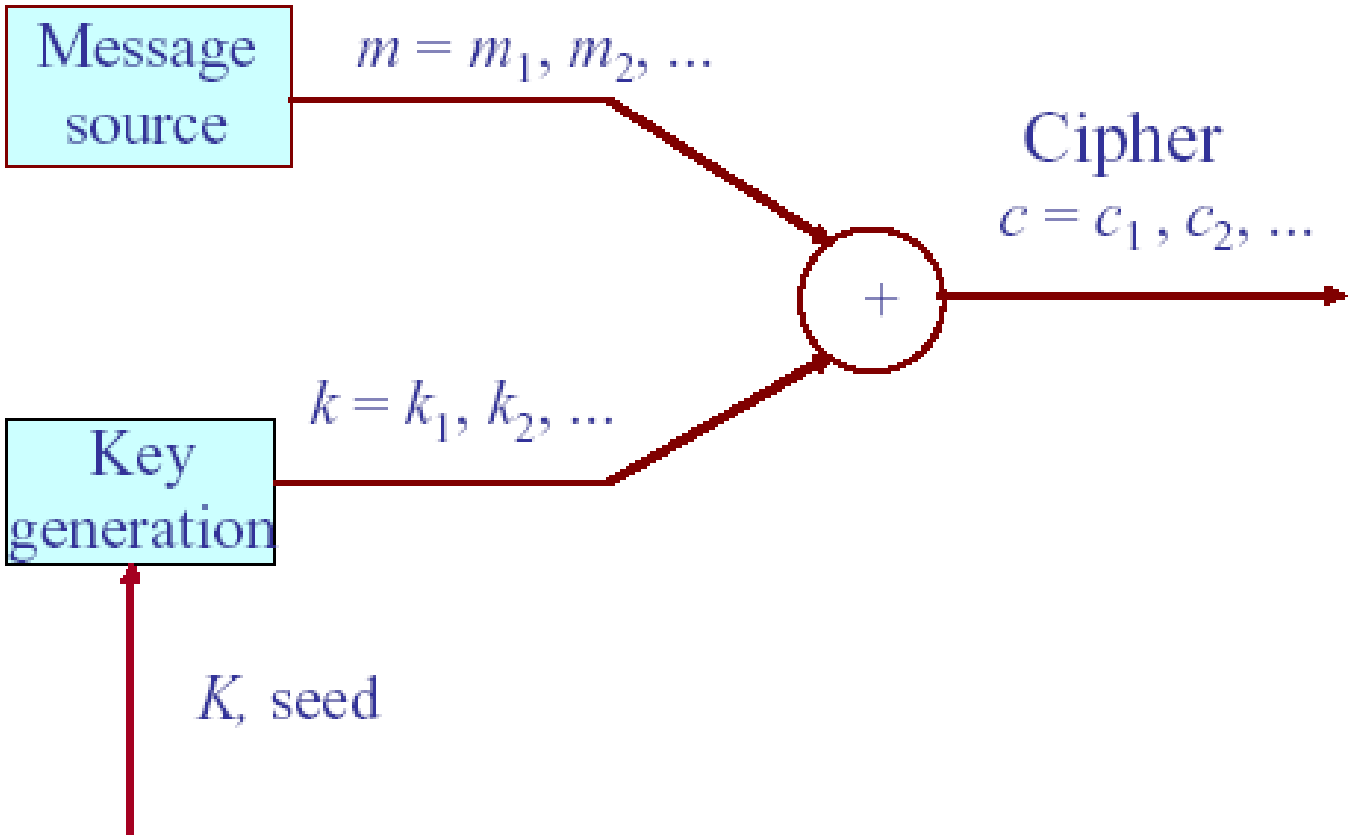
- El mensaje a cifrar NO se divide en bloques
- El cifrado en flujo cifra en tiempo real
- 1917 Mauborgne y Vernam inventaron primer criptosistema de cifrado en flujo:
  - Combinar carácter a carácter texto plano con una secuencia aleatoria de igual longitud utilizando una función simple y reversible (ej. XOR)
  - Enviada una única vez
  - Problema: la clave es tan larga como el propio mensaje y ¿cómo enviar la clave?

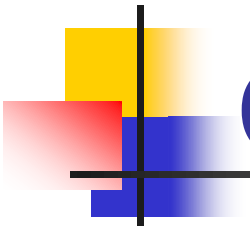
Cifrado en flujo

Generador pseudoaleatorio ⇨ secuencias criptográficamente aleatorias
+
Semilla del generador pseudoaleatorio = clave K

Cifrar: Secuencia pseudoaleatoria XOR con el texto plano
Descifrar: A partir de la semilla reconstruir secuencia pseudoaleatoria y hacer XOR con mensaje cifrado

Cifrado en flujo





Cifrado en flujo

- Veremos criptosistemas de clave privada
  - Especificación de un generador pseudoaleatorio
  - Combinación mediante la función XOR
  - Operaciones byte a byte

- Tipos de generadores:
  - Síncronos
  - Asíncronos

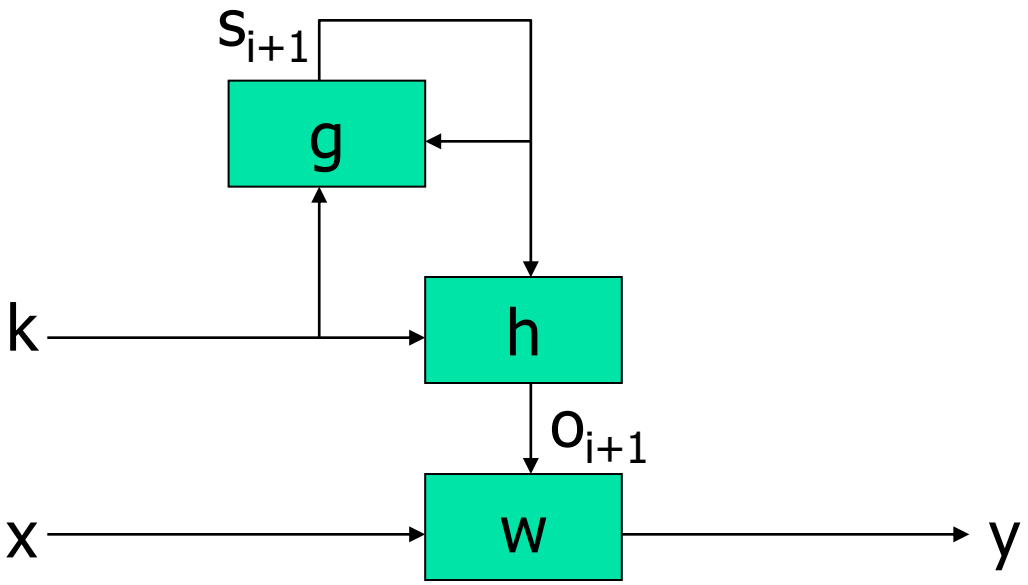
Cifrado en flujo

- **Generador síncrono**: La secuencia se calcula de modo independiente tanto del texto plano como del cifrado.

$s_{i+1} = g(s_i, k)$

$o_i = h(s_i, k)$

$y_i = w(x_i, o_i)$



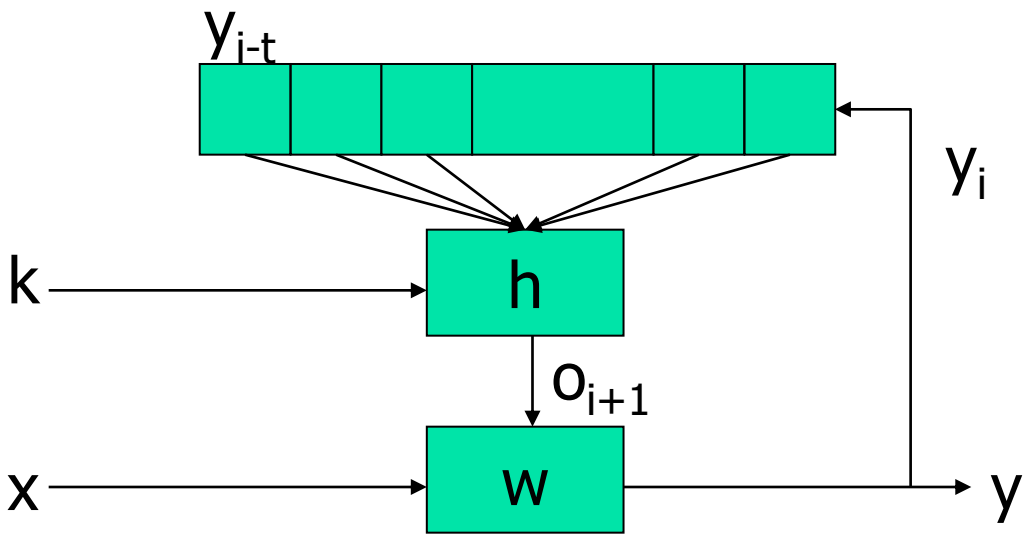
  - Emisor y receptor deben estar sincronizados (técnicas de verificación y de restablecimiento de sincronía)

Cifrado en flujo

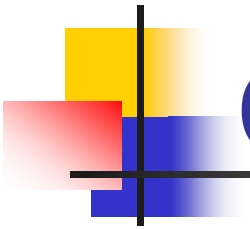
- **Generador asíncrono**: La secuencia generada es función de una semilla más una cantidad fija de los bits anteriores de la propia secuencia.

$o_i = h\ (K, y_{i-t}, y_{i-t+1}, y_{i-t+2}, \ldots, y_{i-1})$

$y_i = w\ (o_i, x_i)$



- Resistentes a pérdida o inserción de bits
- Sensibles a alteración de mensaje cifrado (técnicas de verificación)



Cifrado en flujo

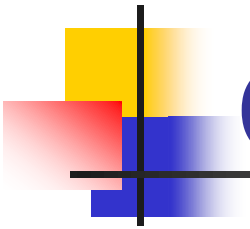
- Generadores de secuencias para cifrado en flujo basados en registros de desplazamiento retroalimentados (Feedback Shift Register):
  - Lineales
  - No lineales

Cifrado en flujo

- **Registros de Desplazamiento Retroalimentados Lineales** (Linear Feedback Shift Register, LFSR)
  - Conjunto de L estados $\{s_0, s_1, ..., s_{L-1}\}$, donde cada estado almacena 1 bit
  - Reloj controla variación de estados
  - Cada unidad de tiempo:
    - s_0 es la salida del registro
    - contenido de s_i se desplaza a s_{i-1} $(1 \leq i \leq L-1)$
    - contenido de s_{L-1}calculado como la suma en módulo 2 de los valores de un subconjunto prefijado del registro.

Cifrado en flujo

- **Registros de Desplazamiento Retroalimentados No Lineales** (Non Linear Feedback Shift Register, NLFSR)
  - Conjunto de L estados $\{s_0, s_1, \ldots, s_{L-1}\}$, donde cada estado almacena 1 bit
  - Reloj controla variación de estados
  - Cada unidad de tiempo:
    - s_0 es la salida del registro
    - contenido de s_i se desplaza a s_{i-1} $(1 \leq i \leq L-1)$
    - contenido de s_{L-1} calculado como una función boolena $f(s_{j-1}, s_{j-2}, \ldots, s_{j-L})$
- En general se usan *n* generadores lineales y una función *f* no lineal para combinar sus salidas: $f(R_1, R_2, \ldots, R_n)$

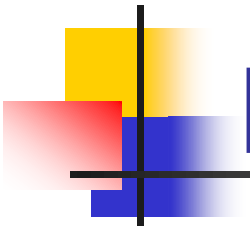


Contenidos

3.1 Cifrado en Flujo ✓

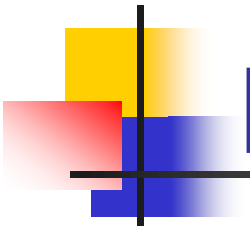
3.2 RC4

3.3 A5



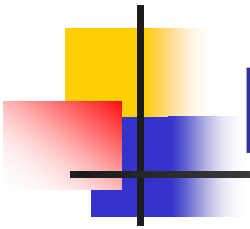
RC4

- Algoritmo de cifrado en flujo de clave privada diseñado por Ron Rivest (1987)
- Algoritmo propietario
- Implementación software
- Incluido en protocolos y estándares como WEP (Wired Equivalent Privacy) o SSL (Secure Socket Layer)
- Clave de hasta 256 bits (típicamente 40-256 bits)



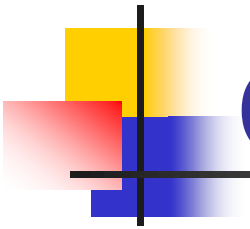
RC4

- Cifrado byte a byte
- Operaciones de cifrado/descifrado: emplea 256 bytes de memoria (s[0] a s[255]) y 3 variables i, j, k.
- Estado inicial:
  - 1) s[i]=i $\forall$i 0 ≤i ≤ 255
  - 2) j=0
  - 3) Para i=0 hasta 255 hacer:
    - j=(j+s[i]+key[i mod key_length]) mod 256
    - Intercambiar s[i] y s[j]



RC4

- Cifrar/descifrar:
  - 1) i=0; j=0
  - 2) para cada byte a cifrar
    - i=(i+1) mod 256
    - j=(j+s[i]) mod 256
    - Intercambiar s[i] y s[j]
    - k=(s[i]+s[j]) mod 256
    - XOR de s[k] con siguiente byte de entrada
- Desechar los primeros bytes de salida del generador y no usarlos para cifrar
- Otras aplicaciones: Lotus Notes, cifrado de claves en windows, MS Access, Adobe Acrobat, Oracle Secure Server, etc.

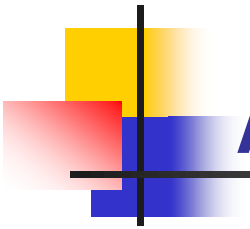


Contenidos

3.1 Cifrado en Flujo ✓

3.2 RC4 ✓

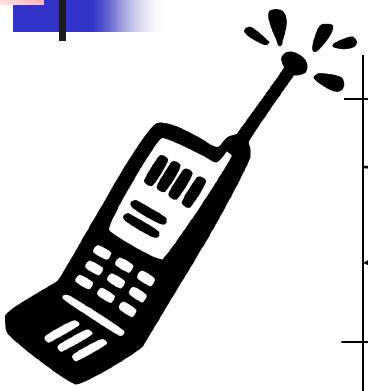
3.3 A5



A5

- GSM desarrollado por Instituto Europeo de Estándares de Telecomunicaciones ⇨ protocolos criptográficos para confidencialidad y autenticación
- A3 algoritmo de autenticación
- A5 algoritmo de cifrado de voz
- A8 algoritmo generador de claves
- COMP128 algoritmo para ejecutar A3 y A8

A5



A8; K_i de la SIM

¿datos aleatorios?

D_1=datos aleatorios

A3 (K_i+D_1) -> resultado$_1$

resultado$_1$

A3 (K_i de la E.B.+D_1) -> resultado$_2$

¿resultado$_1$= resultado$_2$? SI

comunicación libre

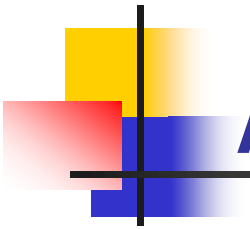
A8; K_i de la SIM

¿datos aleatorios?

D_2=datos aleatorios

A8(K_i+D_2) =K_c

A8(K_i+D_2) =K_c

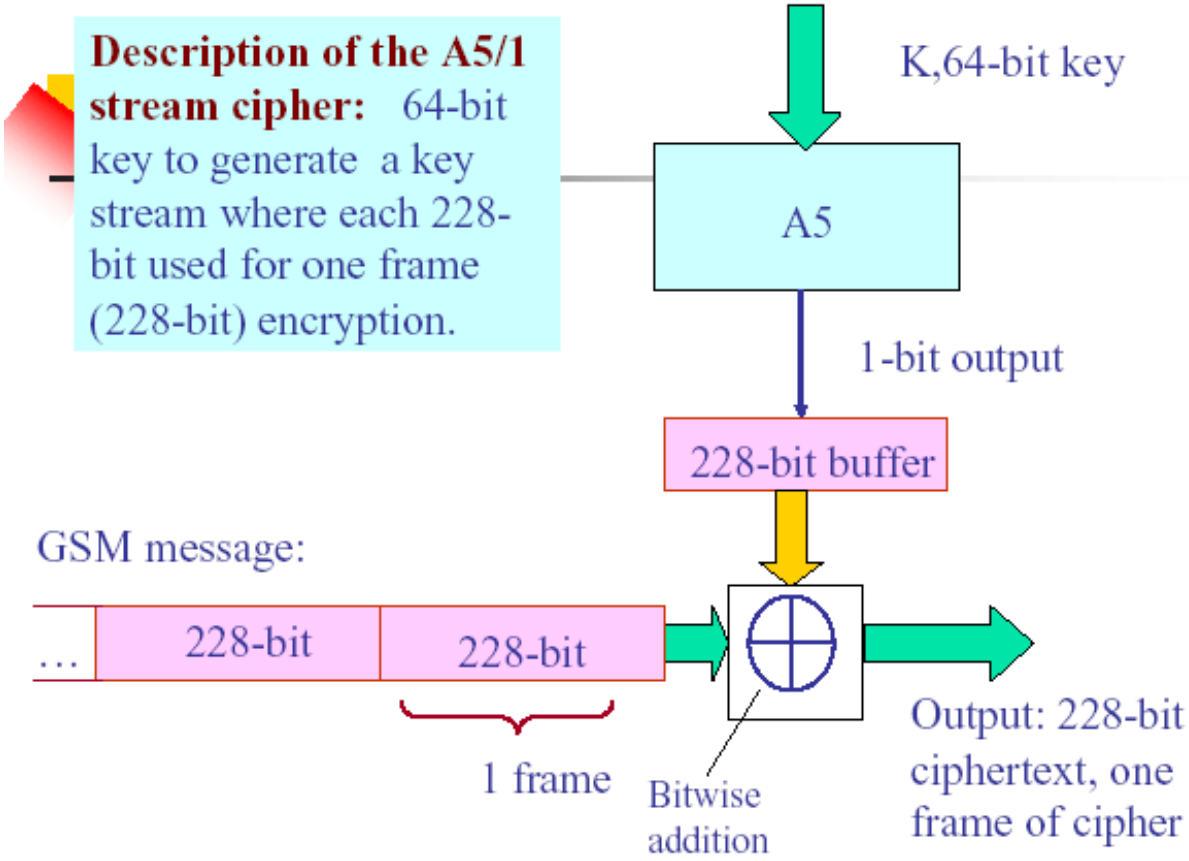


A5

- Proporciona privacidad a las comunicaciones GSM en el interfaz radio (GSM 1 trama cada 4,6 ms; 1 trama 228 bits)
- Dos versiones A5/1 y A5/2
- Ambas son una combinación de tres registros de desplazamiento retroalimentados lineales con señales de reloj irregulares y un combinador no lineal

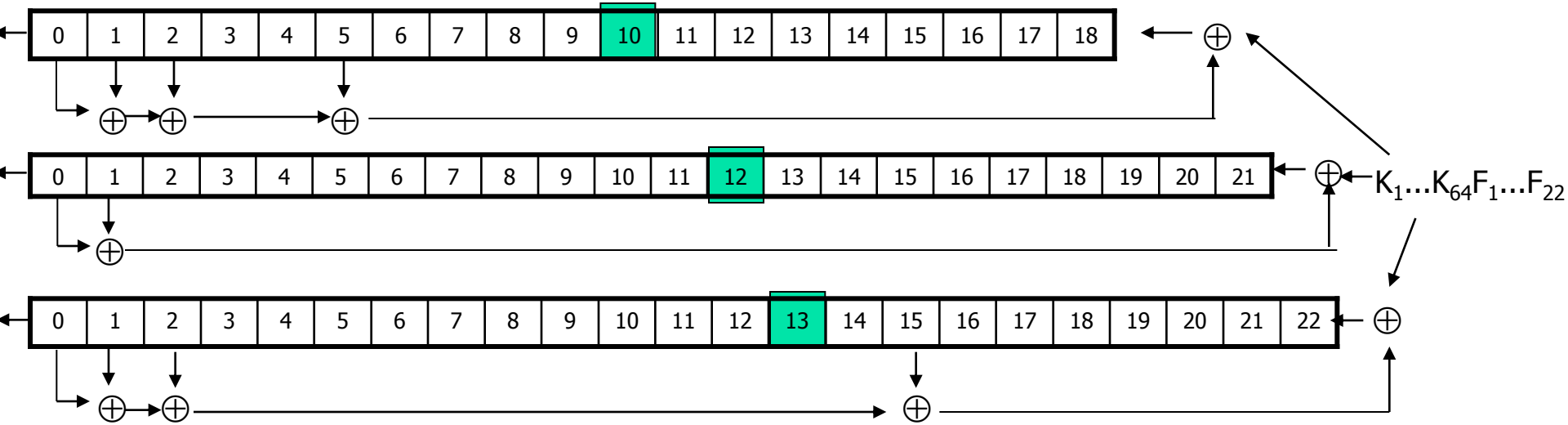
A5

- **A5/1** usa clave secreta de 64 bits y genera secuencia de bits (cada 228 bits se cifra una trama)



A5

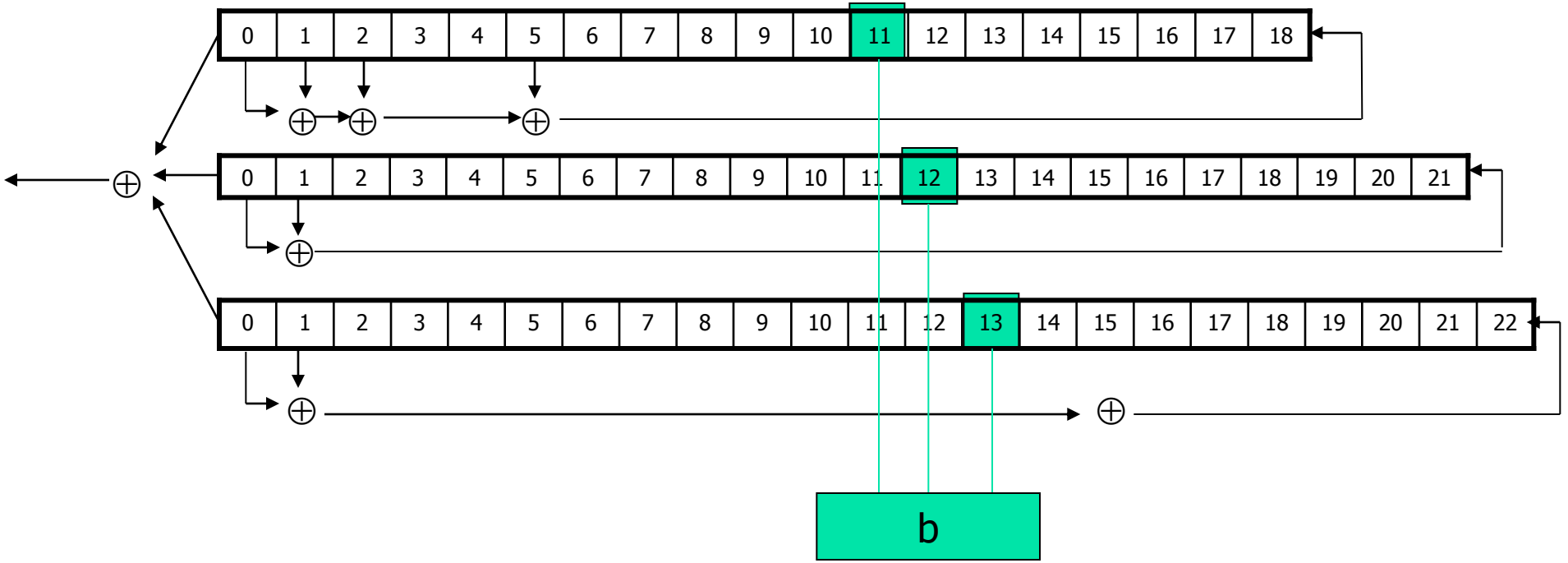
- 3 LFSR de longitudes 19, 22 y 23 bits ($x^{19}+x^5+x^2+x+1$; $x^{22}+x+1$; $x^{23}+x^{15}+x^2+x+1$)
- Inicialización cada trama: desde t=1 hasta t=64 se usa K, desde t=65 hasta t=89 se usa el bit (t-64) del número de trama F

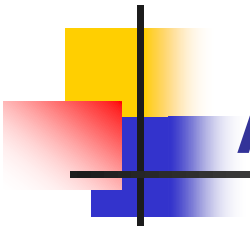


PROCESO DE INICIALIZACIÓN

A5

- Cada LFSR tiene un *tap* de reloj
- Se calcula el valor mayoría (b) de los tres *taps* cada unidad de tiempo
- LFSR recibe señal de reloj sólo si su *tap* coincide con valor b





A5

- Ataques:
  - Por fuerza bruta
  - Goldberg,Wagner, Briceno => “de los 64 bits de la clave diez de ellos son siempre cero”
  - Briceno => ingeniería inversa en Diciembre 1999
  - Biryukov, Shamir => “Real time criptoanalysis of A5/1 on a PC”, 1PC con 128 Mb RAM, 2-4 discos duros de 73 Gb cada uno, escáner digital.
- 3G
  - Generación de claves: MILENAGE
  - Confidencialidad e integridad en interfaz radio: KASUMI