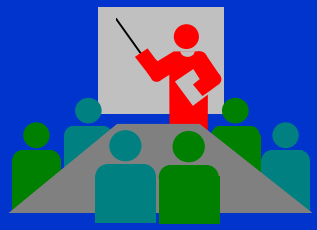
Capítulo 11
Sistemas de Cifra en Flujo

Seguridad Informática y Criptografía



Material Docente de
Libre Distribución

Ultima actualización del archivo: 01/03/06
Este archivo tiene: 51 diapositivas

Dr. Jorge Ramió Aguirre
Universidad Politécnica de Madrid

Cifrador de flujo básico

- Recordando la propuesta de cifrador hecha por Vernam en 1917, los cifradores de flujo (sistemas con clave secreta) usarán:
  - Un algoritmo de cifra basado en la función XOR.
  - Una secuencia cifrante binaria y pseudoaleatoria denominada S y que se obtiene a partir una clave secreta K compartida por emisor y receptor, y un algoritmo generador determinístico.
  - El mismo algoritmo para el descifrado debido el carácter involutivo de la función XOR.

Características de la secuencia cifrante S

Condiciones para una clave binaria segura

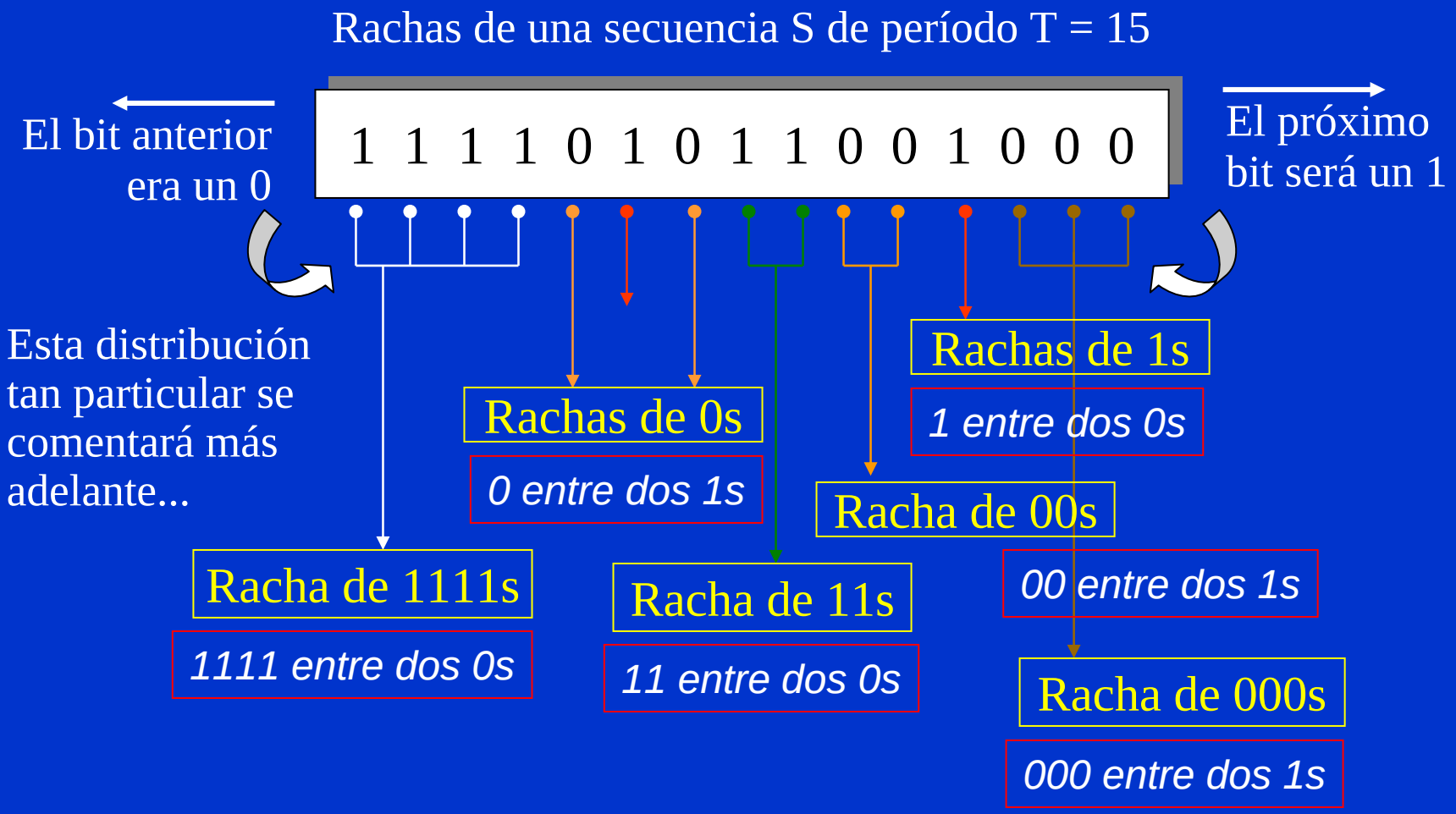
- Período:
  - La clave deberá ser tanto o más larga que el mensaje. En la práctica se usará una semilla K de unos 120 a 250 bits en cada extremo del sistema para generar períodos superiores a 10^{35}.
- Distribución de bits:
  - La distribución de bits de unos (1s) y ceros (0s) deberá ser uniforme para que represente a una secuencia pseudoaleatoria. Para ello deberá cumplir los postulados de Golomb:

Rachas de dígitos: uno o más bits entre dos bits distintos.

Función de autocorrelación fuera de fase AC(k): desplazamiento de k bits sobre la misma secuencia S_i.

http://ee.usc.edu/faculty_staff/bios/golomb.html

Rachas de dígitos en una secuencia



Distribución de las rachas de dígitos

Las rachas, es decir la secuencia de dígitos iguales entre dos dígitos distintos, deberán seguir una distribución estadística de forma que la secuencia cifrante S_i tenga un comportamiento de clave aleatoria o pseudoaleatoria.

Para que esto se cumpla, es obvio que habrá mayor número de rachas cortas que de rachas largas como se observa en el ejemplo anterior.

Como veremos más adelante, esta distribución seguirá una progresión geométrica. Por ejemplo una secuencia S_i podría tener 8 rachas de longitud uno, 4 de longitud dos, 2 de longitud tres y 1 de longitud cuatro.

Autocorrelación fuera de fase AC(k)

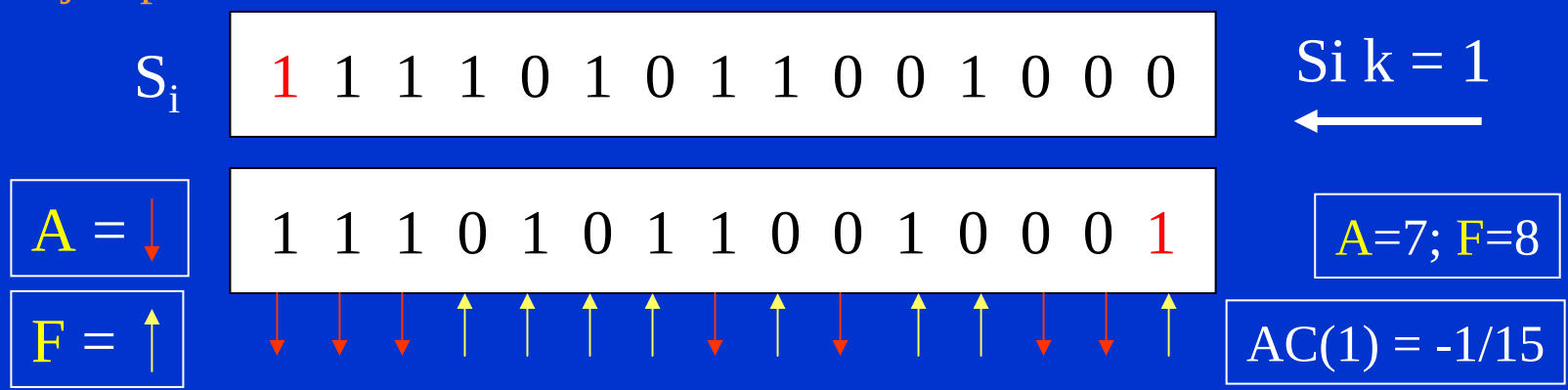
Función de autocorrelación:

– Autocorrelación AC(k) fuera de fase de una secuencia S_i de período T desplazada k bits a la izquierda:

$$AC(k) = (A - F) / T$$

Aciertos ⇒ bits iguales Fallos ⇒ bits diferentes

Ejemplo



Autocorrelación fuera de fase constante

S_i | 1 1 1 1 0 1 0 1 1 0 0 1 0 0 0

Como ejercicio, compruebe que para esta secuencia cifrante S_i la autocorrelación fuera de fase AC(k) para todos los valores de k ($1 \leq k \leq 14$) es constante e igual a -1/15. Esta característica será importante para que la clave sea considerada buena.

Es decir, para que una secuencia cifrante S podamos considerarla segura y apropiada para una cifra, además de cumplir con la distribución de rachas vista anteriormente, deberá presentar una autocorrelación fuera de fase AC(k) constante.

Imprevisibilidad e implementación de S_i

- Imprevisibilidad:
  - Aunque se conozca una parte de la secuencia S_i, la probabilidad de predecir el próximo dígito no deberá ser superior al 50%.
  - Esta característica se definirá a partir de la denominada complejidad lineal.
- Facilidad de implementación:
  - Debe ser fácil construir un generador de secuencia cifrante con circuitos electrónicos y chips, con bajo coste, alta velocidad, bajo consumo, un alto nivel de integración, etc.

Primer postulado de Golomb G1

Postulado G1:

– Deberá existir igual número de ceros que de unos. Se acepta como máximo una diferencia igual a la unidad.

S_1 | 1 1 1 1 0 1 0 1 1 0 0 1 0 0 0 |

En la secuencia S_1 de 15 bits, hay 8 unos y 7 ceros. Luego sí cumple con el postulado G1.

S_2 | 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 |

En la secuencia S_2 de 16 bits, hay 7 unos y 9 ceros. Luego no cumple con el postulado G1.

Significado del postulado G1

S_i | 1 1 1 1 0 1 0 1 1 0 0 1 0 0 0

¿Qué significa esto?

Si una secuencia S_i como la indicada cumple con G1, quiere decir que la probabilidad de recibir un bit 1 es igual a la de recibir un bit 0, es decir un 50%.

Por lo tanto, a lo largo de una secuencia S_i, independientemente de los bits recibidos con anterioridad, en media será igualmente probable recibir un "1" que un "0", pues en la secuencia hay una mitad de valores "1" y otra mitad de valores "0".

Segundo postulado de Golomb G2

Postulado G2:

- En un período T, la mitad de las rachas de S_i serán de longitud 1, la cuarta parte de longitud 2, la octava parte de longitud 3, etc.

S_i	1 1 1 1 0 1 0 1 1 0 0 1 0 0 0

Las rachas de esta secuencia están en una diapositiva anterior

En la secuencia S_i de 15 bits, había 4 rachas de longitud uno, 2 rachas de longitud dos, 1 racha de longitud tres y 1 racha de longitud cuatro. Este tipo de distribución en las rachas para períodos impares, es típica de las denominadas *m-secuencias* como veremos más adelante en el apartado generadores LFSR.

Significado del postulado G2

S_i | 1 1 1 1 0 1 0 1 1 0 0 1 0 0 0

¿Qué significa esto?

Si una secuencia S_i como la indicada cumple con G2, quiere decir que la probabilidad de recibir un bit 1 o un bit 0, después de haber recibido un 1 o un 0 es la misma, es decir un 50%.

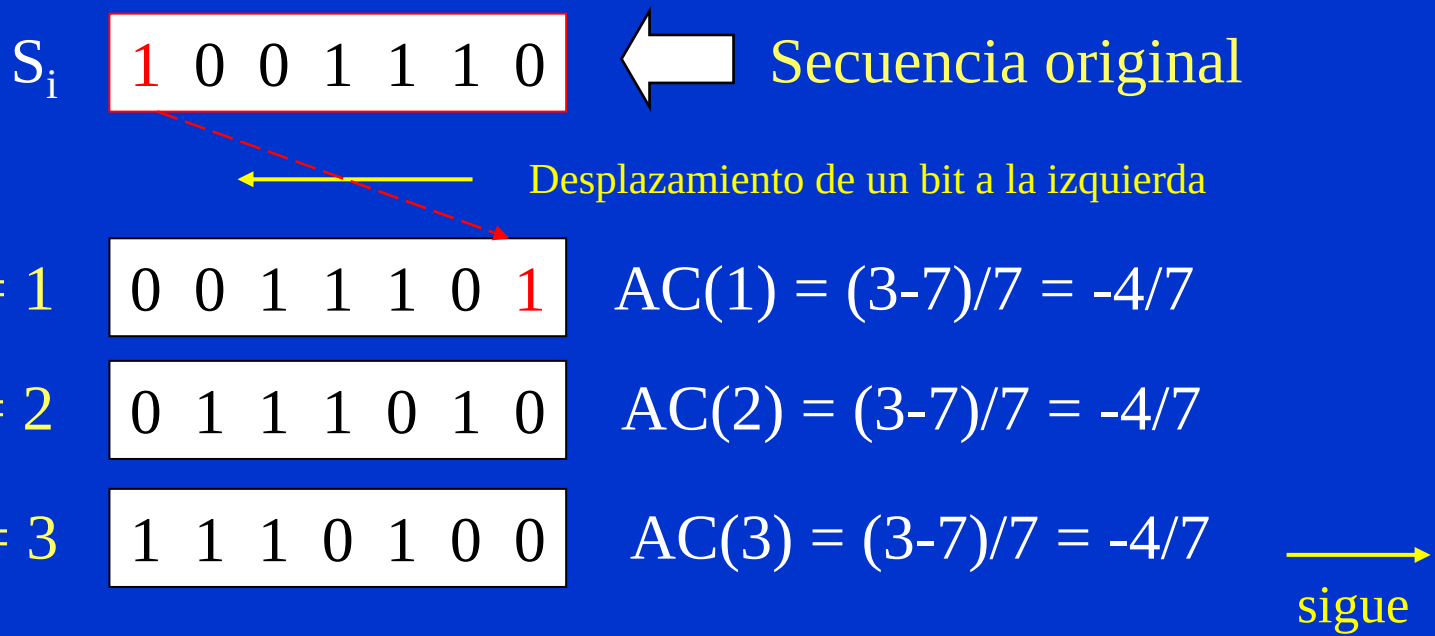
Es decir, recibido por ejemplo un "1", la cadena "10" deberá ser igual de probable que la cadena "11". Lo mismo sucede con un 0 al comienzo, o bien un 00, 01, 10, 11, 000, 001, etc. Existirá así también una equiprobabilidad en función de los bits ya recibidos.

Como comprobaremos más adelante, esto va a significar que la secuencia pasa por todos sus estados, es decir todos sus restos.

Tercer postulado de Golomb G3 (1/2)

Postulado G3:

- La autocorrelación AC(k) deberá ser constante para todo valor de desplazamiento de k bits.



sigue

Tercer postulado de Golomb G3 (2/2)

	1 0 0 1 1 1 0	Secuencia original
k = 4	1 1 0 1 0 0 1	AC(4) = (3-7)/7 = -4/7
k = 5	1 0 1 0 0 1 1	AC(5) = (3-7)/7 = -4/7
k = 6	0 1 0 0 1 1 1	AC(6) = (3-7)/7 = -4/7
k = 7	1 0 0 1 1 1 0	Secuencia original en fase

La secuencia S_i = 1001110 de 7 bits cumple con G3

Autocorrelación no constante (1/2)

S_i | 0 1 1 1 0 1 0 0 | ⇐ Secuencia original

⟵ Desplazamiento de un bit a la izquierda

k = 1	1 1 1 0 1 0 0 0	AC(1) = (4-4)/8 = 0
k = 2	1 1 0 1 0 0 0 1	AC(2) = (4-4)/8 = 0
k = 3	1 0 1 0 0 0 1 1	AC(3) = (2-6)/8 = -1/2
k = 4	0 1 0 0 0 1 1 1	AC(4) = (4-4)/8 = 0

⟶ sigue

Autocorrelación no constante (2/2)

S_i	0 1 1 1 0 1 0 0	Secuencia original
k = 5	1 0 0 0 1 1 1 0	AC(5) = (2-6)/8 = -1/2
k = 6	0 0 0 1 1 1 0 1	AC(6) = (4-4)/8 = 0
k = 7	0 0 1 1 1 0 1 0	AC(7) = (4-4)/8 = 0
k = 8	0 1 1 1 0 1 0 0	Secuencia original en fase

La secuencia S_i = 01110100 de 8 bits no cumple con G3

Significado del postulado G3

S_i | 0 1 1 1 0 1 0 0 | No cumple con G3

S_i | 1 0 0 1 1 1 0 | Sí cumple con G3

¿Qué significa esto?

Si una secuencia cumple con el postulado G3 quiere decir que, independientemente del trozo de secuencia elegido por el atacante, no habrá una mayor cantidad de información que en la secuencia anterior. Así, será imposible aplicar ataques estadísticos a la secuencia recibida u observada al igual como operábamos, por ejemplo y guardando las debidas distancias, con el sistema Vigenère y el ataque de Kasiski.

Generador de congruencia lineal

$x_{i+1} = (a*x_i \pm b)(\text{mod } n)$ será la secuencia cifrante

- Los valores a, b, n caracterizan al generador y se utilizarán como clave secreta.
- El valor x_0 se conoce como semilla; es el que inicia el proceso generador de la clave X_i.
- La secuencia se genera desde i = 0 hasta i = n-1.
- Tiene como debilidad que resulta relativamente fácil atacar la secuencia, de forma similar al criptoanálisis de los cifradores afines vistos en criptografía clásica.

Ejemplo generador de congruencia lineal

Sea:
a = 5 b = 1
n = 16 $x_0 = 10$

$$x_{i+1} = (a*x_i \pm b)(\text{mod } n)$$

Pero...

$S_i = 10, 3, 0, 1, 6, 15, 12, 13, 2, 11, 8, 9, 14, 7, 4, 5$

$x_1 = (5*10+1) \text{ mod } 16 = 3$
$x_2 = (5*3+1) \text{ mod } 16 = 0$
$x_3 = (5*0+1) \text{ mod } 16 = 1$
$x_4 = (5*1+1) \text{ mod } 16 = 6$
$x_5 = (5*6+1) \text{ mod } 16 = 15$
$x_6 = (5*15+1) \text{ mod } 16 = 12$
$x_7 = (5*12+1) \text{ mod } 16 = 13$
$x_8 = (5*13+1) \text{ mod } 16 = 2$
$x_9 = (5*2+1) \text{ mod } 16 = 11$
$x_{10} = (5*11+1) \text{ mod } 16 = 8$
$x_{11} = (5*8+1) \text{ mod } 16 = 9$
$x_{12} = (5*9+1) \text{ mod } 16 = 14$
$x_{13} = (5*14+1) \text{ mod } 16 = 7$
$x_{14} = (5*7+1) \text{ mod } 16 = 4$
$x_{15} = (5*4+1) \text{ mod } 16 = 5$
$x_{16} = (5*5+1) \text{ mod } 16 = 10$

¿Algo falla en este tipo de generador?

$$x_{i+1} = (a * x_i \pm b) (\mathrm{mod}\ n)$$

Ejercicios

¿Qué sucede si
a = 11 b = 1
n = 16 x_0 = 7?

¿Qué sucede si
a = 5 b = 2
n = 16 x_0 = 10?

¿Qué sucede si
a = 5 b = 2
n = 16 x_0 = 1?

¿Qué sucede si
a = 4 b = 1
n = 16 x_0 = 10?

Saque sus propias conclusiones.

Como habrá comprobado, este tipo de generadores de secuencia cifrante no son criptográficamente nada interesantes.

Una vez hecho esto personalmente, pase a la siguiente diapositiva.

Debilidad en este tipo de generadores

$S_i = (11*7 + 1) \bmod 16$

$S_i = 15, 7$

El período que se genera es sólo de tamaño dos ... ☹

$S_i = (5*10 + 2) \bmod 16$

$S_i = 4, 6, 0, 2, 12, 14, 8, 10$

$S_i = (5*1 + 2) \bmod 16$

$S_i = 7, 5, 11, 9, 15, 13, 3, 1$

Se obtiene un período muy bajo y sólo valores pares e impares. El primer caso es igual que el de los apuntes pero con b = 2 ... ☹ ☹

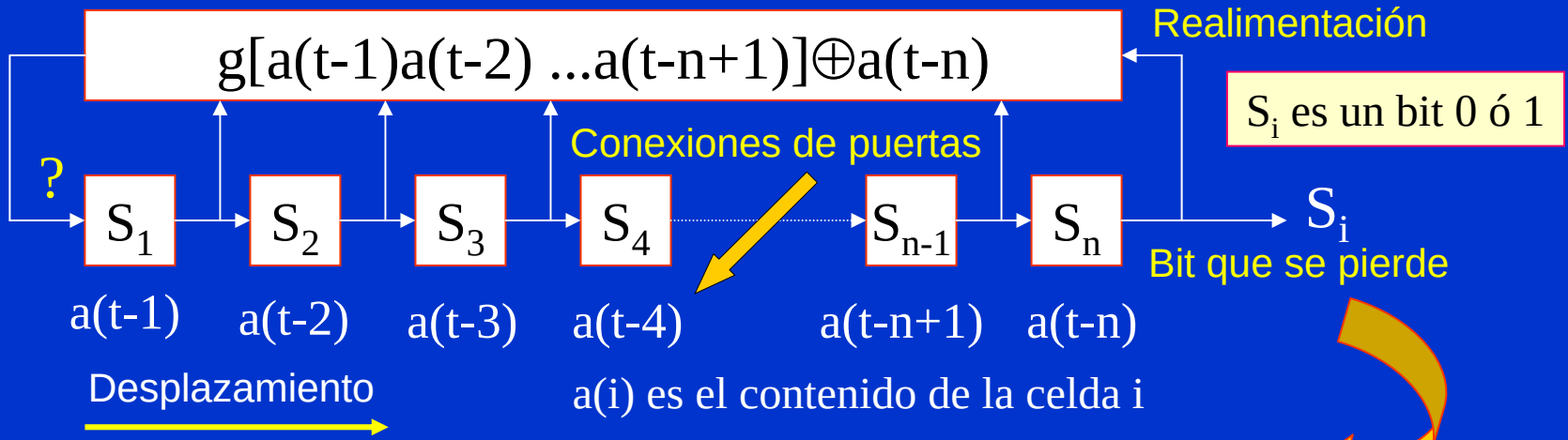
$S_i = (4*10 + 1) \bmod 16$

$S_i = 9, 5, 5, ...$

Peor aún, ya no se genera una secuencia ... ☹ ☹ ☹

Registros de desplazamiento

Generador de secuencia cifrante con registros de desplazamiento



Genera una secuencia con un período máximo 2^n

NLFSR

Registros de Desplazamiento Realimentados No Linealmente

Registros de Desplazamiento Realimentados Linealmente

LFSR

Por claridad se mantendrá la nomenclatura inglesa

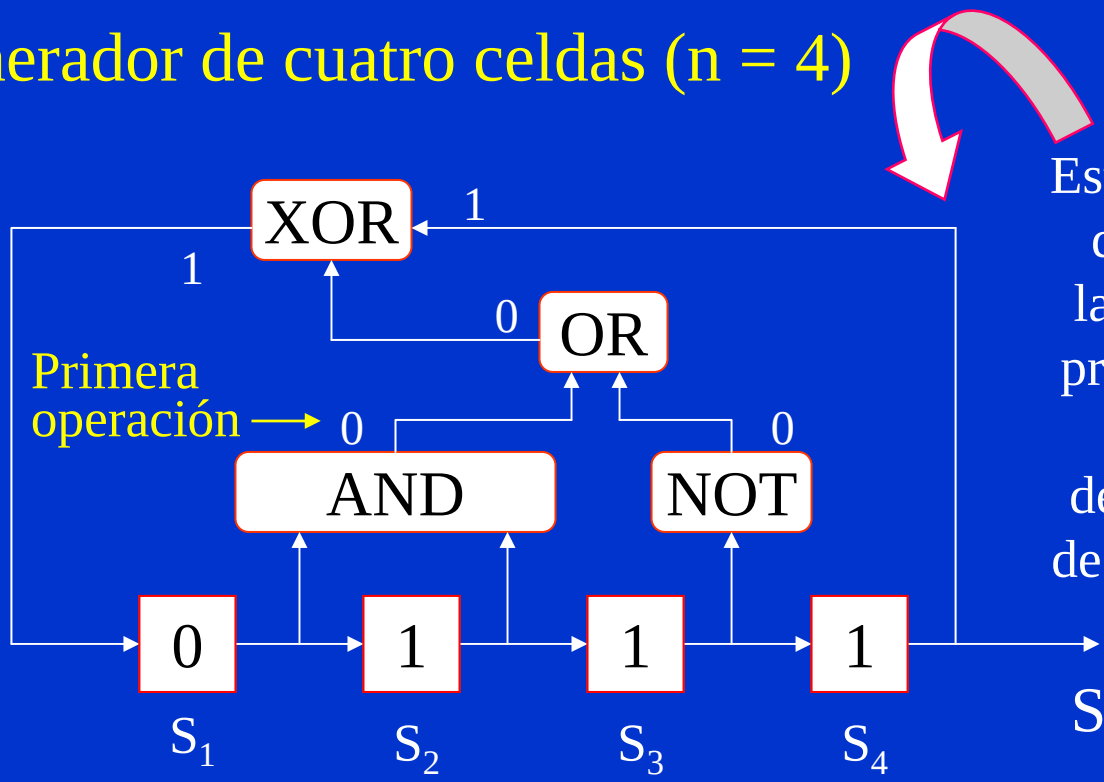
Introducción a los autómatas celulares

- Los registros de desplazamiento son un caso especial de los denominados autómatas celulares finitos unidimensionales.
- Este autómata celular finito es un sistema dinámico con un total de N células, dispuestas en un espacio unidimensional.
- Cada célula tendrá en cada instante un estado E y existirá una función de transición f que, dependiendo de una vecindad establecida entre las células, hará que en cada instante de tiempo dicho autómata evolucione.
- En criptografía interesan los autómatas celulares que sean reversibles, es decir, que permitan la evolución hacia atrás.

http://www.criptored.upm.es/investigacion/tfc_m317a.htm

Generadores NLFSR (1/2)

Un generador de cuatro celdas (n = 4)



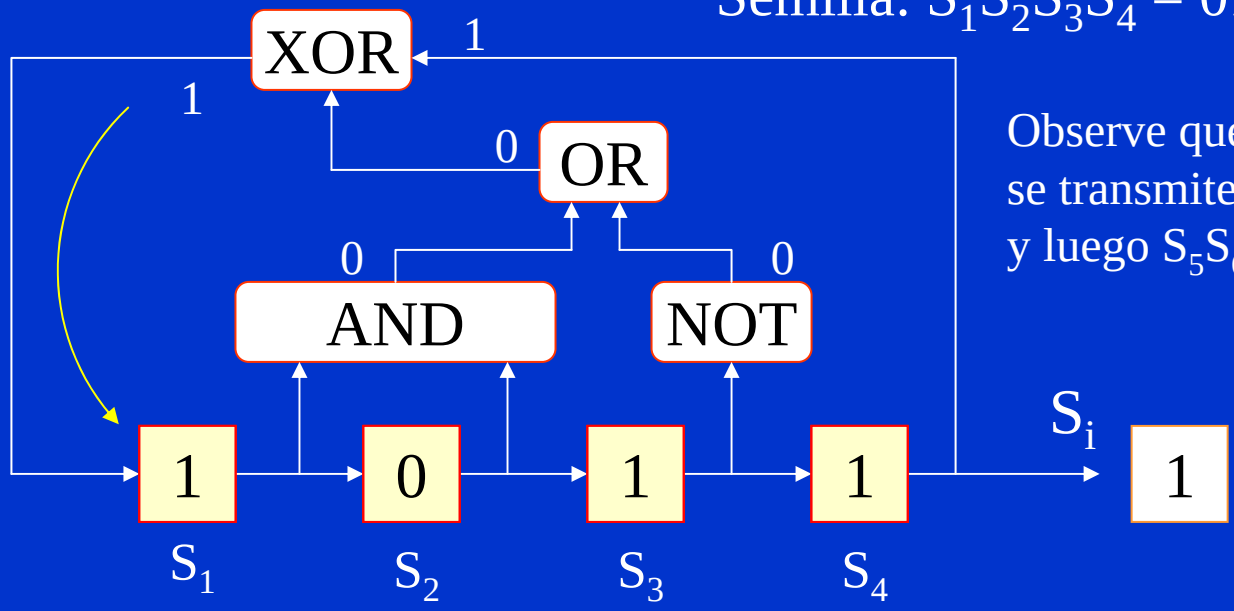
Este es el estado de las celdas y las operaciones previas antes de producirse el desplazamiento de un bit hacia a la derecha.

Sea la semilla: $S_1S_2S_3S_4 = 0111$

Operaciones

Generadores NLFSR (2/2)

Semilla: $S_1S_2S_3S_4 = 0111$



Observe que primero se transmite $S_4S_3S_2S_1$ y luego $S_5S_6S_7 \ldots S_{12}$.

S_i = 1110 1100 1010 0001; su período es máximo, $T_{máx} = 2^n = 2^4 = 16$. Se conoce como secuencia de De Bruijn. El contenido de las celdas pasará por todos los estados posibles: desde 0000 hasta 1111.

http://mathworld.wolfram.com/deBruijnSequence.html

Generadores lineales LFSR

$$a(t) = C_1a(t-1) \oplus C_2a(t-2) \oplus C_3a(t-3) \oplus \ldots \oplus C_na(t-n)$$

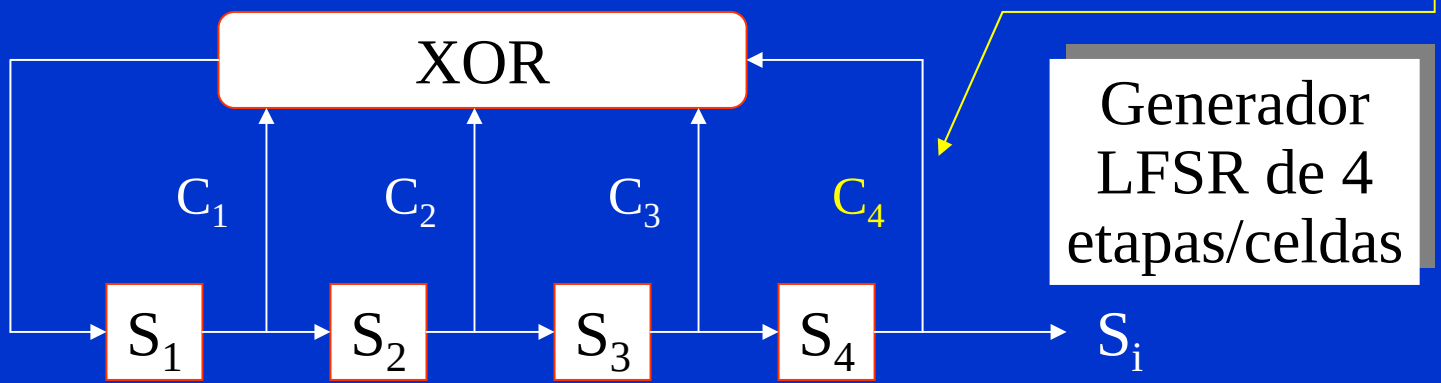
$C_i = \{1,0\} \Rightarrow$ conexión/no conexión celda $\quad C_n = 1$

Función única: XOR

$T_{máx} = 2^n - 1$

Polinomio asociado:

$$f(x) = C_nx^n + C_{n-1}x^{n-1} + ... + C_2x^2 + C_1x + 1$$



Generador LFSR de 4 etapas/celdas

Tipos de generadores lineales LFSR

Observación: como la única función de realimentación de un LFSR es un XOR, no estará permitida la cadena de todos ceros.

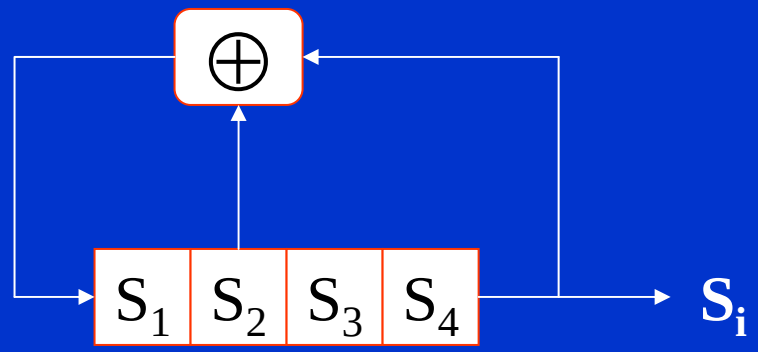
En función del polinomio asociado tendremos:

- LFSR con polinomios factorizables
  - No serán criptográficamente interesantes.
- LFSR con polinomios irreducibles
  - No serán criptográficamente interesantes.
- LFSR con polinomios primitivos
  - Según los postulados de Golomb, este tipo de polinomio que genera todos los estados lineales posibles del cuerpo de trabajo n, será el que nos entregue una secuencia cifrante de interés criptográfico con período $T = 2^n - 1$.

Generador LFSR con f(x) factorizable

Generador f(x) factorizable de cuatro celdas (n = 4)

Sea $f(x) = x^4 + x^2 + 1$



f(x) es factorizable porque:

Sea $f(x_1) = f(x_2) = (x^2+x+1)$

$f(x) = f(x_1) \bullet f(x_2)$

$f(x) = (x^2+x+1) \bullet (x^2+x+1)$

$f(x) = x^4+2x^3+3x^2+2x+1$

Tras la reducción módulo 2

Luego $f(x) = x^4 + x^2 + 1$

Problema

T dependerá de la semilla

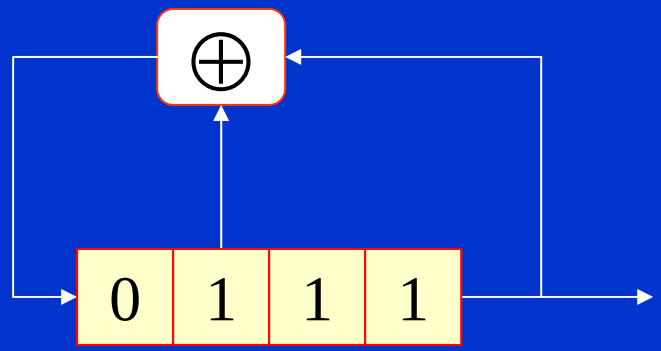
$T \leq 2^n - 1$

Y además, habrá períodos secundarios divisores de T.

Ejemplo LFSR con f(x) factorizable (1/2)

$f(x) = x^4 + x^2 + 1$

Sea la semilla:
$S_1S_2S_3S_4 = 0111$



Primer bit: resultado de la operación $S_1 = S_2 \oplus S_4$

Registro	Bit S_i	Registro	Bit S_i
0111	1	1110	0
0011	1	1111	1
1001	1	0111	1
1100	0		

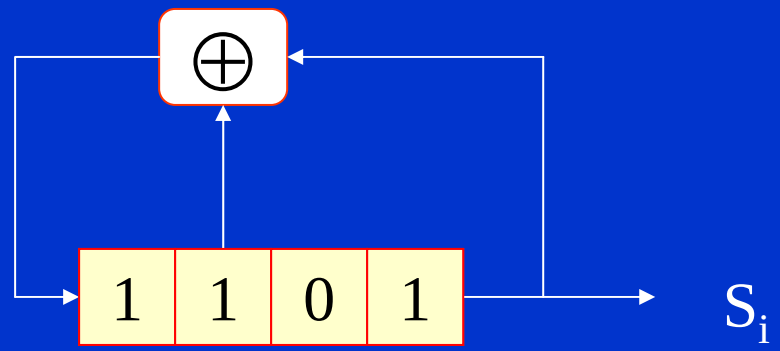
. . . semilla

$S_i = 111001$ $T = 6$

Ejemplo LFSR con f(x) factorizable (2/2)

$f(x) = x^4 + x^2 + 1$

Sea ahora la semilla:
$S_1S_2S_3S_4 = 1101$



Registro	Bit S_i
1101	1
0110	0
1011	1
1101	1

. . . semilla

Primer bit: resultado de la operación $S_1 = S_2 \oplus S_4$

$S_i = 101$
$T = 3$

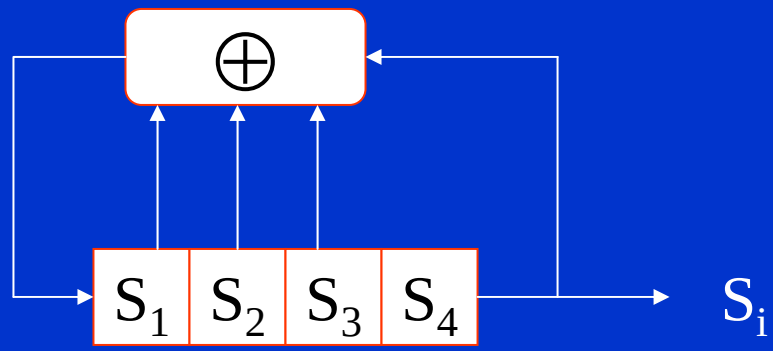
T es un período secundario y en en este caso es incluso menor que la semilla.

Generador LFSR con f(x) irreducible

Generador f(x) irreducible de cuatro celdas (n = 4)

Sea $f(x) = x^4 + x^3 + x^2 + x + 1$

Es imposible factorizar en módulo 2 la función f(x) mediante dos polinomios $f(x_1)$ y $f(x_2)$ de grado menor



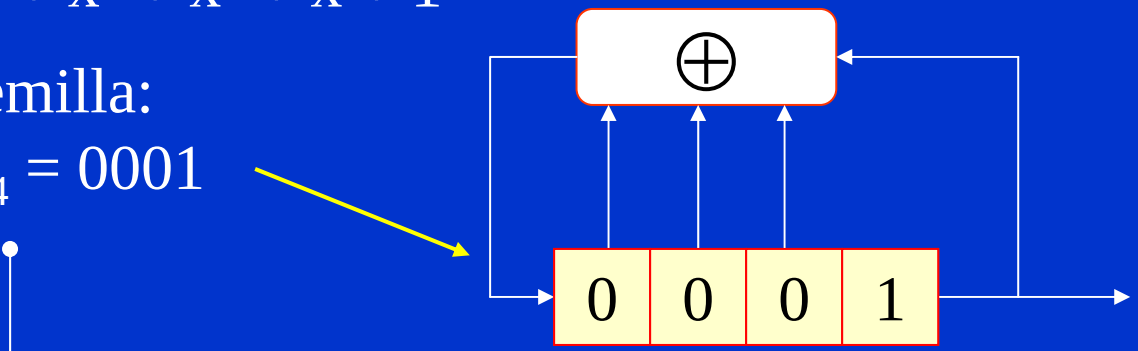
Problema

Ahora T ya no depende de la semilla pero será un factor de $T_{máx} = 2^n - 1$ y no obtendremos un período máximo.

Ejemplo de LFSR con f(x) irreducible

$f(x) = x^4 + x^3 + x^2 + x + 1$

Sea la semilla:
$S_1S_2S_3S_4 = 0001$



Registro	Bit S_i
0001	1
1000	0
1100	0
0110	0

Registro	Bit S_i
0011	1
0001	1

. . . semilla

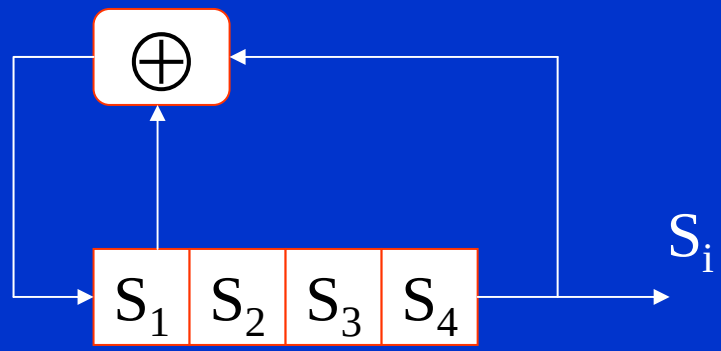
$S_i = 100011$ $T = 5$ siendo
$T_{máx} = 2^n - 1 = 2^4 - 1 = 15$

Generador LFSR con f(x) primitivo

Generador f(x) primitivo de cuatro celdas (n = 4)

Sea $f(x) = x^4 + x + 1$

f(x) no es factorizable como $f(x_1) \bullet f(x_2)$ en módulo 2. Será además un generador del grupo.



T ya no dependerá de la semilla y será el valor máximo $T_{máx} = 2^n - 1$. Se van a generar así las llamadas m-secuencias.

Habrá $\phi(2^n - 1)/n$ polinomios primitivos

Polinomios http://mathworld.wolfram.com/PrimitivePolynomial.html

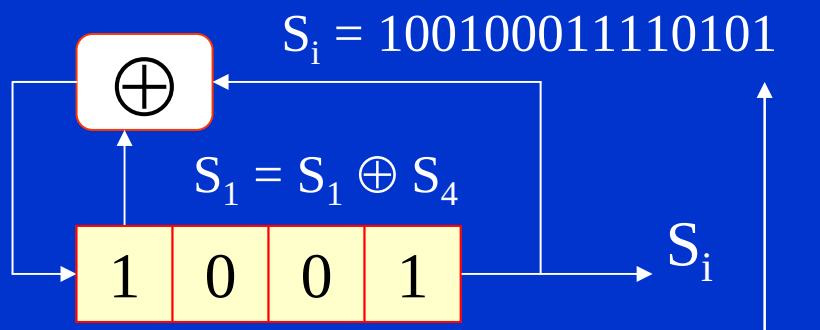
Generación polinomios http://www.theory.csc.uvic.ca/~cos/gen/poly.html

Ejemplo de LFSR con f(x) primitivo

Generador f(x) primitivo de cuatro celdas (n = 4)

$f(x) = x^4 + x + 1$

$S_1S_2S_3S_4 = 1001$



Registro	Bit S_i							
1001	1							
0100	0	1110	0		1010	0		
0010	0	1111	1		1101	1		
0001	1	0111	1		0110	0		
1000	0	1011	1		0011	1		
1100	0	0101	1		1001		T = 15	

$T = 2^n - 1$

$T = 2^4 - 1$

$T = 15$

Secuencias de un LFSR con f(x) primitivo

Características

- Tendrá una secuencia máxima de 2^n - 1 bits.
- Cumplirá con G1:
  - Hay 2n bits 1 y 2n-1 bits 0.
- Cumplirá con G2:
  - El vector binario de las celdas pasa por todos los estados excepto la cadena de ceros que está prohibida. Distribución típica de rachas de una m-secuencia.

m-secuencia

- Cumplirá con G3:
  - Los aciertos (A) serán iguales a 2^{n-1} - 1.

Rachas en S_i de un LFSR con f(x) primitivo

Rachas de Longitud	Rachas de Ceros	Rachas de Unos
1	2^{n-3}	2^{n-3}
2	2^{n-4}	2^{n-4}
...	...	...
p	2^{n-p-2}	2^{n-p-2}
...	...	...
n-2	1	1
n-1	1	0
n	0	1
TOTAL	2^{n-2}	2^{n-2}

Rachas de una m-secuencia

Sin embargo, no es un generador ideal para la cifra porque su Complejidad Lineal es muy baja. →

Debilidad de un LFSR con f(x) primitivo

Como este tipo de LFSR genera una secuencia de longitud máxima, ésta será previsible y se puede encontrar la secuencia completa S_i de 2^n - 1 bits ...

¡ con sólo conocer 2n bits !

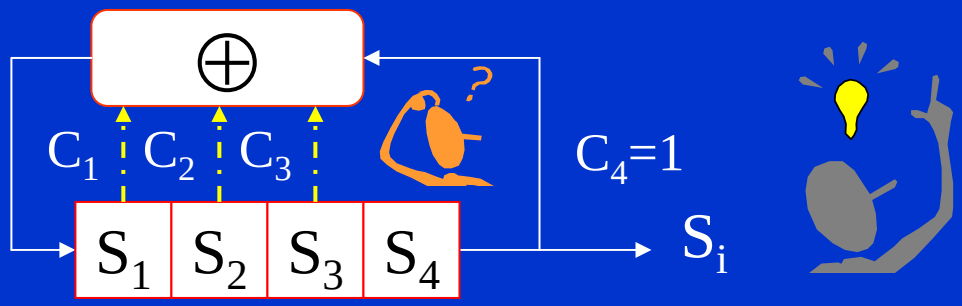
Por ejemplo, si conocemos sólo 2•10 = 20 bits en un sistema de 10 celdas con un período 2^{10}-1 = 1.023, seremos capaces de encontrar las conexiones de las celdas o valores de C_i y generar la secuencia completa S_i. Esta debilidad es la que usa el ataque conocido como algoritmo de Berlekamp-Massey.

http://planetmath.org/encyclopedia/BerlekampMasseyAlgorithm.html

Algoritmo http://ihome.ust.hk/~trippen/Cryptography/BM/frameset.html

Ejemplo de ataque de Berlekamp-Massey

Si conocemos $2*n = 8$ bits $S_1S_2S_3S_4S_5S_6S_7S_8$ de un LFSR de 4 celdas $C_1C_2C_3C_4$, tenemos el sistema de ecuaciones:



$$S_5 = C_1 \cdot S_1 \oplus C_2 \cdot S_2 \oplus C_3 \cdot S_3 \oplus C_4 \cdot S_4$$
$$S_6 = C_1 \cdot S_5 \oplus C_2 \cdot S_1 \oplus C_3 \cdot S_2 \oplus C_4 \cdot S_3$$
$$S_7 = C_1 \cdot S_6 \oplus C_2 \cdot S_5 \oplus C_3 \cdot S_1 \oplus C_4 \cdot S_2$$
$$S_8 = C_1 \cdot S_7 \oplus C_2 \cdot S_6 \oplus C_3 \cdot S_5 \oplus C_4 \cdot S_1$$

Si asignamos valores de esos $2*n = 8$ bits $S_1S_2S_3S_4S_5S_6S_7S_8$ seremos capaces de resolver este sistema

Primero se transmite $S_4S_3S_2S_1$ (semilla) y luego bits $S_5S_6S_7S_8$.

Solución al ataque de Berlekamp-Massey

$$S_5 = C_1 \cdot S_1 \oplus C_2 \cdot S_2 \oplus C_3 \cdot S_3 \oplus C_4 \cdot S_4$$
$$S_6 = C_1 \cdot S_5 \oplus C_2 \cdot S_1 \oplus C_3 \cdot S_2 \oplus C_4 \cdot S_3$$
$$S_7 = C_1 \cdot S_6 \oplus C_2 \cdot S_5 \oplus C_3 \cdot S_1 \oplus C_4 \cdot S_2$$
$$S_8 = C_1 \cdot S_7 \oplus C_2 \cdot S_6 \oplus C_3 \cdot S_5 \oplus C_4 \cdot S_1$$

Si $b_1b_2b_3b_4b_5b_6b_7b_8 = 11001000$ son correlativos y como hay cuatro celdas y primero se transmite la semilla, entonces:

$S_1 = 0 \quad S_5 = 1$
$S_2 = 0 \quad S_6 = 0$
$S_3 = 1 \quad S_7 = 0$
$S_4 = 1 \quad S_8 = 0$

$$1 = C_1 \cdot 0 \oplus C_2 \cdot 0 \oplus C_3 \cdot 1 \oplus C_4 \cdot 1$$
$$0 = C_1 \cdot 1 \oplus C_2 \cdot 0 \oplus C_3 \cdot 0 \oplus C_4 \cdot 1$$
$$0 = C_1 \cdot 0 \oplus C_2 \cdot 1 \oplus C_3 \cdot 0 \oplus C_4 \cdot 0$$
$$0 = C_1 \cdot 0 \oplus C_2 \cdot 0 \oplus C_3 \cdot 1 \oplus C_4 \cdot 0$$

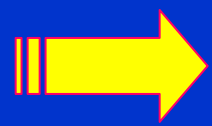
$C_1 = 1$ $C_2 = 0$

$C_3 = 0$ $C_4 = 1$

Conclusiones ataque Berlekamp-Massey

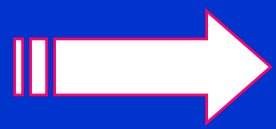
CONCLUSIONES:

- Como se conoce la configuración del generador LFSR, y S_i es una m-secuencia de período $2^n - 1$, entonces por el conjunto de n celdas pasarán todos los restos del campo de Galois de 2^n, excepto la cadena de n ceros que sabemos está prohibida en estos sistemas generadores lineales.
- Para el ejemplo anterior, esto quiere decir que cualquier grupo de 2n = 8 dígitos correlativos nos permite generar la secuencia máxima, en este caso de $2^n = 16$ bits.
- La solución es aumentar la complejidad lineal del generador por ejemplo conectando varios LFRs.

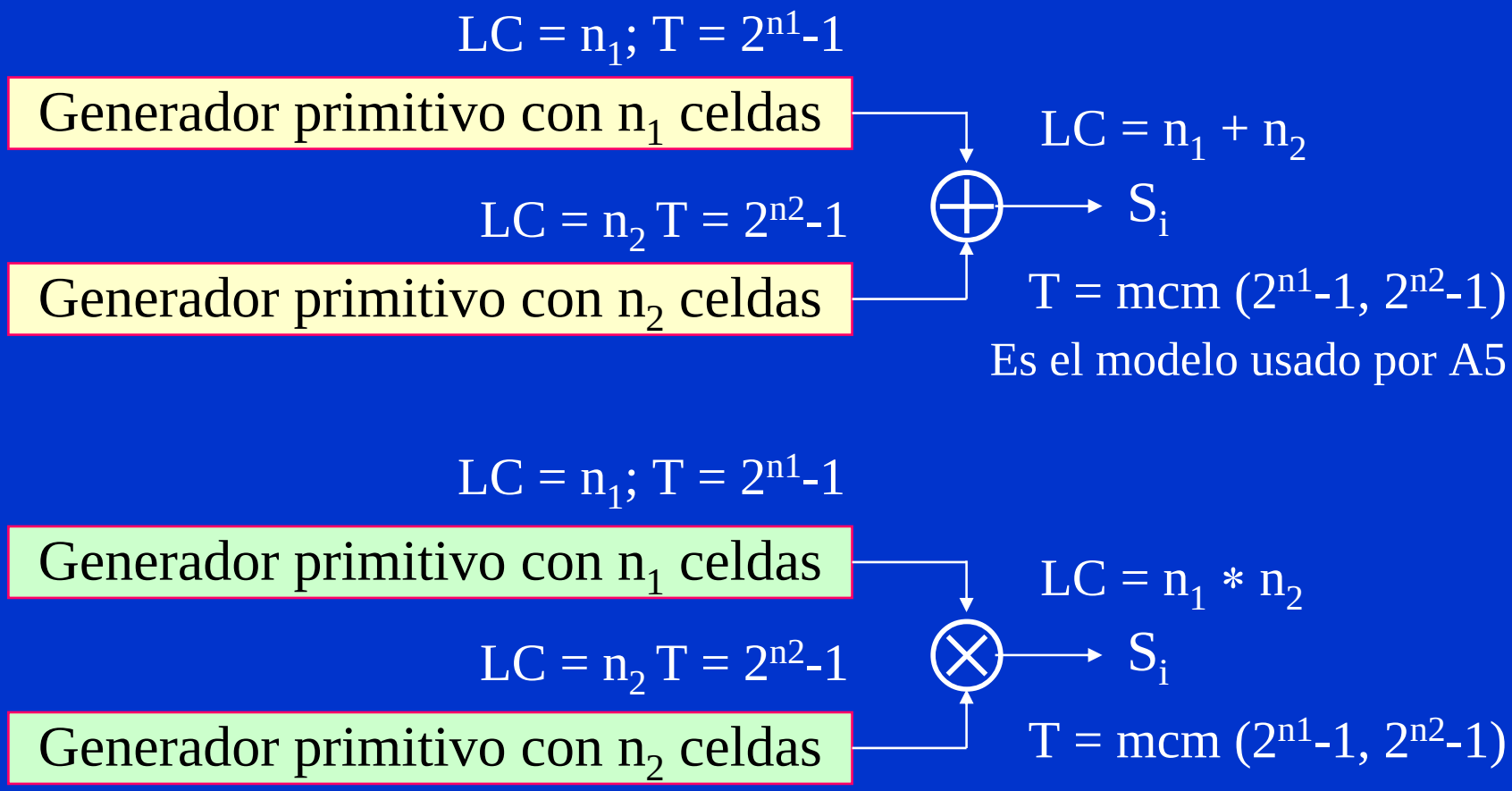


Complejidad lineal LC

- Un LFSR con polinomio primitivo de n celdas tendrá una complejidad lineal LC igual a n; es decir con 2n bits se puede generar la secuencia completa como hemos visto.
- Lo ideal es que si este LFSR entrega una secuencia S_i con un período igual a $2^n - 1$, su LC fuese cercana a este valor.
- Para aumentar esta LC podemos usar:
  - Operaciones no lineales de las secuencias del LFSR
    - Operaciones de suma
    - Operaciones de multiplicación
  - Filtrado no lineal de los estados del LFSR.

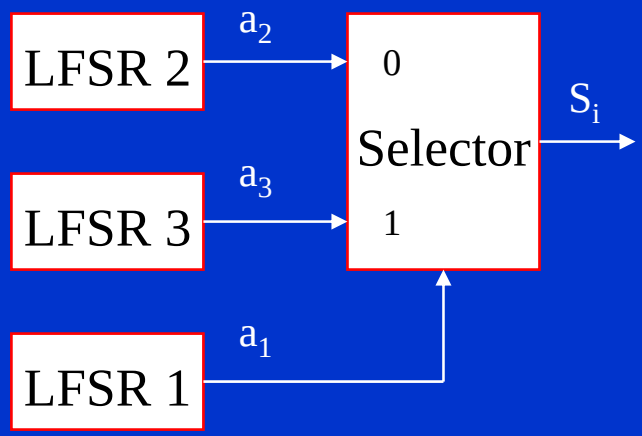


Operaciones no lineales con dos registros



Generadores LFSR con filtrado no lineal

Generador de Geffe



- Si a_1 es un 0 $\Rightarrow$ S_i es el bit de a_2
- Si a_1 es un 1 $\Rightarrow$ S_i es el bit de a_3

Luego: $S_i = a_2 \oplus a_1a_2 \oplus a_1a_3$

$LC = (n_1 + 1)n_2 \oplus n_1n_3$

$T = mcm\ (2^{n1}\text{-}1,\ 2^{n2}\text{-}1,\ 2^{n3}\text{-}1)$

Se mejora la LC e incluso se aumenta si ponemos más LFSRs pero este generador es débil ante ataques por correlación de bits. Existen muchos esquemas en esta línea: Beth-Piper, Gollmann, Massey-Rueppel, etc., que no serán tratados en este capítulo.

Encontrará ataques algebraicos a cifradores de flujo en la siguiente Web.

http://www.cryptosystem.net/stream

Algoritmos de cifrado en flujo

Sistemas más conocidos:

- A5: http://www.argo.es/~jcea/artic/hispasec33.htm 
  - Algoritmo no publicado propuesto en 1994. Versiones A5/1 fuerte (Europa) y A5/2 débil (exportación).
- RC4: http://www.wisdom.weizmann.ac.il/~itsik/RC4/rc4.html 
  - Algoritmo de RSA Corp. (*Rivest Cipher* #4) desarrollado en el año 1987, usado en Lotus Notes. Posteriormente se usa en el navegador de Netscape desde 1999 y luego en otros navegadores más actuales. No es público.
- SEAL: http://www.gemplus.com/smart/rd/publications/pdf/HG97chis.pdf 
  - Algoritmo propuesto por IBM en 1994.

El algoritmo de cifra A5

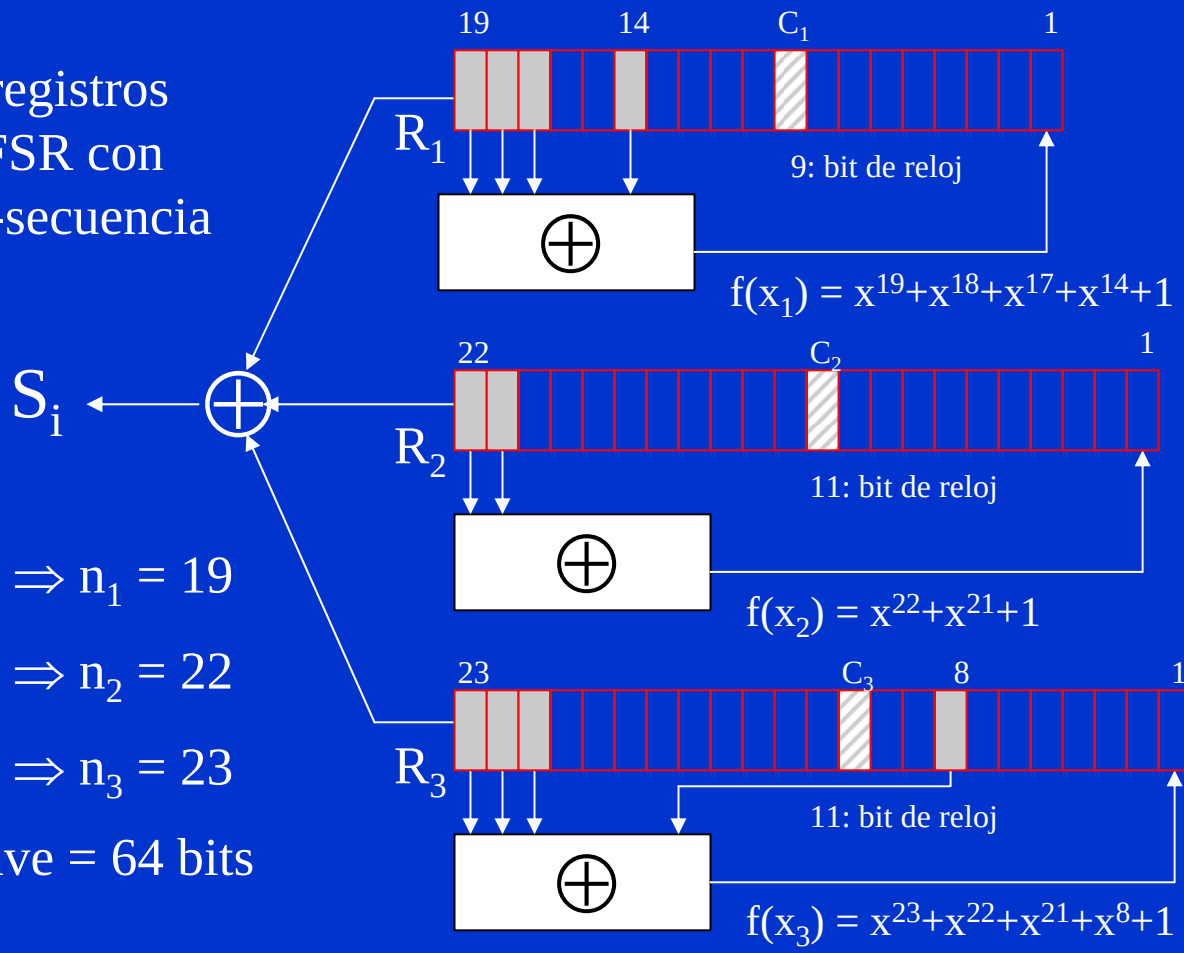
- El uso habitual de este algoritmo lo encontramos en el cifrado del enlace entre el abonado y la central de un teléfono móvil (celular) tipo GSM.
- Cada trama de conversación entre A y B tiene 228 bits, de los cuales 114 son en sentido A $\rightarrow$ B y otros 114 en sentido B $\rightarrow$ A. El generador entregará los 228 bits pseudoaleatorios para la cifra de cada trama.
- Con cerca de 130 millones de usuarios en Europa y otros 100 millones de usuarios en el resto del mundo en 1999, el sistema A5/1 sucumbió en diciembre de ese año a un ataque realizado por Alex Biryukov, Adi Shamir y David Wagner.

http://cryptome.org/a51-bsw.htm

http://www.criptored.upm.es/guiateoria/gt_m116a.htm

Esquema del algoritmo de cifra A5/1

3 registros LFSR con m-secuencia



$R_1 \Rightarrow n_1 = 19$

$R_2 \Rightarrow n_2 = 22$

$R_3 \Rightarrow n_3 = 23$

Clave = 64 bits

Una función mayoría entre C_1, C_2 y C_3 hace que sólo los registros en los que coincide el bit con ese valor produzcan desplazamiento. En cada paso habrá dos o tres registros en movimiento.

Función mayoría y ejemplo de secuencia

$$F(C_1,C_2,C_3) = C_1C_2 \oplus C_1C_3 \oplus C_2C_3$$

Si el bit de la celda del registro coincide con el resultado de F, dicho registro estará en movimiento y se desplazará, en caso contrario no desplazará.

Esta función mayoría entre las celdas C_1, C_2 y C_3, permite que al menos dos de los tres registros se desplacen en cada paso.

C_1	C_2	C_3	
0	0	0	Desplazan todos
0	0	1	No desplaza R_3
0	1	0	No desplaza R_2
0	1	1	No desplaza R_1

C_1	C_2	C_3	
1	0	0	No desplaza R_1
1	0	1	No desplaza R_2
1	1	0	No desplaza R_3
1	1	1	Desplazan todos

Consideraciones sobre el período de A5/1

El período T vendrá dado por el mínimo común múltiplo de los tres períodos individuales:

$$T = mcm\ (2^{n1} - 1, 2^{n2} - 1, 2^{n3} - 1)$$

Como n_1, n_2 y n_3 son primos entre sí, también lo serán los valores $(2^{n1} -1)$, $(2^{n2} - 1)$ y $(2^{n3} - 1)$. Luego el período T será el producto:

$$T = T_1 * T_2 * T_3$$

Entonces $T = (2^{19}-1)(2^{22}-1)(2^{23}-1) = 524.287 * 4.194.303 * 8.388.607$

$$T = 18.446.702.292.280.803.327 \ < \ 2^{64}$$

Este valor demasiado bajo ☹ sucumbe ante ataques distribuidos tal como veremos cuando se estudien debilidades del algoritmo DES.

Registros y función mayoría en A5/2

- Usa los mismos tres registros de desplazamiento con polinomio primitivo que A5/1:
  - $f(x_1) = x^{19} + x^{18} + x^{17} + x^{14} + 1$
  - $f(x_2) = x^{22} + x^{21} + 1$
  - $f(x_3) = x^{23} + x^{22} + x^{21} + x^{8} + 1$
- Además, usa un cuarto registro R_4 con un polinomio primitivo:
  - $f(x_4) = x^{17} + x^{12} + 1$
- Usa cuatro copias de una función mayoría F para cada uno de los cuatro registros que se define como:
  - $F(x_1,x_2,x_3) = x_1x_2 + x_1x_3 + x_2x_3$

Otras operaciones de A5/2

  - En R_1 las entradas a la función F_1 son las celdas 13, 15 y 16.
  - En R_2 las entradas a la función F_2 son las celdas 10, 14 y 17.
  - En R_3 las entradas a la función F_3 son las celdas 14, 17 y 19.
  - En R_4 las entradas a la función F_4 son las celdas 4, 8 y 11. La salida de esta copia determina qué registros de R_1,R_2,R_3 se desplazarán en el ciclo.
- Complementación de celdas y sumas en salida de F:
  - En R_1 se complementa la celda 15 y se suma la celda 19 a F.
  - En R_2 se complementa la celda 17 y se suma la celda 22 a F.
  - En R_3 se complementa la celda 14 y se suma la celda 23 a F.

Fin del capítulo

Cuestiones y ejercicios (1 de 3)

1. ¿Por qué en los sistemas de cifra en flujo se usa una función XOR tanto en emisor como en receptor? ¿Son las claves aquí inversas?
2. Si tenemos una clave de 16 bits, ¿cuál es el período máximo que podremos lograr en una secuencia cifrante lineal? ¿Por qué?
3. ¿Qué rachas encuentra en la secuencia 1101001111010100?
4. ¿Por qué es lógico esperar más rachas cortas que rachas largas?
5. Si en una secuencia cifrante se observa una correlación fuera de fase no constante, ¿qué significa? ¿Podríamos hacer un ataque similar al que permite romper el sistema de cifra polialfabético de Vigenère?
6. A nivel de probabilidades de ocurrencia de bits, ¿qué significan los postulados de Golomb G1 y G2?
7. ¿Qué significa que una secuencia cifrante pase por todos los estados o restos posibles? ¿Cuál sería en este caso su período?

Cuestiones y ejercicios (2 de 3)

8. ¿Qué secuencias se obtiene con un generador de congruencia lineal en el que a = 7, b = 4 y n = 8? ¿Y si ahora hacemos a = 3?
9. ¿Qué tipo de ataque podríamos intentar para romper una secuencia cifrante obtenida con un generador de congruencia lineal?
10. ¿Por qué en un registro de desplazamiento siempre se realimenta el bit de la última celda, es decir bit que sale a línea?
11. En el generador NLFSR de los apuntes si se cambia la función AND por una OR, ¿qué sucede con la secuencia? Saque conclusiones.
12. Decimos que los generadores LFSR tienen asociado un polinomio $f(x) = a_n x^n + a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + ... + a_2x^2 + a_1x + 1$, donde a_i toma los valores de 0 ó 1. ¿Por qué f(x) termina en 1?
13. ¿Qué polinomio elegiría para un generador con LFSR, un polinomio factorizable, uno irreducible o uno primitivo? ¿Por qué?

Cuestiones y ejercicios (3 de 3)

14. ¿Por qué no está permitida la cadena de n ceros en un generador LFSR de n etapas y en cambio sí en los NLFSR?
15. En el generador LFSR con f(x) primitivo de 4 etapas, la secuencia pasa por todos los restos excepto 0000. Si usamos hora una semilla distinta, ¿debemos hacer otra vez todos los cálculos o no? ¿Por qué?
16. Justifique la distribución de las rachas en una m-secuencia. ¿Por qué tiene ese comportamiento extraño al final de las rachas largas?
17. ¿Qué debilidad de los sistemas LFSR con polinomios primitivos usa el algoritmo de Berlekamp-Massey para realizar el ataque?
18. En un ataque de Berlekamp-Massey, ¿importa en qué posición de la secuencia se encuentran los 2n bits? ¿Por qué?
19. ¿Por qué cree que el algoritmo A5/1 es débil? ¿Cuál fue el peor error cometido por sus creadores?

Software de laboratorio de flujo: próximamente en página web de la asignatura.