Proyecto CLCript
Cuadernos de Laboratorio de Criptografía. Entrega nº 16. Ultima actualización 11/10/19
Autor: Dr. Jorge Ramió Aguirre (@criptored)
Práctica sobre cifra en flujo con registros de desplazamiento realimentados linealmente LFSR con FlujoLab

- Software FlujoLab: https://www.criptored.es/software/sw_m001m.htm
- Software SAMCript: https://www.criptored.es/software/sw_m001t.htm
- Códigos y tablas de uso frecuente en criptografía CLCript: https://www.criptored.es/descarga/Codigos_y_tablas_de_uso_frecuente_en_criptografia.pdf

Objetivos:
1. Saber cómo se cifra y cómo se descifra en flujo bit a bit, textos y archivos.
2. Comprobar el ataque de Berlekamp-Massey a las m-secuencias.
3. Comprender cómo se generan secuencias cifrantes con mayor complejidad lineal, usando más de un registro.

I. Cifrado y descifrado en flujo de textos con registros de desplazamiento simples

Ejercicio 1

1.1. Con FlujoLab (Generadores-LFSR-General) genera una secuencia cifrante de 4.095 bits con el polinomio primitivo $x^{12} + x^6 + x^4 + x + 1$ representado como (12,6,4,1,0) y usando la semilla $S_1S_2S_3S_4S_5S_6S_7S_8S_9S_{10}S_{11}S_{12}$ = 111000000111.

1.2. Con esa secuencia, cifra el texto txt que se indica más abajo; no añadas espacios en blanco o un punto al final, ni "retorno de carro" o "Enter".
Este mensaje secreto se destruirá dentro de 1, 2, 3,... segundos

1.3. Comprueba con SAMCript que en binario la cifra XOR de los cuatro primeros caracteres del texto en claro "Este" cuyos valores ASCII en binario son:
E = 01000101
s = 01110011
t = 01110100
e = 01100101
Con los primeros 32 bits de la secuencia de clave:
S_i = 11100000011100001000100100101011
Debe ser C = 10100101000000111111110101001110

1.4. Con las tablas de CLCript comprueba que los primeros 30 bits del criptograma, agrupados de 6 en 6 (101001 010000 001111 111101 010011), se corresponden con los cinco caracteres pQP9T en Base 64.

1.5. Justifica por qué en el criptograma en Base 64 aparecen dos signos igual al final (==).
Ayuda: el texto en claro tiene 64 caracteres (512 bits).

1.6. Descifra el criptograma C que se indica más abajo en Base 64, sabiendo que el polinomio primitivo generador de la secuencia cifrante era (11,2,0) con la semilla 11101011110.
C = MpLB5933Xm5FcX1F2/tJScAx3EmYUbOiJNo=

1.7. ¿Por qué en este caso el criptograma tiene sólo un signo igual al final?

1.8. Si se representa el texto "La vida puede ser maravillosa." (no se incluyen las comillas) en Base 64 se obtiene: TGEgdmlkYSBwdWVkZSBzZXIgbWFyYXZpbGxvc2Eu. ¿Sería esto último una cifra o no? ¿Por qué?

1.9. ¿Qué sucede si cifras este texto en claro con la clave 0000000000?

1.10. ¿Qué vulnerabilidad tendría una cifra en flujo en la que la secuencia cifrante tuviese muchas y largas cadenas de ceros?

1.11. ¿Por qué en este último caso el criptograma no tiene ningún signo igual al final?

Comprueba tu trabajo:

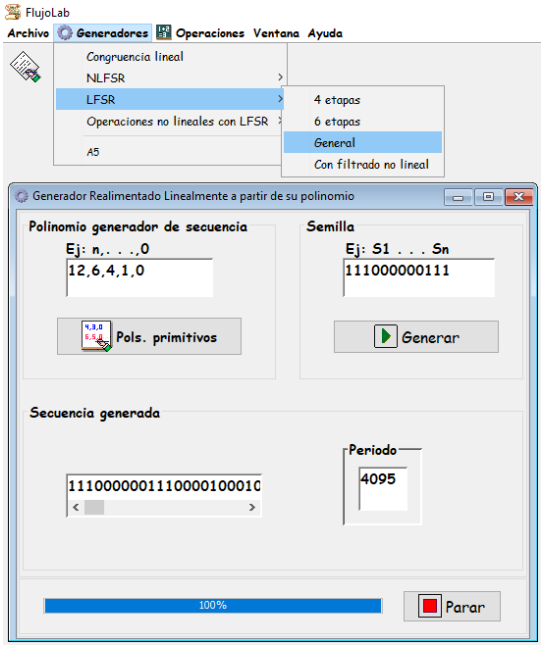

Figura 1. Generando la secuencia solicitada.

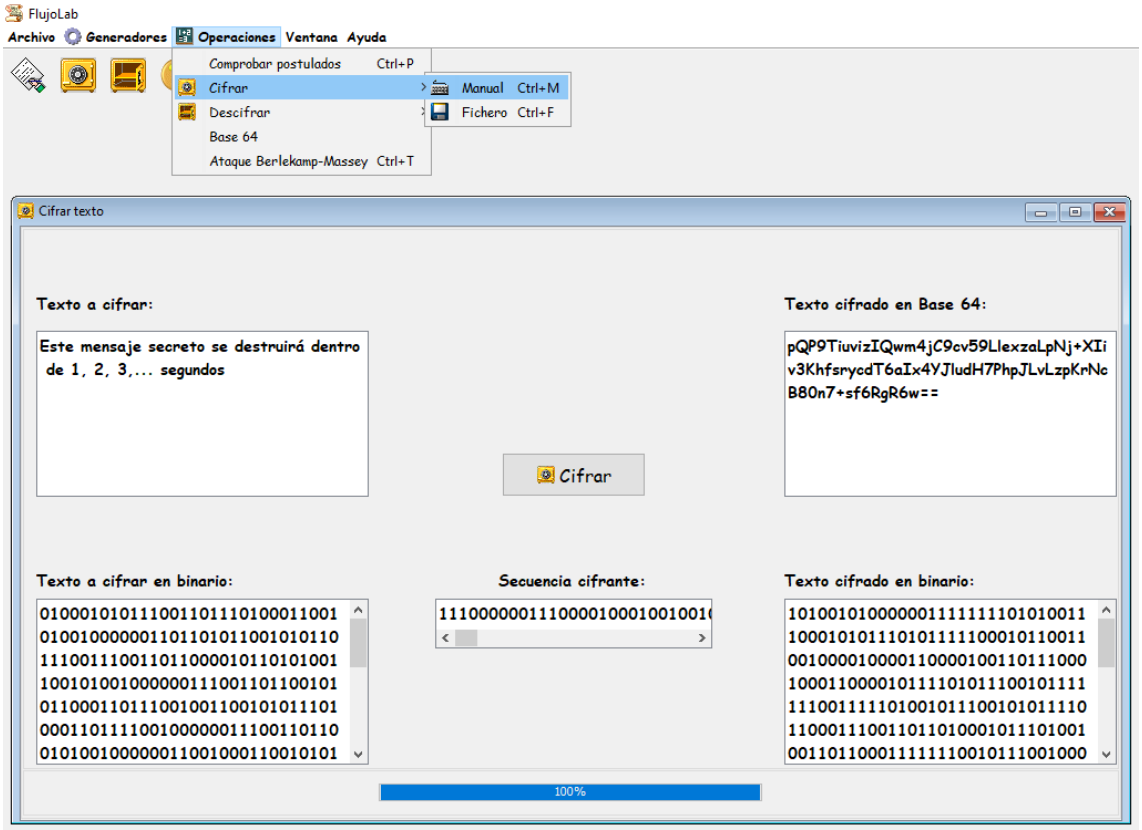

Figura 2. Cifra del texto en claro txt que se indica y salida en Base 64.

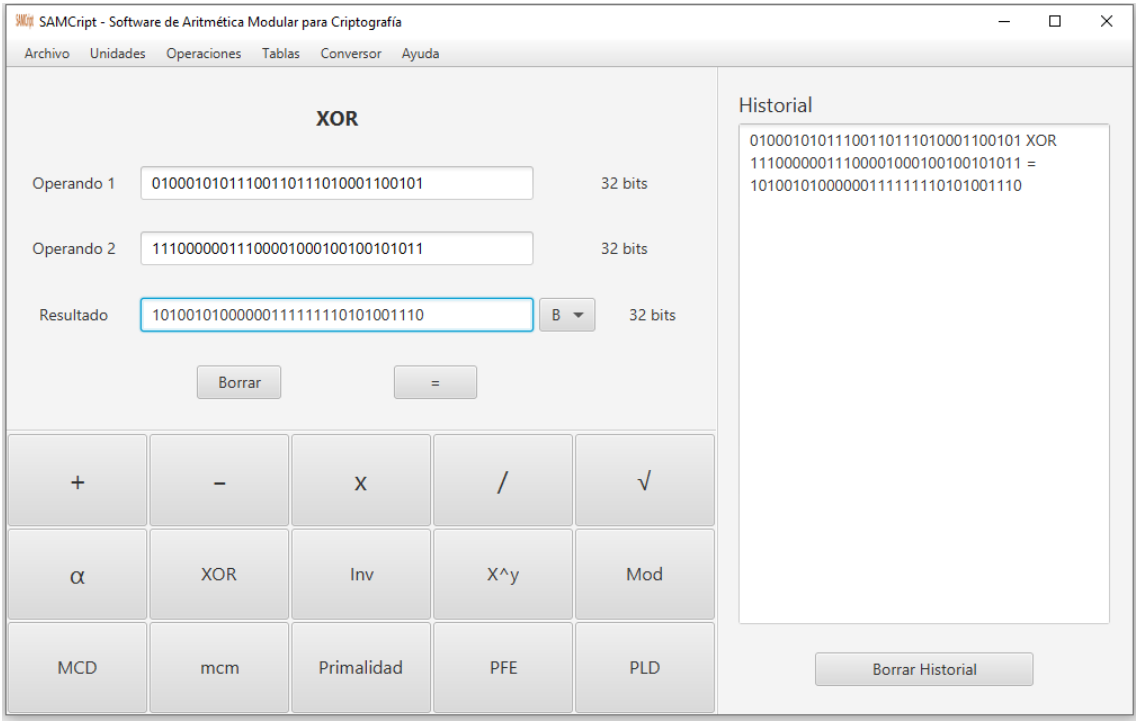

Figura 3. Comprobación de la cifra XOR de los primeros 32 bits del texto en claro con los primeros 32 bits de la secuencia cifrante.

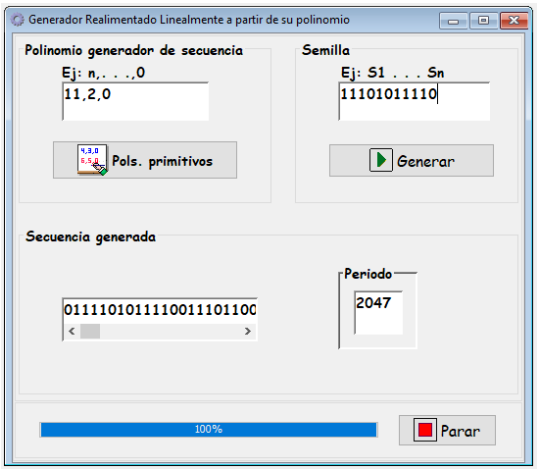

Figura 4. Generación de la secuencia cifrante con polinomio (11,2,0) y semilla 11101011110.

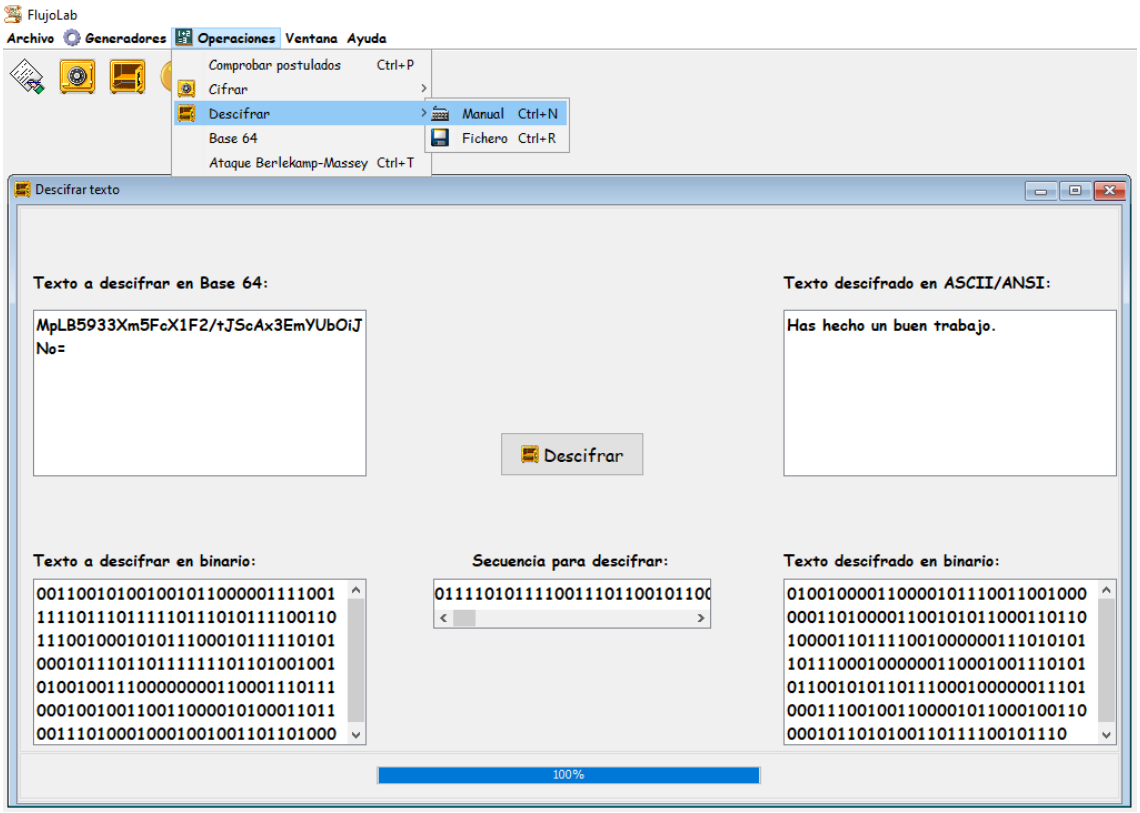

Figura 5. Descifrado del criptograma.

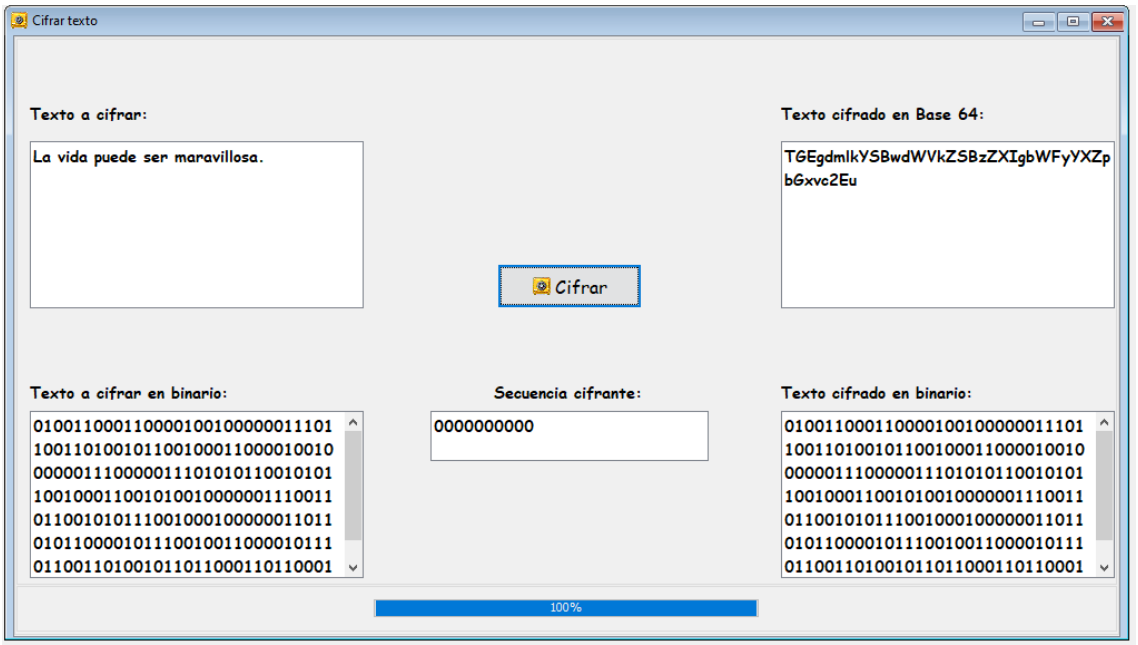

Figura 6. Cifrando en flujo con una cadena de 0s equivale a codificar el texto en Base 64.

II. Cifrado y descifrado en flujo de archivos con registros de desplazamiento simples

Ejercicio 2

2.1. Elige el archivo ejecutable Cifrador.jar de FlujoLab con el que estás trabajando y cífralo con la secuencia cifrante del registro primitivo (15,1,0) y una semilla 111111101111111. Introducido el archivo y la secuencia de clave, pulsa en Cifrar. Elige como nombre del archivo de salida Cifrador.jar y pulsa Guardar para que comience la operación de cifra.

2.2. Comprueba que (obviamente) el archivo Cifrador.cif no es un ejecutable.

2.3. Si el archivo Cifrador.jar tiene 738 KB, ¿cuántas veces se habrá repetido la secuencia cifrante durante la cifra?

2.4. Descifra ahora el archivo Cifrador.cif con la misma secuencia de clave y elige como nombre del archivo descifrado de salida Cifrador.dcf.

2.5. Cambia el nombre del archivo Cifrador.dcf por CifradorBis.jar y comprueba que has recuperado el archivo ejecutable original. Nota: el sistema operativo te avisará antes que cambiar la extensión de un archivo puede dejarlo inutilizable.

Comprueba tu trabajo:

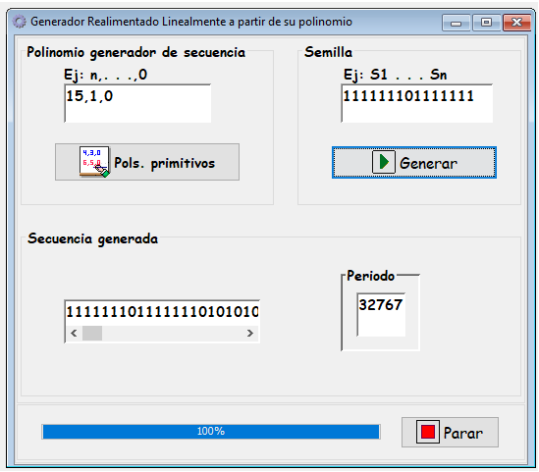

Figura 7. Secuencia cifrante para el cifrado del archivo Word.

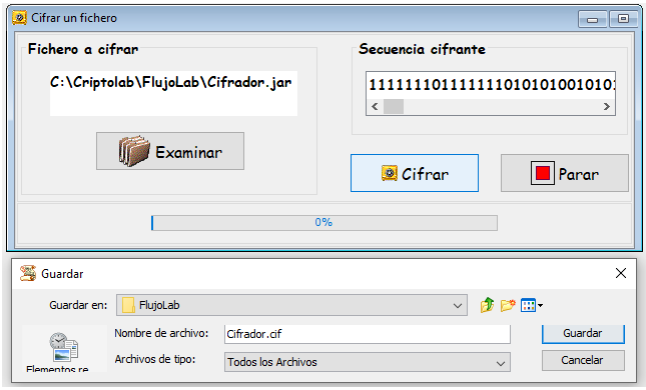

Figura 8. Ventana de preparación para cifrar el archivo Cifrador.jar como Cifrador.cif.

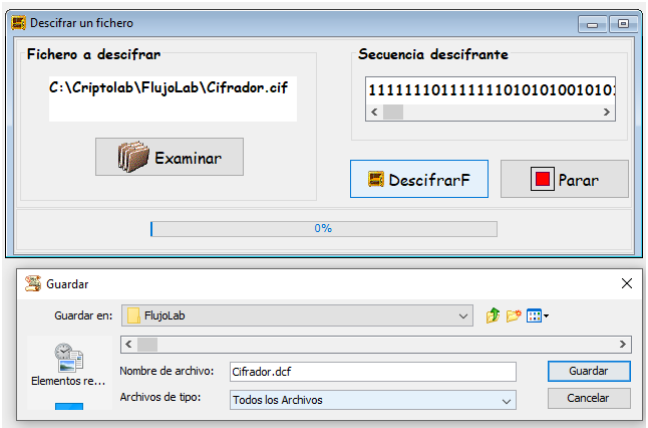

Figura 9. Ventana de preparación para descifrar el archivo Cifrador.cif como Cifrador.dcf.

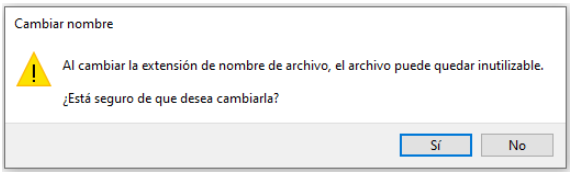

Figura 10. Advertencia del S.O. por el cambio de extensión en un archivo.

III. Ataque de Berlekamp Massey a m-secuencias

Ejercicio 3

3.1. Genera una secuencia con el polinomio (10,9,8,7,6,5,4,3,0) y la semilla 1111001111.

3.2. De la secuencia de 1.024 bits, elige 20 bits correlativos que están desde la posición 51 en adelante. Serán los 2xn = 2x10 = 20 bits para necesarios para el ataque.

3.3. Comprueba que esos 20 bits son: 00101100001111000000. Guárdalos en un archivo para poder usarlos posteriormente

3.4. Cierra FlujoLab y vuelve a arrancar el programa. Copia esos 20 bits como entrada del ataque (Operaciones – Ataque Berlekamp-Massey) e inicia el ataque. En pocos minutos, FlujoLab encontrará que el polinomio generador era (10,9,8,7,6,5,4,3,0).

3.5. Guarda el trabajo de ataque en un archivo de nombre AtaqueBM-CLCript16.html. Abre el archivo HTML y observa las ecuaciones que ha utilizado FlujoLab para encontrar el polinomio.

3.6. Observa que los bits de la semilla $S_1S_2S_3S_4S_5S_6S_7S_8S_9S_{10}$ son los 10 primeros capturados pero leídos al revés, es decir: 0000110100.

Comprueba tu trabajo:

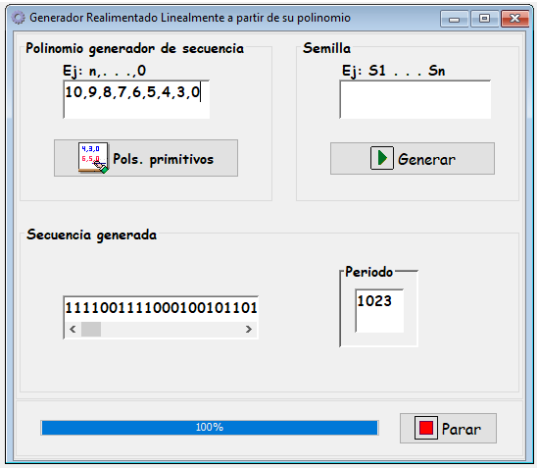

Figura 11. Secuencia cifrante con polinomio (10,9,8,7,6,5,4,3,0) y la semilla 1111001111.

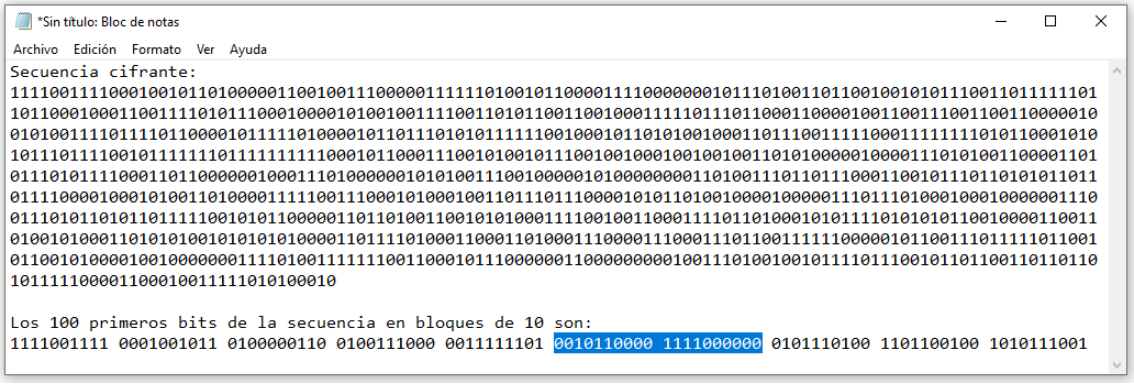

Figura 12. Marcado en azul los 20 bits de la secuencia a partir de la posición 51.

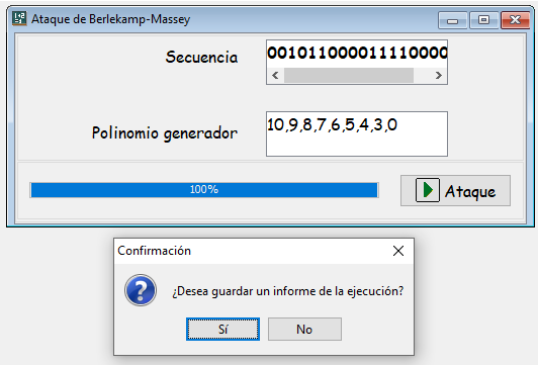

Figura 13. Resultado del ataque de Berlekamp Massey a un polinomio de grado 10, conociendo 20 bits correlativos de la secuencia.

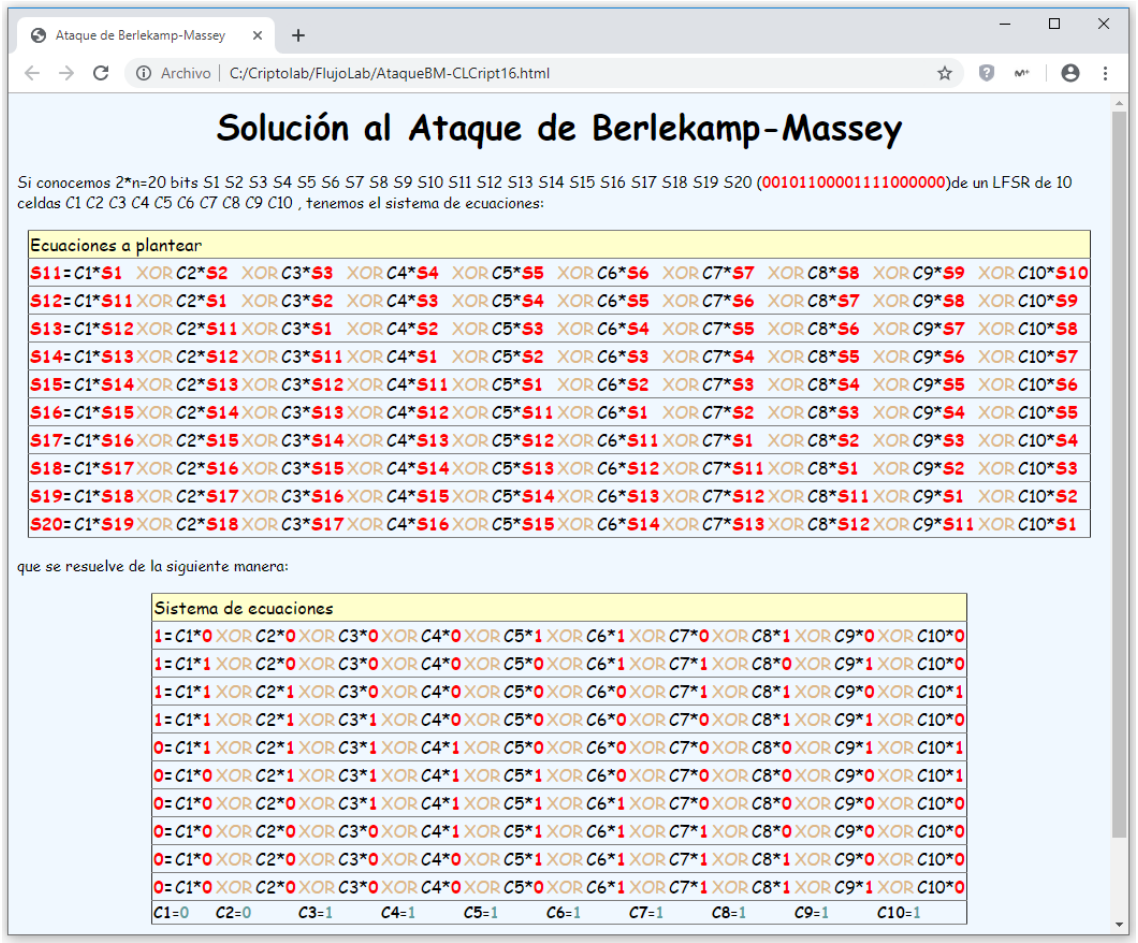

Figura 14. Informe de FlujoLab al ataque de Berlekamp-Massey realizado.

IV. Secuencias cifrantes con mayor complejidad Lineal LC

Ejercicio 4

4.1. Desde el Menú Generadores – Operaciones No Lineales con LFSR, vamos a generar dos secuencias cifrantes con dos LFSR primitivos usando la suma.

- 4.1.1. En el primer caso el primer polinomio será (8,6,5,1,0) con semilla 11100111 y el segundo polinomio será (9,5,0) con semilla 000010000.
- 4.1.2. Genera la secuencia indicada.
- 4.1.3. Usando el software SAMCript, comprueba que el periodo de la secuencia cifrante generada es igual al mcm (T_{LFSR1}, T_{LFSR2}) = mcm (2^8-1, 2^9-1) = mcm (255, 511) = 130.305.

4.1.4. Comprueba que en esta secuencia sólo se cumple el postulado 1 de Golomb. No hace falta que lo guardes en un archivo.

4.1.5. En el segundo caso el primer polinomio será (8,6,5,1,0) con semilla 11100111 y el segundo polinomio será (10,7,0) con semilla 0000110000.

4.1.6. Genera la secuencia indicada.

4.1.7. Usando el software SAMCript, comprueba que el periodo de la secuencia cifrante generada es igual al mcm (T_{LFSR1}, T_{LFSR2}) = mcm (2^8-1, 2^{10}-1) = mcm (255, 1.023) = 89.955.

4.1.8. Comprueba que ahora no se cumple ningún postulado de Golomb. No hace falta que lo guardes en un archivo.

4.1.9. Observa que, aunque en el segundo caso hemos usado un polinomio mayor [(10,7,0) en vez de (9,5,0)] el periodo de la secuencia cifrante ha sido bastante menor (89.955 en vez de 130.305). ¿Por qué sucede esto?

4.2. Usa ahora la opción multiplicación con los polinomios (5,2,0) con semilla 00100 y (6,1,0) con semilla 101010. Genera la secuencia y comprueba los postulados de Golomb. Guarda el archivo como PostuladosGolombMultiplicaciónCLCript16.html.

4.2.1. Comprueba que no se cumple ningún postulado y que en la secuencia cifrante hay 512 unos y 1.441 ceros, casi tres veces más ceros que unos.

4.2.2. ¿Por qué ocurre esto? ¿Por qué no es aconsejable la opción multiplicación y sí es adecuada la suma xor?

Comprueba tu trabajo:

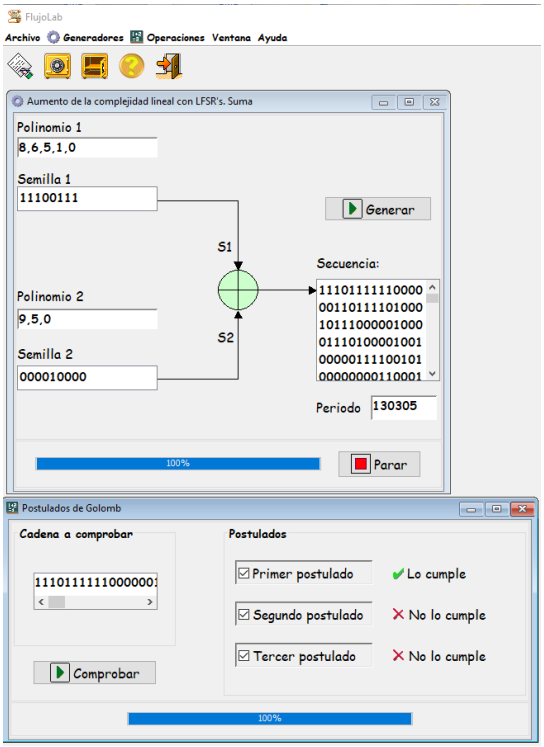

Figura 15. Secuencia suma XOR con dos polinomios primitivos (8,6,5,1,0) y (9,5,0).

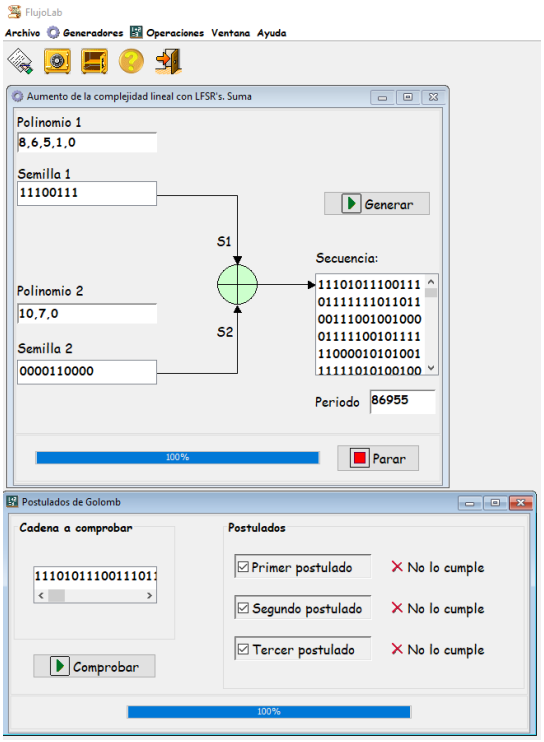

Figura 16. Secuencia suma XOR con dos polinomios primitivos (8,6,5,1,0) y (10,7,0).

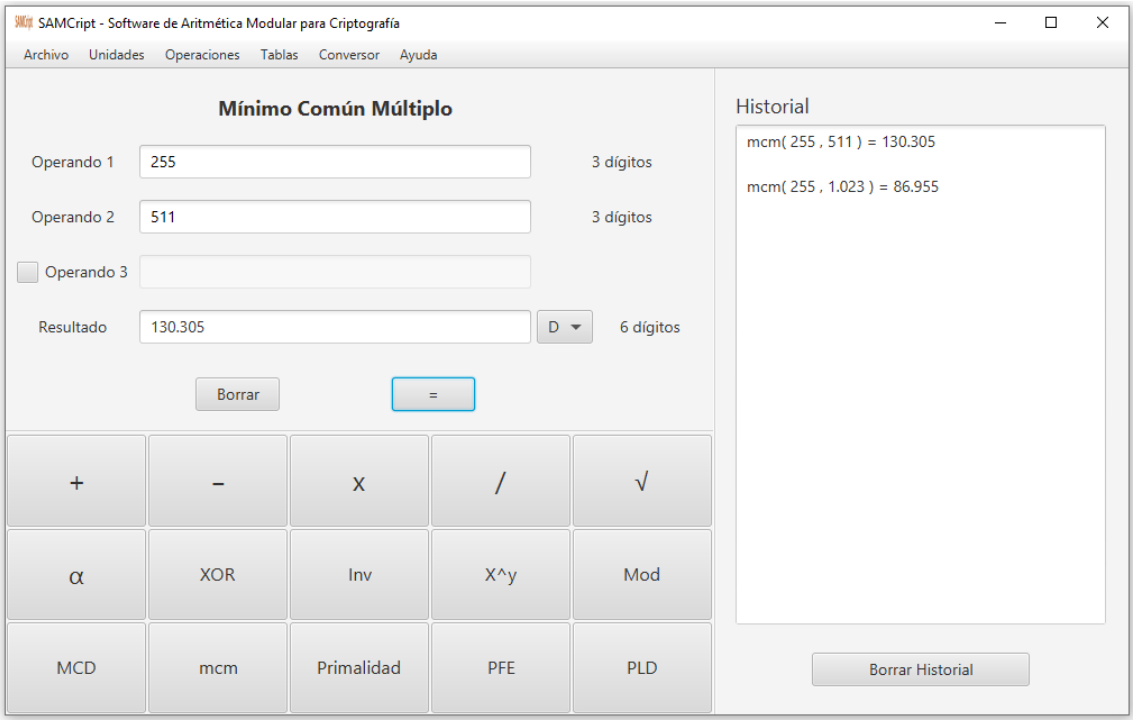

Figura 17. Periodos mcm (T_{LFS1}, T_{LSF2}).

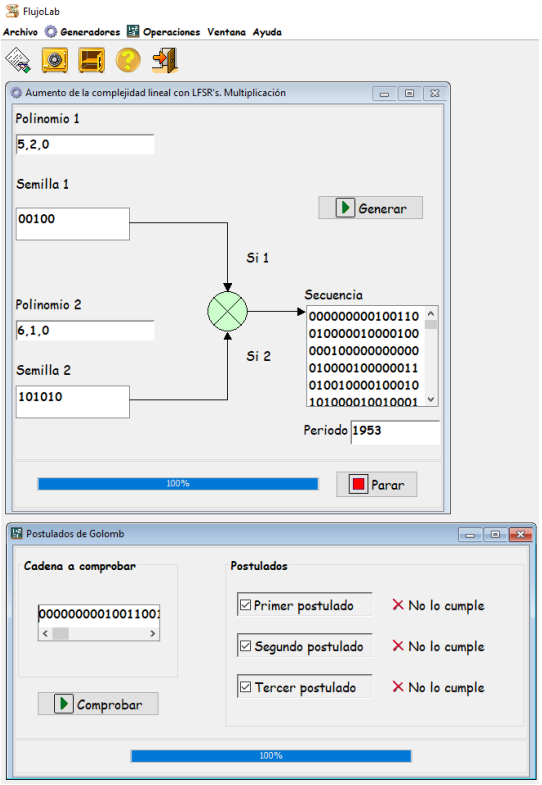

Figura 18. Secuencia multiplicación con dos polinomios primitivos (5,2,0) y (6,1,0).

Madrid, 11 de octubre de 2019
Dr. Jorge Ramió Aguirre