Protocolo de Comunicación Segura

Variante 39

Alberto Morcillo Villa 100383509,

Alfonso Serrano Corbelle 100363775,

Manuel Eduardo Medina Matute 100376437

Álvaro Morata Hontanaya 100405846

20-05-2020

ÍNDICE

1.Datos de identificación

El grupo de trabajo es el número 6, y su variante asignada la número 39.

Los integrantes del grupo son:

- Alberto Morcillo Villa 100383509
- Alfonso Serrano Corbelle 100363775
- Manuel Eduardo Medina Matute 100376437
- Álvaro Morata Hontanaya 100405846

Var39	Sign-then-Encrypt	Negociación_DH	Flujo_LFSR	Firma_mensaje_EG_con_Resumen	Firma_Cert_RSA_con_Resumen

2.Especificación del sistema

2.1. Representación de datos

Vamos a representar los datos en decimal.

2.2. Algoritmos y esquemas

Esquema Principal del Trabajo

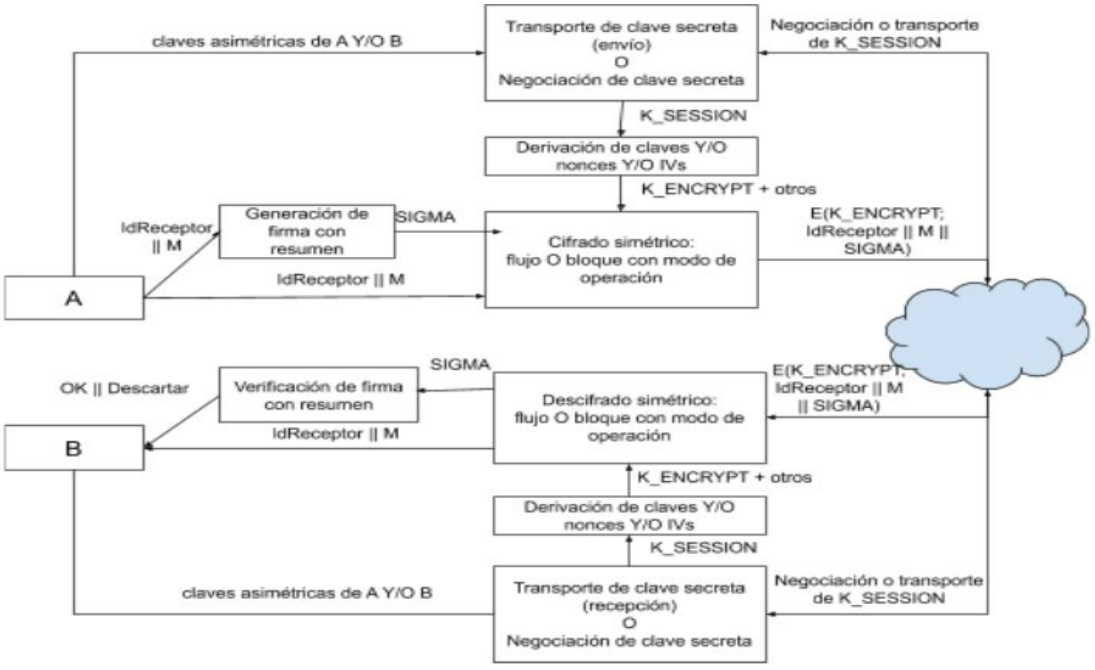


Figura 1: Modelo Sign-then-Encrypt

Negociación de clave con Diffie-Hellman.

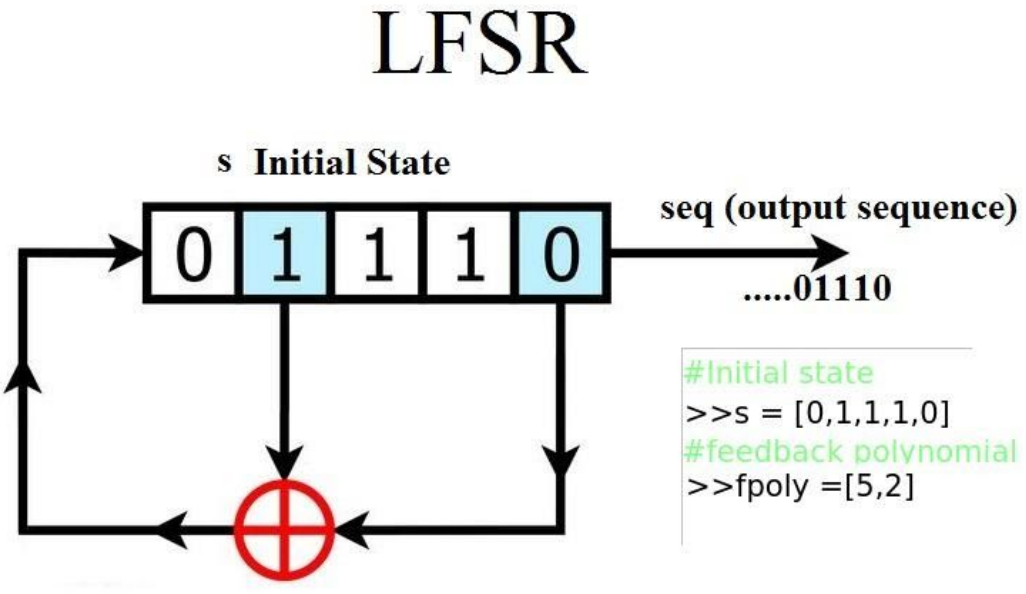
Es un algoritmo criptográfico que permite establecer claves entre partes que no han tenido contacto previo, permitiendo realizar ese contacto, e intercambio de claves de forma segura, por medio de un canal inseguro y de manera anónima.

El simple funcionamiento del algoritmo consiste en que, para cada parte implicada, se elige dos números públicos y un número secreto, con una fórmula matemática cada parte utiliza los dos números públicos y su número secreto para generar un resultado que se intercambiarán de forma pública, para luego, utilizando una última fórmula, se combina el resultado público con su número secreto y así generar la misma clave para ambas partes.
La teoría indica que el revertir esta función es tan compleja como calcular un logaritmo discreto, un problema NP.

Cifrado simétrico de flujo LFSR

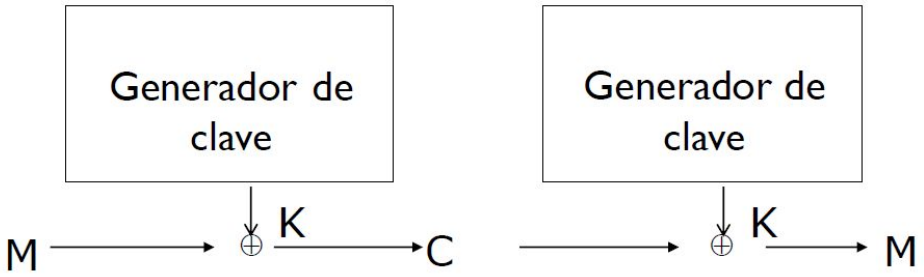
Un LFSR es un mecanismo de generación de números aleatorios en base una semilla cuyas siglas se traducen como registro de desplazamiento con retroalimentación lineal.

El LFSR se basa en un registro de desplazamiento en el cual la entrada es un bit proveniente de aplicar una función de transformación lineal a un estado anterior.
El registro opera de forma determinista en base a una semilla, la generación tiene un polinomio asociado que determina el periodo de la función, este método es de interés criptográfico cuando este polinomio asociado es primitivo ya que en esta circunstancia el periodo alcanzado es el máximo posible.

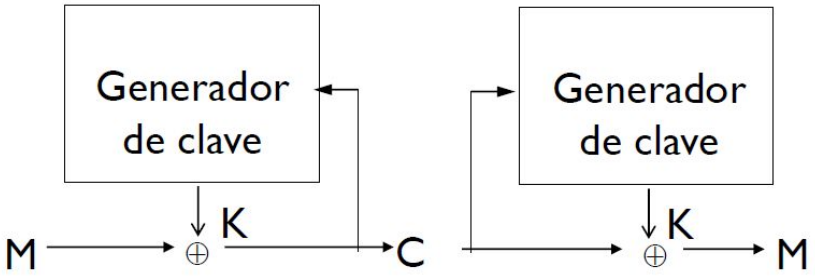


La aplicación criptográfica en este algoritmo sigue el modelo de un cifrador de flujo cuyos valores generados sirven para cifrar el mensaje, esto requiere algún tipo de sincronía entre emisor y receptor.

A continuación el modelo síncrono, que necesita de una sincronización externa.



Y el modelo auto síncrono, el cual se sincroniza mediante el propio mensaje cifrado.



Firma digital con ElGamal

El algoritmo de cifrado y descifrado asimétrico creado por Taher ElGamal tiene un funcionamiento similar al algoritmo Diffie-Hellman. Al estar basado sobre este, su funcionamiento se centra en la complejidad para calcular el logaritmo discreto. El uso de este algoritmo se centra tanto en la firma digital como en la cifra de datos. En la variante de este protocolo de comunicación segura su uso será para firmar el mensaje junto a un resumen generado con una función hash.

En firma digital, además, el algoritmo de ElGamal se considera como un algoritmo de tipo I, es decir, la firma está separada del mensaje y aparece como un apéndice. Además, este algoritmo es un algoritmo aleatorio: la firma depende de los apéndices elegidos.

Los parámetros iniciales para ElGamal son:

- Un primo ***p*** tal que el cálculo de logaritmos discretos módulo *p* sea complicado.
- Un elemento generador ***g*** de CG(p).
- Una función hash ***H*** consistente (con una alta resistencia a colisiones).

A continuación se describen los pasos para firmar un mensaje con ElGamal:

I. Alicia selecciona una clave x_A: $1 < x_A < p - 1$ e $y_A = g^{XA}$ (mód p).
II. Elegimos también un k aleatorio coprimo con $p - 1$.
III. Calculamos el par de apéndices (r, s):

$$r = g^k \text{ (mód p)}$$

$$s = (H(M) - x_A \bullet r) \bullet k^{-1} \text{ (mód p - 1)}$$

IV. Por tanto, el resumen del mensaje se corresponde con:

$$H(M) = x_A \bullet r + k \bullet s \text{ (mód p − 1)}$$

V. Alicia envía a Benito lo siguiente: M, r y s
VI. Benito procede a verificar lo obtenido con estas dos expresiones:

$$V_1 = y_A{}^r \bullet r^s \text{ (mód p)}$$

$$V_2 = g^{H(M)} \text{ (mód p)}$$

VII. Si las dos expresiones coinciden, Benito acepta el mensaje recibido por Alicia.

Función resumen TTH (Toy Tetragraph Hash)

La función hash tth genera un hash de 4 letras dado un mensaje de una longitud fija. La función utiliza letras en lugar de números binarios de forma que divide el texto en claro en fragmentos de 16 letras sin contar otros símbolos o espacios. En el caso de que no sea la longitud divisible por 16 habrá al final un resto llamado padding, una serie de A's. La función utiliza una matriz S de tamaño 2x2 que empieza con únicamente 0's pero que al final del algoritmo estará formado por las 4 letras resultantes.

El algoritmo se desarrolla de la siguiente forma:

Se transforma cada bloque de 16 letras a su valor numérico y se mete en una matriz M de tamaño 4x4 de izquierda a derecha.

Acto seguido se suma de forma recursiva la matriz M respecto a la siguiente fórmula:

S0,0 = S0,0 + (M0,0 + M1,0 + M2,0 + M3,0 mod 27) mod 27

S0,1 = S0,1 + (M0,1 + M1,1 + M2,1 + M3,1 mod 27) mod 27

S1,0 = S1,0 + (M0,2 + M1,2 + M2,2 + M3,2 mod 27) mod 27

S1,1 = S1,1+ (M0,3 + M1,3 + M2,3 + M3,3 mod 27) mod 27

Para su correcto funcionamiento debe ser con el alfabeto castellano, de forma que de usar otro la conversión entre letras y números (y viceversa) no funcionaría.

Al final de la última iteración S está formado por los números que, al transformar en sus equivalentes en letras se convierte en las 4 letras resultado.

Hash_C

Función Hash que consiste en la siguiente fórmula:
$Hash\_C = \sum i\, n_i = 1 \ mod\ 94$
Extraída del problema 4.6 de la recopilación de problemas de examen.

Firma del certificado con RSA

Es un algoritmo de cifrado asimétrico creado por Rivest, Shamir y Adleman en 1977, y el nombre del algoritmo son siglas de sus creadores. Es uno de los más utilizados de este tipo y sirve tanto para cifrar como descifrar información.

El ser asimétrico significa que utiliza dos claves, una pública y otra privada, de forma que cuando se quiere mandar un mensaje el emisor usa la clave pública del receptor para cifrar el mensaje y el receptor lo descifra usando su clave privada. En RSA cualquier mensaje cifrado con la privada puede descifrarse con la clave pública. Los pasos para usar este algoritmo son:

- Generación de claves: antes de cualquier intercambio es necesario que el emisor genere las claves

1. Se eligen dos números primos distintos p y q.
2. Se calcula n = p*q
3. Se calcula φ(n) =(p-1)*(q-1), siendo φ la función de Euler
4. Se elige un número primo e menor que φ y que sea coprimo con él, llamado e.
5. Se encuentra un d que cumpla e * d ≡ 1 mod φ (n).

La clave pública será (n, e) y la privada será (n, d).

- Cifrado

Alicia comparte con Benito su clave pública y guarda en secreto la clave privada. Para que Benito le pueda enviar un mensaje a Alicia lo transforma en un número entero. m que debe ser menor que n. Obviamente para transformarlo debe haber una forma de transformarlo que ambos hayan determinado y acordado.

El texto cifrado se calcula con: $c \equiv m^e \mod n$

Y Benito le manda c a Alicia.

- Descifrado

Alicia para descifrar el mensaje solo tiene que hacer $m \equiv c^d \mod n$

Es capaz de leerlo porque $c^d = (m^e)^d$

Para generar una firma usando RSA y así que el receptor pueda estar seguro de quién fue el emisor se siguen los siguientes pasos:

1. Usando el hash sacamos el resumen R = H(M) del mensaje M
2. Usando la clave privada el emisor firma el resumen: $F = D_{RSA}$ (R, d)
3. Usando la clave privada el emisor firma M: $C = R_{RSA}$ (R, d)
4. Se envía la (C, F) al receptor, siendo $(C, F) = (E_{RSA}(e,M), D_{RSA}(d,R))$
5. Se descifra el mensaje usando la clave pública y se verifica la firma

2.3. Claves asimétricas

Las claves asimétricas, halladas mediante intercambio de claves Diffie Hellman son:

Persona	**Alicia**	**Benito**
Clave Privada	a = 6	b = 8
Clave Pública	A = 11	B = 16

AC tiene su pareja de claves asimétricas para la firma mediante RSA, que es como sigue:

Persona	**AC**
Clave Privada	**(143, 7)**
Clave Pública	**(143, 103)**

2.4. Coherencia de representaciones y tamaños de bloques y espacios de trabajo

Para mantener una coherencia adecuada en las representaciones y tamaños de bloques se ha hecho lo siguiente:

El alfabeto que se ha utilizado para el cifrado simétrico, de manera que se conservase la coherencia en los datos es el siguiente:

A → 0	L → 11	V → 22
B → 1	M → 12	W → 23
C → 2	N → 13	X → 24
D → 3	Ñ → 14	Y → 25
E → 4	O → 15	Z → 26
F → 5	P → 16	a → 27
G → 6	Q → 17	b → 28
H → 7	R → 18	c → 29
I → 8	S → 19	d → 30
J → 9	T → 20	e → 31
K → 10	U → 21	

Por tanto, a la hora de pasar a binario el texto en claro se han codificado las letras en grupos de 5 bits. Las letras minúsculas se han añadido para que esos valores tuviesen una representación después de cifrar el mensaje en claro.

Como el cifrado simétrico que se ha utilizado es de flujo, no hay ningún detalle má que comentar acerca de la estructura de esta parte.

3.Comunicación de las partes

Intercambio de claves Diffie Hellman

Se realiza Diffie Hellman, la verificación de certificados se debe dar cuando Alicia y Benito se intercambian las claves públicas, se ha puesto en el apartado siguiente por claridad.

p = 37

g = 5

$A=(g^a \bmod p) = 11$

$B=(g^b \bmod p) = 16$

ALICIA:

a=6 (Privada)

A = 11 (Pública)

$K=B^a \bmod p = 10$

BENITO:

b=8 (Privada)

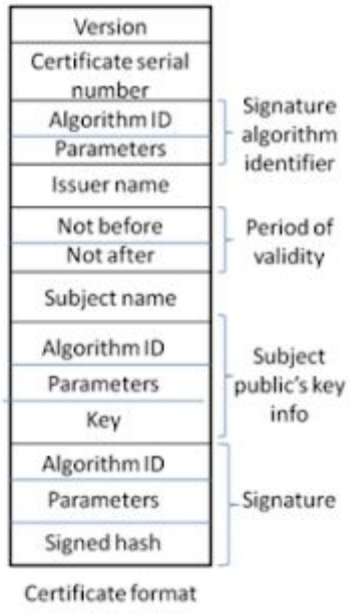
B = 16 (Pública)

$K = A^b \bmod p = 10$

$K = g^{(a \cdot b)} \bmod p = g^{(b \cdot a)} \bmod p = K = 10 = K\_SESSION = SEED$

Certificados de Clave Pública

Campos del Certificado (X509 Simplificado)
Función Hash a usar: *Hash_C*



Certificate format

Certificado AC

Versión: 1

Serial Number: 0

Signature Algorithm Identifier: 1 (RSA con Hash C)

Issuer Name: 1 (CA)

Period of Validity: 2020 - 2022

Subject Name: 0 (CA)

Subject Public´s Key Info: 1 (RSA con Hash C), exp (103), mod (143)

Signature: 1 (RSA con Hash C)

Firma : 2

Firma RSA de AC

p= 11, q =13

n=143

φ(n) =(p-1)*(q-1) = 120 - > e = 103

clave pública = (n, e) = (143, 103)

d * e ≡ 1 mod φ (n) - > d * 103 ≡ 1 mod 120

Hash C = (1+0+1+1+2020+2022+0+1+103+143+1) mod 94 = 4293 mod 94 = **63**

Cálculo del inverso (Por Euler)

103d mod 120 = 1

a = 103 n = 120 Φ(120) = 32

103^(32-1) mod 120 = **7 = d**

clave privada = (n, d) = (143, 7)

Firmamos el Hash con la clave privada:

F = $Hash^d$ mod n = 63^{7} mod 143 = **2 = Firma**

Certificado Alicia

Versión: 1

Serial Number: 1

Signature Algorithm Identifier: 1 (RSA con Hash C)

Issuer Name: 1 (CA)

Period of Validity: 2020 - 2022

Subject Name: 1 (Alicia)

Subject Public´s Key Info: 2 (Intercambio Diffie Hellman), A = 11

Signature: 1 (RSA con Hash C)

Firma : 46

Firma RSA de Alicia

Hash Alicia = (1+1+1+1+2020+2022+1+2+11+1) mod 94 = 4061 mod 94 = **19**

Firmamos el Hash con la clave privada:

F = $Hash^d$ mod n = 19^{7} mod 143 = **46 = Firma**

Certificado Benito

Versión: 1

Serial Number: 2

Signature Algorithm Identifier: 1 (RSA con Hash C)

Issuer Name: 1 (CA)

Period of Validity: 2020 - 2022

Subject Name: 2 (Benito)

Subject Public´s Key Info: 2 (Intercambio Diffie Hellman), B = 16

Signature: 1 (RSA con Hash C)

Firma : 104

Firma RSA de Benito

Hash C = (1+2+1+1+2020+2022+2+2+16+1) mod 94 = 4068 mod 94 = **26**

Firmamos el Hash con la clave privada:

F = $Hash^d$ mod n = 26^{7} mod 143 = **104 = Firma**

Intercambio de Certificados

Se necesita verificar la identidad de AC para comprobar la procedencia de la clave pública para deshacer la firma, estos datos se transmiten mediante un canal seguro con AC.

Alicia verifica la identidad de AC

Hash AC = (1+0+1+1+2020+2022+0+1+103+143+1) mod 94 = 4293 mod 94 = **63**

Calculado a partir del mensaje

m = c^e mod n = 2^103 mod 143 = **63**

Se cumple la integridad

Alicia verifica la identidad de Benito

Hash Benito = (1+2+1+1+2020+2022+2+2+16+1) mod 94 = 4068 mod 94 = **26**

Calculado a partir del mensaje

m = c^e mod n = 104^103 mod 143 = **26**

Se cumple la integridad

Benito verifica la identidad de AC

Hash AC = (1+0+1+1+2020+2022+0+1+103+143+1) mod 94 = 4293 mod 94 = **63**

Calculado a partir del mensaje

m = c^e mod n = 2^103 mod 143 = **63**

Se cumple la integridad

Benito verifica la identidad de Alicia

Hash Alicia = (1+1+1+1+2020+2022+1+2+11+1) mod 94 = 4061 mod 94 = **19**

m = c^e mod n = 46^103 mod 143 = **19**

Se han verificado todas las identidades

Firma del mensaje que manda Alicia

Cálculo del Hash del mensaje y concatenación

Id. Alicia = A

M = BOB

SIGMA = Hash(M)

TTH (M):

M:

1	15	1	0
0	0	0	0
0	0	0	0
0	0	0	0

S:

0	0
0	0

□

1 mód 27	15 mód 27
1 mód 27	0 mód 27

□

B	O
B	A

SIGMA = BOBA = 1 + 15 + 1 + 0 (mód 27) = 17

Id. Alicia || M || SIGMA = ABOBBOBA

Firma del mensaje con ElGamal

p = 37

k = 25

g = 5

r = 5^{25} (mod 37) = 19

s = (17 - 6 * 19) * 25^{-1} (mód 36) = 35

Cálculo del inverso:

25x mod 36

a = 25 n = 36 Φ(36) = 12

25^{12-1} mod 36 = **= x = 25^{-1} = 13**

Cifrado del Mensaje que manda Alicia

SEED = 10_{10} = 1010_2

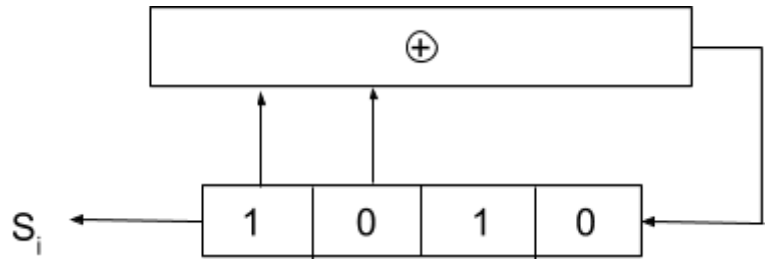
Polinomio de conexión = $x^4 + x^3 + 1$

Letra	**Decimal**	**Binario**
A	0	00000
B	1	00001
O	15	01111
B	1	00001
B	1	00001
O	15	01111
B	1	00001
A	0	00000

La cadena de bits resultante es:

M_f = 00000 00001 01111 00001 00001 01111 00001 00000

El registro LFSR sería como sigue:



1	0	1	0
0	1	0	1
1	0	1	1
0	1	1	1
1	1	1	1
1	1	1	0
1	1	0	0
1	0	0	0
0	0	0	1
0	0	1	0
0	1	0	0
1	0	0	1
0	0	1	1
0	1	1	0
1	1	0	1
1	0	1	0

A partir de este ya es cíclico.

M_f = 00000 00001 01111 00001 00001 01111 00001 00000

S_i = 10101 11100 01001 10101 11100 01001 10101 11100

$M_f \oplus S_i$ = 10101 11101 00110 10100 11101 00110 10100 11100

Secuencia cifrada: UcGTcGTb

Descifrado del mensaje recibido por Benito

Secuencia cifrada recibida: UcGTcGTb

SEED = $10_{10} = 1010_2$

La sincronización de los LFSR se producirá de forma externa, por lo tanto, Benito tendrá la misma secuencia S_i.

S_i = 10101 11100 01001 10101 11100 01001 10101 11100

SC = 10101 11101 00110 10100 11101 00110 10100 11100

$SC \oplus S_i = M_f$ = 00000 00001 01111 00001 00001 01111 00001 00000

Mensaje en claro = ABOBBOBA

Descomposición:

id || Mensaje || Hash

A = id Alicia

BOB = mensaje en claro

HASH = BOBA

Verificación de la Firma recibida por Benito

Datos recibidos por Benito:

HASH = BOBA = 1 + 15 + 1 + 0 = 17

r = 19

s = 35

g = 5

Clave pública de Alicia: $y_A = 11$

Se comprueban las verificaciones:

$V_1 = 11^{19} * 19^{35}$ (mód 37) = 22

$V_2 = 5^{17}$ (mód 37) = 22

$V_1 = V_2 = 22$

Por tanto, la firma se verifica y es válida.

Firma del mensaje enviado por Benito

Cálculo del Hash del mensaje y concatenación

Id. Benito = B

M = ROSS

SIGMA = Hash(M)

TTH (M):

M:

18	15	19	19
0	0	0	0
0	0	0	0
0	0	0	0

S:

0	0
0	0

□

18 mód 27	15 mód 27
19 mód 27	19 mód 27

□

R	O
S	S

SIGMA = ROSS = 18 + 15 + 19 + 19 (mód 27) = 17

Id. Benito || M || SIGMA = BROSSROSS

Firma del mensaje con ElGamal

p = 37

k = 11

g = 5

r = 5^{11} (mód 37) = 2

s = (71 - (8 * 2)) * 11^{-1} (mód 36) = 5

Cálculo del inverso:

11x mod 36

a = 11 n = 36 Φ(36) = 12

11^{12-1} mod 36 = **= x = 11^{-1} = 23**

Cifrado del Mensaje enviado por Benito

SEED = 10_{10} = 1010_2

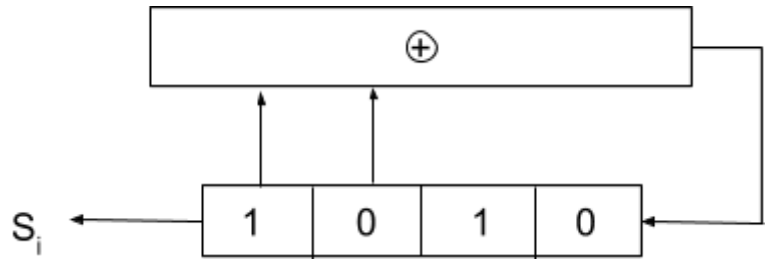
Polinomio de conexión = $x^4 + x^3 + 1$

letra	Decimal	Binario
B	1	0001
R	18	10010
O	15	1111
S	19	10011
S	19	10011
R	18	10010
O	15	1111
S	19	10011
S	19	10011

La cadena de bits resultante es:

M_f =00001 10010 01111 10011 10011 10010 01111 10011 10011

El registro LFSR sería como sigue:



1	0	1	0
0	1	0	1
1	0	1	1
0	1	1	1
1	1	1	1
1	1	1	0
1	1	0	0
1	0	0	0
0	0	0	1
0	0	1	0
0	1	0	0
1	0	0	1
0	0	1	1
0	1	1	0
1	1	0	1
1	0	1	0

A partir de este ya es cíclico.

M_f = 00001 10010 01111 10011 10011 10010 01111 10011 10011

S_i = 10101 11100 01001 10101 11100 01001 10101 11100 01001

$M_f \oplus S_i$ = 10100 01110 00110 00110 01111 11011 11010 01111 11010

Secuencia cifrada = TÑGGOaZOZ

Descifrado del mensaje recibido por Benito

Secuencia cifrada recibida: UcGTcGTb

SEED = $10_{10} = 1010_2$

La sincronización de los LFSR se producirá de forma externa, por lo tanto, Benito tendrá la misma secuencia S_i.

S_i = 10101 11100 01001 10101 11100 01001 10101 11100 01001

SC = 10100 01110 00110 00110 01111 11011 11010 01111 11010

$S_i \oplus$ SC = 00001 10010 01111 10011 10011 10010 01111 10011 10011

mensaje en claro= BROSSROSS

Descomposición:

id || Mensaje || Hash

B = id Alicia

ROSS = mensaje en claro

HASH = ROSS

Verificación de la Firma recibida por Alicia

Datos recibidos por Alicia:

HASH = ROSS = 18 + 15 + 19 + 19 = 71

r = 2

s = 5

g = 5

Clave pública de Benito: $y_B = 16$

Se comprueban las verificaciones:

$V_1 = 16^2 * 2^5$ (mód 37) = 15

$V_2 = 5^{71}$ (mód 37) = 15

$V_1 = V_2 = 15$

Por tanto, la firma se verifica correctamente.