A Diffie-Hellman Compact Model Over Non-Commutative Rings Using Quaternions

J. A. Kamlofsky – J. P. Hecht – O. A. Hidalgo Izzi – S. Abdel Masih

***Abstract*— The key exchange cryptographic protocol created by Whitfield Diffie and Martin Hellman was one of the pioneers of asymmetric cryptography (AC). Many of the asymmetric cryptography protocols are based on operations performed in commutative algebraic structures. Today many of them are vulnerable to subexponential or quantum attacks. The development of algorithms in non-commutative structures can strengthen these protocols. This line of development is an actual growing trend.**

In particular Hecht (2009) has presented a key exchange model based on the Diffie-Hellman protocol over non-commutative rings using matrices of order four with Z_{256} elements, particularly interesting to provide cryptographic security to devices with low computing power.

The set of quaternions, like the set of all the matrices form non-commutative ring structures. However, quaternions have more compact notation and shown lower runtimes in many comparable operations.

In this paper we propose the use of quaternions in this key exchange protocol, and present experimental results showing lower run-times in this cryptosystem when using quaternions at equivalent security.

***Keywords*— Asymmetric cryptography, quaternions in cryptography, quaternions, non-commutative cryptography, post-quantum cryptography.**

I. Introducción: Estado del arte y trabajos relacionados

A. Tendencias en la criptografía asimétrica

Desde la década pasada ha crecido el interés en el desarrollo de criptosistemas asimétricos alternativos que sean resistentes a los ataques de complejidad subexponencial [1, 2] y los potenciales ataques vía computadora cuántica [3] que afectan a la actual generación de algoritmos basados en campos numéricos [4] y de campos algebraicos reducibles a ellos [5]. La mayoría de estos esquemas coinciden en lo que se denomina colectivamente como criptografía post-cuántica [6] y por su naturaleza algebraica como criptografía no conmutativa [7]. En tal sentido se han logrado importantes avances en la definición de estructuras que planteen nuevas funciones trampa de una vía para las cuales no se conocen y probablemente no existan ataques de complejidad polinómica, lógicamente esta última conjetura debe tomarse con sumo recaudo. En esta línea se ha presentado un esquema de distribución de claves Diffie-Hellman [8] basado en un anillo de polinomios matriciales cuya seguridad reside en el problema de la descomposición simétrica generalizada (GSDP) [9]. En el citado trabajo se presentan los antecedentes del método y el desarrollo algorítmico. El sistema se ha caracterizado como compacto porque no requiere el uso de bibliotecas de precisión extendida y por ende es aplicable a entornos de bajo poder computacional y memoria RAM reducida (smartcards, token USB criptográficos, etc.)

B. Estructuras de Anillo:

Definición 1: Sea A un conjunto no vacío y dos operaciones internas, usualmente llamadas suma (+) *y producto* (.). *La terna* $(A, +, .)$ *es un anillo si y sólo si*

I) $(A, +)$ *es un grupo conmutativo.*

II) $(A, .)$ *es un semigrupo.*

III) *El producto es distributivo a izquierda y derecha respecto de la suma.*

Definición 2: Un anillo $(A, +, .)$ *es conmutativo si*

$$\forall\, a, b \in A \Rightarrow a \,.\, b = b \,.\, a$$

Por lo tanto, el anillo será no conmutativo si

$$\exists\ a, b \in A : a.\ b \neq b \,.\, a$$

Definición 3: Un anillo $(A, +, .)$ *es un anillo con identidad o unidad si existe un elemento neutro o identidad multiplicativa, denotado con 1, tal que*

$$a \,.\, 1 = 1 \,.\, a = a \ \ \forall\, a \in A$$

Definición 4: Un anillo con *identidad es un anillo de división si sus elementos no nulos son inversibles. Es decir,*

$$\forall\, a \in A : a \neq 0\ \exists\ x, y \in A : a.\ x = 1\ \ \wedge\ y.\ a = 1$$

El primer anillo de división no conmutativo fue el anillo de los cuaterniones [10].

La importancia de destacar la no conmutatividad de un anillo de división radica en el teorema de Wedderburn [11] que establece que todo anillo de división finito es conmutativo [12]. Luego, disponer de infinitos elementos en el anillo a utilizar agregará mayor complejidad a la resolución del algoritmo de cifrado.

Otros ejemplos de anillos de división no conmutativos son los siguientes:

I) Sean $(A, +, .)$ un anillo y $n \in N$. El conjunto de matrices cuadradas de orden n con coeficientes en A, simbolizado por $M_n(A)$, es un anillo con respecto a las operaciones usuales de suma y producto de matrices. Si $n>1$, entonces $M_n(A)$ no es conmutativo.

II) Los Octoniones o números de Cayley pertenecen también a este tipo de anillos. Son la extensión no asociativa de los cua-

terniones y cada octonión forma y representa una combinación lineal de la base: *1, e1, e2, e3, e4, e5, e6, e7*.
III) Sedeniones, Tessarines, cocuaterniones o bicuaterniones son otros ejemplos de anillos no conmutativos.

C. Algebra de Cuaterniones:

Definición 5: Sea $(A, +, .)$ *un anillo conmutativo con unidad. Un cuaternión con coeficientes en A es un número q de la forma*

$$q = a + i\,.b + c.\,j + d\,.k$$

Donde
$a, b, c, d \in A$
i, j, k son unidades imaginarias que verifican las igualdades
$i^2 = j^2 = k^2 = -i\,.\,j\,.\,k = -1$
y además
$i\,.\,j = -j\,.\,i = k;$
$j\,.\,k = -k\,.\,j = i;$
$i\,.\,k = -k\,.i = j.$

El conjunto de los cuaterniones es H, es decir,

$$H = \{\, a + i\,.\,b + c\,.\,j + d\,.\,k \;: a, b, c, d \in A \,\}$$

Éstos fueron creados en 1843 por William Hamilton [13] para poder demostrar el Teorema de Euler, que afirmaba que todo número natural *n* puede escribirse como suma de cuatro cuadrados perfectos. Más precisamente,

Si $n \in N$ *entonces* $n = a^2 + b^2 + c^2 + d^2$ *, con* $a, b, c, d \in Z$

Los cuaterniones forman una estructura de Anillo de división no conmutativo.

Tienen una notación compacta y resultan muy sencillos para trabajar. Son muy eficientes en comparación con la operación de matrices: requieren menor cantidad de operaciones básicas y menor espacio de almacenaje. Además, solucionan el problema de *pérdida de dimensionalidad* conocido como *gimbal lock* [14].

Se los utiliza mayormente en aplicaciones que requieran secuencias de rotaciones en el espacio en tiempo real: navegación aeroespacial, realidad virtual, y visión robótica [15-18]. En [19] se muestra una comparación de tiempos de ejecución en la realización de rotaciones en el espacio con matrices y con cuaterniones. Hay varios trabajos que presentan aplicaciones e implementaciones criptográficas mediante el uso de cuaterniones [20–23].

Diferentes notaciones o representaciones de cuaterniones:

Forma Vectorial:
$q = a + i\,.b + c.\,j + d\,.k$

Forma Cartesiana:
$q = (a, b, c, d)$

Forma Trigonométrica:
$q = |q|.(\, cos(\alpha/2) + v'.\, sen(\alpha/2))$

con: $|q| = (a^2 + b^2 + c^2 + d^2)^{1/2}$
$\alpha/2 = \arccos(a/|q|)$
$v' = (b\,,\, c\,,\, d)/\,(\, b^2 + c^2 + d^2)^{1/2}$

Forma Matricial:

$$q = \begin{pmatrix} a & -b & d & -c \\ b & a & -c & -d \\ -d & c & a & -b \\ c & d & b & a \end{pmatrix}$$

Forma exponencial:
$e^q = e^{a+b.i+c.j+d.k}$. *Si* $q = s + v \Rightarrow\ e^q = e^s\,(cos|v| + v'.sen|v|\,)$

Definiciones Básicas:

Sea un cuaternión $q = (a\,,\, b\,,\, c\,,d)$ *entonces*

Cuaternión conjugado de q:

El conjugado del cuaternion q es: $q^* = (a\,,\, -b\,,\, -c\,,\, -d).$

Norma del cuaternión q:
$|q| = (q.q^*)^{1/2} = (q^*.q)^{1/2} = (a^2 + b^2 + c^2 + d^2)^{1/2}$

Cuaternión unitario (o normalizado):
Si $q \neq (0, 0, 0, 0)$, *el cuaternión unitario asociado a q es:*
$q_1 = q\,/\,|q|$

Opuesto del cuaternión q: $-q = (-a\,,\, -b\,,\, -c\,,\, -d)$

Inverso del cuaternión q:
Si $q \neq (0, 0, 0, 0)$ *entonces su inverso es:* $q^{-1} = q\,/\,|q|^2$

Cuaternión puro:
El cuaternión q es puro si su parte real es nula. Es decir, si $a = 0.$

Operaciones básicas con cuaterniones:

Suma, resta y producto de un escalar por un cuaternión:
Se realiza de la misma forma que para cualquier vector de 4 dimensiones.

Producto:
Sean $q_1 = (w_1, x_1, y_1, z_1)$ y $q_2 = (w_2, x_2, y_2, z_2)$, el producto $q_1\,.q_2$ se define:
$q_1\,.q_2 = (w_1.w_2 - x_1.x_2 - y_1.y_2 - z_1.z_2\,,\, w_1.x_2 + x_1.w_2 + y_1.z_2 - z_1.y_2\,,$
$w_1.y_2 - x_1.z_2 + y_1.w_2 + z_1.x_2\,,\, w_1.z_2 + x_1.y_2 - y_1.x_2 + z_1.w_2)$

Nota: El producto de cuaterniones no es conmutativo.

Cociente:
Sean dos cuaterniones $q_1 = (w_1, x_1, y_1, z_1)$ y $q_2 = (w_2, x_2, y_2, z_2)$, su cociente $q_3 = (w_3, x_3, y_3, z_3) = q_1\,/\,q_2$ se define:
$q_3 = q_1\,/\,q_2 = q_1.(q_2)^{-1} = q_1.(q_2\,/\,|q_2|^2)$

Potencia:
Sea el cuaternión expresado en forma trigonométrica:
$q = |q|\,.\,[cos(\alpha/2) + v'.sen(\alpha/2)]$ y sea *n* entero. Entonces
$q^n = |q|^n\,.\,[cos\,(n.\alpha/2) + v'.sen(n.\alpha/2)]$

Algunas particularidades:

Vectores en el espacio:
Dado un vector $v = (x\,,\, y\,,\, z)$ en el espacio, se le puede asociar un cuaternión puro de la forma $q = (0,\, x\,,\, y\,,\, z).$ Por esta razón, a los vectores del espacio se los llama *Cuaterniones puros*.

Producto entre cuaterniones puros:
Sean: $q_1 = (0, x_1, y_1, z_1)$, $q_1 = (0,\ x_2,\ y_2, z_2).$ Su producto será:
$q_3 = q_1\,.\,q_2 = (0, x_1, y_1, z_1)\,.\,(0, x_2, y_2, z_2) = (w_3, x_3, y_3, z_3)$ con:
$w_3 = -x_1.x_2 - y_1.y_2 - z_1.z_2,\ x_3 = y_1.z_2 - z_1.y_2,\ y_3 = z_1.x_2 - x_1.z_2,\ z_3 =$

$x_1.y_2 - y_1.x_2$, de donde: w_3 es la expresión del producto escalar multiplicado por (-1) y (x_3, y_3, z_3) es el producto vectorial tal como hoy se lo conoce.

La anti-conmutatividad del producto de cuaterniones puros: Se puede verificar fácilmente que $q_1 . q_2 \neq q_2 . q_1$ ya que $q_1 . q_2 = (w_3, x_3, y_3, z_3)$, $q_2 . q_1 = (w_3, -x_3, -y_3, -z_3)$.

Producto del cuaternión puro y su conjugado:
Sean $q = (0, x, y, z)$ y $q^* = (0, -x, -y, -z)$, entonces:
$q.q^* = (x_2 + y_2 + z_2, 0, 0, 0)$

II. Objetivo del trabajo, motivacion y relevancia del tema

A. Objetivo del trabajo:

El objetivo del presente trabajo es el desarrollo de una mejora al modelo compacto original DH-C [9], empleando una estructura algebraica de cuaterniones en reemplazo de las matrices modulares.

B. Motivación:

El uso de Cuaterniones ya presentó ventajas frente al uso de matrices cuadradas en diversas aplicaciones [14 – 19]. También se presentaron aplicaciones criptográficas con cuaterniones [20 - 23]. En el trabajo de Hecht [9] se usaron polinomios de grado 16 como claves privadas. Y el cálculo de las claves de sesión se inician con el cálculo de polinomios de matrices de orden 4 con elementos en Z_{256}. A la complejidad de estos cálculos se contrapone la gran simpleza del cálculo de potencias de cuaterniones unitarios. Así resultó evidente que podría obtenerse una importante mejora en los tiempos de cómputo de este protocolo.

C. Relevancia del tema:

Este desarrollo se puede aplicar directamente al intercambio de claves en plataformas de porte reducido. La optimización que se presenta mejora significativamente el desempeño algorítmico y representa un avance en este campo de la criptografía asimétrica. Cabe destacar que la misma algoritmia podrá ser empleada en otras aplicaciones criptográficas tales como cifrado ElGamal generalizado, firma digital, transporte de claves y pruebas de autenticación de conocimiento cero [4].

III. Implementación del protocolo con Cuaterniones

A. Acerca de esta implementación:

El protocolo aquí presentado tiene mucha similitud con el protocolo DH-C presentado en [9]. Difiere en dos etapas de normalización presentes en esta implementación con cuaterniones, y en la longitud de 32 bits de cada porción de clave generada, además de usarse cuaterniones en lugar de matrices.

Las dos normalizaciones realizadas permiten que el cálculo de las potencias de cuaterniones se limiten a multiplicaciones del argumento de senos y cosenos por el valor del exponente logrando operaciones mucho más sencillas que potencias de matrices, manteniendo aún la esencia del cuaternión original.

Las normalizaciones mencionadas, permiten también, mantener los cálculos dentro de numeración de precisión simple, lo cual es necesario para operar en procesadores de menor porte.

B. El protocolo DH-C con cuaterniones:

Inicialización:

(a) ALICE elige dos cuaterniones A y B no nulos con coeficientes en Z_{256} y m, n elementos no nulos de Z_{16}.

(b) ALICE calcula q_A, q_B: la normalización de A y B.

(c) ALICE elige como clave privada, un polinomio no nulo entero $f(x)$ en $Z_{16}(x)$ con exponentes y coeficientes en Z_{16} tal que $f(q_A) \neq 0$.

(d) BOB elige como clave privada, un polinomio no nulo entero $h(x)$ en $Z_{16}(x)$ con exponentes y coeficientes en Z_{16} tal que $h(q_A) \neq 0$.

(e) ALICE envía a BOB q_A, q_B, m, n por el canal inseguro.

Cálculo de tokens:

(f) ALICE calcula su token: Con $f'(q_A)$ normalización de $f(q_A)$, luego: $r_A = f'(q_A)^m . q_B . f'(q_A)^n$ y lo envía a BOB por el canal inseguro.

(g) BOB calcula su token: Con $h'(q_A)$ normalización de $h(q_A)$, luego: $r_B = h'(q_A)^m . q_B . h'(q_A)^n$ y lo envía a ALICE por el canal inseguro.

Cálculo de Claves de sesión:

(h) ALICE calcula su clave: $k_A = f'(q_A)^m . r_B . f'(q_A)^n$.

(i) BOB calcula su clave: $k_B = h'(q_A)^m . r_A . h'(q_A)^n$.

Puede verificarse: $k_A = f'(q_A)^m . h'(q_A)^m . q_B . h'(q_A) . f'(q_A)^n$

Y además: $k_B = h'(q_A)^m . f'(q_A)^m . q_B . f'(q_A)^n . h'(q_A)^n$

(j) ALICE calcula: $K_A = (k_A.256)(mod\ 256)$

(k) BOB calcula: $K_B = (k_B.256)(mod\ 256)$

Finalmente: $K_A = K_B$.

La Fig. 1 muestra un esquema de funcionamiento del protocolo.

C. Un Ejemplo numérico

Inicialización:

(a) ALICE elige : $m = 14$, $n = 9$, y los cuaterniones: $A = (175, 157, 200, 46)$, $B = (236, 121, 118, 26)$.

(b) ALICE Normaliza los cuaterniones A y B (se muestran con 3 decimales):
$q_A = (0.560, 0.503, 0.640, 0.147)$
$q_B = (0.809, 0.415, 0.404, 0.089)$

(c) ALICE elige clave privada:
$f(x) = 12x^{15} + 6x^{11} + 3x^7 + 12$

(d) Bob elige clave privada:
$h(x) = x^{14} + 15x^{10} + 3x^8 + 10x^5 + 3x + 7$

(e) Alice envía a Bob: 14, 19, $(0.560, 0.503, 0.640, 0.147)$,

(0.809, 0.415, 0.404, 0.089) por el canal inseguro.

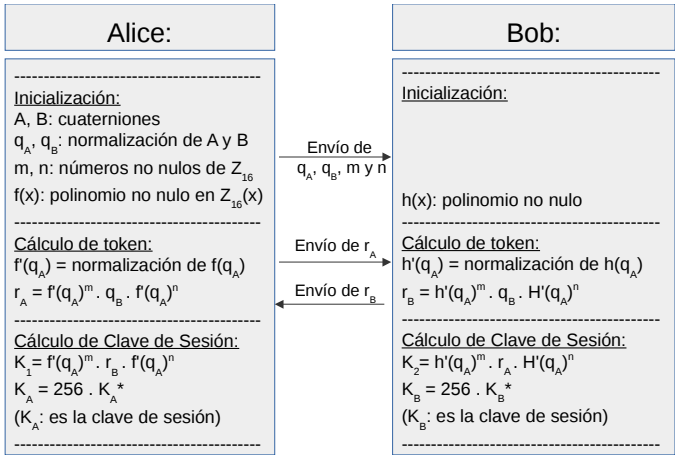


Figura 1. Diagrama de funcionamiento del protocolo Diffie-Hellman Compacto usando cuaterniones.

Cálculo de tokens:

(f) Alice calcula su token:
$f(0.560, 0.503, 0.640, 0.147) = 12.(0.560, 0.503, 0.640, 0.147)^{15} + 6.(0.560, 0.503, 0.640, 0.147)^{11} + 3.(0.560, 0.503, 0.640, 0.147)^{7} + 11 = (-4.709, 3.181, 4.053, 0.932)$. Su normalización: $f'(q_A) = (-0.668, 0.451, 0.575, 0.132)$
$r_A = (-0.668, 0.451, 0.575, 0.132)^{14} . (0.809, 0.415, 0.404, 0.089) . (-0.668, 0.451, 0.575, 0.132)^{9}$
$r_A = (-0.661, 0.043, -0.093, 0.029)$ y lo envía a Bob.

(g) Bob calcula su token:
$h(0.560, 0.503, 0.640, 0.147) = (0.560, 0.503, 0.640, 0.147)^{14} + 15(0.560, 0.503, 0.640, 0.147)^{10} + 3.(0.560, 0.503, 0.640, 0.147)^{8} + 10.(0.560, 0.503, 0.640, 0.147)^{5} + 3.(0.560, 0.503, 0.640, 0.147) + 7 = (-10.252, -4.205, -5.357, -1.232)$. Su normalización: $h'(q_A) = (-0.828, -0.339, -0.433, -0.067)$
$r_B = ((-0.828, -0.339, -0.433, -0.067)^{14} . (0.809, 0.415, 0.404, 0.089) . ((-0.828, -0.339, -0.433, -0.067)^{9}$
$r_B = (0.317, -0.201, -0.319, -0.067)$ y lo envía a Alice.

Cálculo de Claves de sesión:

(h) Alice calcula su clave:
$k_A = (-0.668, 0.451, 0.575, 0.132)^{14} . (0.317, -0.201, -0.319, -0.067) . (-0.668, 0.451, 0.575, 0.132)^{9} = (-0.069, 0.200, -0.319, 0.082)$
$K_A = ((-0.069, 0.200, -0.319, 0.082).256) (mod\ 256)$
$K_A = (239, 51, 62, 21)$

(i) Bob calcula su clave:
$k_B = ((-0.828, -0.339, -0.433, -0.067)^{14} . (-0.661, 0.043, -0.093, 0.029) . ((-0.828, -0.339, -0.433, -0.067)^{9} = (-0.069, 0.200, 0.243, 0.082)$
$K_B = ((-0.069, 0.200, -0.319, 0.082).256) (mod\ 256)$
$K_B = (239, 51, 62, 21)$

IV. Resultados Experimentales

El experimento realizado consiste en 25 pruebas en las cuales, en cada una de ellas, se obtuvieron 10.000 claves de 128 bits usando matrices y 10.000 claves de 128 bits usando cuaterniones. Durante cada prueba se midieron los tiempos de ejecución.

A. Equipamiento Usado

El computador usado contiene un procesador AMD FX(tm)-8320 Eight-Core Processor × 8 de 64 bits con 8Gb de memoria RAM.

En el mismo se instaló una distribución de Ubuntu 14.04 LTS (Long Time Support) de Cannonical, el cual tiene un núcleo Linux Debian.

Ambos algoritmos se programaron en Python 2.7.6.

B. Comparación Experimental

La tabla 1 muestra 25 registros de mediciones de tiempos de ejecución necesarios para obtener claves de 128 bits usando el protocolo Diffie-Hellman Compacto con matrices y con cuaterniones. Por cada experimento se realizaron 10.000 operaciones de obtención de claves de 128 bits.

El tiempo de ejecución promedio para la obtención de 10.000 claves de 128 bits usando matrices fue de 8,585102 segundos mientras que con la implementación con cuaterniones, ese tiempo se redujo a 4,928539 segundos.

TABLA 1: TIEMPOS DE EJECUCIÓN EN LA IMPLEMENTACIÓN DEL PROTOCOLO DIFIE-HELLMAN COMPACTO USANDO MATRICES Y CUATERNIONES.

	CPU Time (s)			CPU Time (s)	
Nº test	Matrices	Cuaterniones	Nº test	Matrices	Cuaterniones
1	8,600102	4,837513	14	8,581125	4,946582
2	8,506882	4,878939	15	8,627992	4,953737
3	8,693089	4,859479	16	8,606326	4,937578
4	8,597678	4,905438	17	8,607832	4,970296
5	8,699418	4,913928	18	8,481426	4,930410
6	8,563409	4,914061	19	8,492134	4,894224
7	8,584500	4,869740	20	8,499029	4,913316
8	8,499618	4,923639	21	8,677493	4,914754
9	8,659884	5,120607	22	8,565197	4,925393
10	8,550620	4,946923	23	8,607323	4,886488
11	8,662200	4,866079	24	8,648141	5,111923
12	8,529670	4,899871	25	8,617395	4,941260
13	8,469057	4,951300			

Nota 1: Como el coeficiente de variabilidad para ambos casos es menor a 0,10 se acepta al promedio como un indicador de tendencia central adecuado.

Nota 2: En este experimento se obtuvieron 1.000.000 de claves de 32 bits sin error.

V. Ventajas de la Solución Propuesta

A. Mejora global en los tiempos de cómputo

Con los resultados experimentales aquí presentados y en las condiciones del experimento, puede afirmarse que la implementación del protocolo Diffie-Hellman Compacto con cuaterniones es 42,59% más veloz que empleando matrices, tal cual fue presentado originalmente.

B. Flexibilidad para la obtención de claves de longitud menor a 128 bits

Como cada cuaternión resultado presenta porciones de 32

bits de clave, la implementación aquí presentada puede usarse para la obtención claves de módulos de 32 bits en lugar de módulos de 128 bits, reduciendo aún más los tiempos de obtención para claves de longitud distinta a *k* x128 bits.

C. Inmunidad a ataques cuánticos y a ataques de complejidad subexponencial

Dado que los cuaterniones conforman estructuras algebraicas de anillos no conmutativos, para estas clases de estructuras, no se conocen ataques mediante computadoras cuánticas ni de complejidad subexponencial.

D. Solución apta para procesadores de bajo porte

Los cálculos se realizaron dentro de numeración de precisión simple, lo cual es necesario para que puedan realizarse en procesadores de menor porte.

VI. Conclusiones

En este trabajo se presentó una implementación con cuaterniones del modelo DH-C [9], inmune a ataques de complejidad subexponencial y de computadoras cuánticas y apto para procesadores de menores portes.

Se logró la obtención de claves de sesión de 128 bits con menores tiempos de cómputo en comparación con la solución original. Dado que con cada cuaternión se obtienen bloques de 32 bits de clave, en caso de requerirse claves con longitud diferente a 128 bits, esta solución proporciona mejoras adicionales en los tiempos de cómputo.

En síntesis, se presenta un aporte significativo para el desarrollo de criptosistemas asimétricos en plataformas reducidas, sin sacrificar la seguridad criptográfica.

Referencias

[1] Magliveras S.S., Stinson D.R., van Trung T.: New approaches to designing public key cryptosystems using one-way functions and trapdoors in finite groups, Technical Report CORR, 2000-2049 (2000)

[2] Shpilrain V., Zapata G.: Combinatorial group theory and public-key cryptography, Preprint arXiv/math.gr, 0410068 (2004)

[3] Shor P.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM J. Comput., 5, 1484-1509 (1997)

[4] Menezes A.J., van Oorschot P.C., Vanstone S.A.: Handbook of Applied Cryptography. CRC Press (1997)

[5] Shikata J. et al, Optimizing the Menezes-Okamoto-Vanstone (MOV) Algorithm for Non-Supersingular Elliptic Curves, K. Y. Lam, E. Okamoto and C. Xing (Eds.): ASIACRYPT'99, LNCS 1716, pp. 86–102, (1999)

[6] Barreto,P. et al, Introdução à criptografia pós-quântica, Minicursos do XIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais — SBSeg 2013, Cap.2 (2013).

[7] Gerritzen L. et al (Editors), Algebraic Methods in Cryptography, Contemporary Mathematics, AMS, Vol. 418, (2006)

[8] Diffie W., Hellman M.E: New directions in cryptography, IEEE Transactions on information theory, 22, 644-654, (1976).

[9] Hecht J.P., Un modelo compacto de criptografía asimétrica empleando anillos no conmutativos. V Congreso Iberoamericano de Seguridad informática CIBSI, Montevideo, (2009).

[10] Gentile, E. R. "Estructuras algebraicas I." (1977): 77.

[11] Wonenburger, M. J. "Anillos de division". *Gaceta Matemática*, *13* (1961): 131-136.

[12] Wedderburn, J. H. (1921). On division algebras. Transactions of the American Mathematical Society, 22(2), 129-135.

[13] Hamilton, W. R.: Lectures on Quaternions: Containing a Systematic Statement of a New Mathematical Method, Hodges and Smith, (1853).

[14] Salmerón Quiroz, Bernardino Bernardino, et al. "REPORTE DEL DESARROLLO TECNICO DE LA INVESTIGACION PROYECTO SIP 20082294 Proyecto" Autogeneración de equipo Educativo en Robótica, usando Modelado vía Cuaterniones (2009).

[15] Kuipers, J. B. Quaternions And Rotation Sequences: A Primer With Applications To Orbits, Aerospace And Virtual Reality Princeton University Press, Princeton, New Jersey, (1999).

[16] Sánchez-Peña R. S., Alonso R., Control de vehículos espaciales. RIAII 2.3: 6-24, (2005).

[17] Serrano E., Sirne R,, y La Mura G. Rotaciones, secuencia aeroespacial y cuaterniones Una revisión de las relaciones fundamentales. Ciencia y Tecnología 1.14 (2014).

[18] Torres del Castillo, G., La representación de rotaciones mediante cuaterniones. Miscelanea Matemática 29 (1999): 43-50.

[19] Kamlofsky J., Bergamini L. Cuaterniones en Visión Robótica. V Congreso de Matemática Aplicada, Computacional e Industrial MACI, Tandil, (2015).

[20] Sankar V., Selvakumar A., Analyse and implement of cryptography with high security using cuaternion. International Journal of Innovative Research in Information Security. 1: 2 (2014).

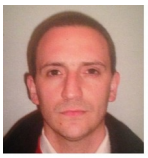
[21] Malekian E., Zakerolhosseini A, and Mashatan A. "QTRU: A Lattice Attack Resistant Version of NTRU PKCS Based on Quaternion Algebra". Preprint, Available from the Cryptology ePrint Archive: http://eprint.iacr. org/2009/386. Pdf (2009).

[22] Anand, PM Rubesh, Gaurav Bajpai, and Vidhyacharan Bhaskar. "Real-time symmetric crypto graphy using quaternion julia set". Int J Comp Sci Network Security 9.3 (2009): 20-26.

[23] Czaplewski, Bartosz, Mariusz Dzwonkowski, and Roman Rykaczewski. "Digital Fingerprinting Based on Quaternion Encryption Scheme for Gray-Tone Images." *Journal of Telecommunications & Information Technology* 2014.2 (2014).

Jorge Kamlofsky. Se graduó de Licenciado en Matemática en la Universidad Abierta Interamericana (UAI) y de Especialista en Criptografía y Seguridad Teleinformática en la Facultad de Ingeniería del Ejército (EST-IUE). Se encuentra finalizando la Maestría en Tecnología Informática (UAI) e iniciando un Doctorado en Ingeniería en la Universidad Nacional de Lomas de Zamora (UNLZ). Actualmente es profesor adjunto de Matemática Discreta y Física II y Auxiliar Docente de Seguridad Informática en la Facultad de Tecnología Informática de la UAI. También es investigador en el Centro de Altos Estudios en Tecnología Informática (CAETI), dependiente de la UAI.

Pedro Hecht (M'2012). Se graduó como Licenciado en Análisis de Sistemas (ESIO-DIGID), y Doctor de la Universidad de Buenos Aires (UBA). Actualmente es Profesor Titular de Criptografía I y II de la Maestría en Seguridad Informática dependiente de las Facultades de Cs. Económicas, Cs. Exactas y Naturales y de Ingeniería de la Universidad de Buenos Aires (UBA) e idéntico cargo en la Facultad de Ingeniería del Ejército (EST-IUE). Además es el Coordinador Académico de la citada Maestría (UBA), Profesor titular de Biofísica (UBA), Director de proyectos e investigador en modelos matemáticos de UBACyT y Director titular de EUDEBA. Es miembro de Criptored, IEEE Argentina, ACM SIGCSE, ACM SIGITE y otras. Área de interés: álgebra no conmutativa aplicada a la criptografía.



Oscar Hidalgo Izzi (L.51064). Estudiante de la Licenciatura en Matemática en la Universidad Abierta Interamericana (UAI). Actualmente elaborando el Trabajo final de grado.

Samira Abdel Masih. Se graduó de Licenciada en Matemática en la Facultad de Matemática, Astronomía y Física de la Universidad Nacional de Córdoba (UNC), y de Doctora en Ciencias Matemáticas en la Facultad de Ciencias Exactas y Naturales de la Universidad de Buenos Aires (UBA). Actualmente es Profesora Titular de Cálculo Infinitesimal II en la Facultad de Tecnología Informática de la Universidad Abierta Interamericana (UAI). También es investigadora en el Centro de Altos Estudios en Tecnología Informática (CAETI), dependiente de la UAI.