XXI Congreso de Ecuaciones Diferenciales y Aplicaciones
XI Congreso de Matemática Aplicada
Ciudad Real, 21-25 septiembre 2009
(pp. 1–8)

Un criptosistema de clave pública basado en códigos convolucionales

J.-J. Climent[1], V. Herranz[2], C. Perea[2] y V. Tomás[1]

[1] *Departament de Ciència de la Computació i Intel·ligència Artificial, Universitat d'Alacant, Ap. correus 99, E-03080 Alacant. E-mails:* jcliment@ua.es, vtomas@dccia.ua.es.
[2] *Centro de Investigación Operativa. Dpto. de Estadística, Matemáticas e Informática. Universidad Miguel Hernández, Avda. Universidad, s/n. E-03202 Elche. E-mails:* mavi.herranz@umh.es, perea@umh.es.

Palabras clave: Criptosistema, esquema de McEliece, código convolucional, representación entrada-estado-salida, codificación, decodificación.

Resumen

En este artículo presentamos un criptosistema de clave pública basado en el sistema criptográfico de McEliece en el que la matriz generadora del código bloque se obtiene de la representación entrada-estado-salida de un código convolucional óptimo.

1. Introducción

El propósito de los códigos correctores y el de los códigos criptográficos es diferente y en cierto sentido, opuesto. Podemos decir que el objetivo de los códigos correctores es hacer más clara la información transmitida, mientras que el objetivo de los códigos criptográficos es ocultar y hacer confusa dicha información. No obstante, existen puntos de encuentro entre ambas teorías y, en particular, los códigos correctores de errores permiten construir sistemas criptográficos de clave pública. Uno de los criptosistemas de clave pública basados en códigos correctores de errores es el criptosistema de McEliece [7], que consiste en utilizar un código fácilmente decodificable como clave privada, pero enmascarándolo para presentar como clave pública un código general, con el fin de enfrentar al criptoanalista con un problema computacionalmente imposible. Hasta el momento no se conocen ataques al criptosistema de McEliece, con ordenadores clásicos o cuánticos, que tengan orden subexponencial [5]. La seguridad de este esquema está basada en la NP-complejidad del problema de decodificación para códigos lineales generales [2, 6, 11].

En este trabajo, presentamos un criptosistema de clave pública basado en el esquema criptográfico de McEliece en el que obtenemos la matriz generadora del código bloque a

partir de la representación entrada-estado-salida de un código convolucional. Para ello, en la sección 2 introducimos los conceptos relativos a los códigos convolucionales así como, el algoritmo algebraico de decodificación que emplearemos en el sistema criptográfico que introducimos en la sección 3. Finalmente, ilustramos el criptosistema propuesto con un ejemplo más detallado en la sección 4.

2. Preliminares

Un **código convolucional** $\mathcal{C}$ de tasa k/n y complejidad δ puede ser descrito como

$$\left.\begin{array}{rcl} \boldsymbol{x}_{t+1} & = & A\boldsymbol{x}_t + B\boldsymbol{u}_t \\ \boldsymbol{y}_t & = & C\boldsymbol{x}_t + D\boldsymbol{u}_t \end{array}\right\}, \quad \boldsymbol{v}_t = \left[\begin{array}{c} \boldsymbol{y}_t \\ \boldsymbol{u}_t \end{array}\right], \; t = 0, 1, 2, \ldots; \quad \boldsymbol{x}_0 = \mathbf{0}, \tag{1}$$

donde A, B, C y D son matrices de tamaños $\delta \times \delta$, $\delta \times k$, $(n-k) \times \delta$ y $(n-k) \times k$ respectivamente, $\boldsymbol{x}_t$ es el **vector de estados**, $\boldsymbol{y}_t$ es el **vector de paridad**, $\boldsymbol{u}_t$ es el **vector de información** y $\boldsymbol{v}_t$ es el **vector código**. Tanto las entradas de las matrices como de los vectores son elementos de un cuerpo finito $\mathbb{F}$.

Decimos entonces que las matrices (A, B, C, D) son una representación *entrada-estado-salida* del código $\mathcal{C}$. Esta representación fue introducida por Rosenthal, Schumacher y York [9] y ha sido utilizada en los últimos años para analizar y construir códigos convolucionales [1, 3, 4, 8, 9, 12]. Para cada entero positivo j consideramos las matrices

$$\begin{aligned} \Phi_j(A, B) &= \left[\begin{array}{cccccc} A^{j-1}B & A^{j-2}B & \cdots & AB & B \end{array}\right], \\ \Omega_j(A, C) &= \left[\begin{array}{cccccc} C & CA & \cdots & CA^{j-2} & CA^{j-1} \end{array}\right]', \end{aligned}$$

donde $[.]'$ denota la matriz transpuesta de $[.]$, y para $j = 0, 1, 2, \ldots$, consideramos la matriz de **descripción local de trayectorias**

$$\mathcal{T}_j = \left[\begin{array}{ccccc} D & O & \cdots & O & O \\ CB & D & \cdots & O & O \\ CAB & CB & \cdots & O & O \\ \vdots & \vdots & & \vdots & \vdots \\ CA^{j-2}B & CA^{j-3}B & \cdots & D & O \\ CA^{j-1}B & CA^{j-2}B & \cdots & CB & D \end{array}\right].$$

Rosenthal [8] propone un algoritmo iterativo algebraico de decodificación que es capaz de decodificar códigos convolucionales que tienen una cierta estructura algebraica subyacente. Haremos uso de este algoritmo para completar la decodificación en el criptosistema de clave privada que proponemos.

La idea general es que, recibida una secuencia de longitud T, dado un entero positivo Θ y conocido un vector de estados $\boldsymbol{x}_\tau$ a partir del cual comenzar la decodificación, podemos calcular el vector de estados $\boldsymbol{x}_{\tau+T-\Theta}$ y los errores ocurridos entre los instantes τ y $\tau+T-\Theta$, recuperando así la secuencia original. Para ello, necesitamos que se cumplan las condiciones siguientes:

1. La matriz $\Phi_{T-\Theta+1}(A, B)$ debe tener rango completo δ, con lo que es la matriz de paridad del código bloque $\ker(\Phi_{T-\Theta+1}(A, B))$.

2. $d_1 = d(\ker(\Phi_{T-\Theta+1}(A,B))) > 3$, para que la capacidad correctora de errores del código $\ker(\Phi_{T-\Theta+1}(A,B))$ sea al menos 1.
3. La matriz $\Omega_\Theta(A,C)$ debe tener rango completo δ. Representará la matriz generadora de un código bloque.
4. $T > 2\Theta > 0$, para que podamos introducir al menos 1 error.
5. D no debe tener ninguna columna de ceros.

Si $\boldsymbol{e}$ es el vector de errores que pueden aparecer en una secuencia recibida de longitud T, debe ocurrir que $w(\boldsymbol{e}) \leq \lambda$, donde $\lambda = \min\left(\left\lfloor \frac{d_1-1}{2} \right\rfloor, \left\lfloor \frac{T}{2\Theta} \right\rfloor\right)$. Dada esta situación es posible calcular de forma única la secuencia transmitida $\{\boldsymbol{v}_t\}$.

A continuación desarrollamos el proceso de decodificación. Supongamos que el receptor ha recibido un mensaje $\widehat{\boldsymbol{v}}_0, \widehat{\boldsymbol{v}}_1, \ldots$ Tomamos una parte de la secuencia, $\widehat{\boldsymbol{v}}_\tau, \widehat{\boldsymbol{v}}_{\tau+1}, \ldots, \widehat{\boldsymbol{v}}_{\tau+T}$, y suponemos que anteriormente hemos realizado correctamente la decodificación de otros bloques que nos han permitido conocer $\boldsymbol{x}_\tau$. Si tomamos los últimos Θ bloques de la secuencia recibida debe cumplirse que

$$\begin{bmatrix} \widehat{\boldsymbol{y}}_{\tau+T-\Theta+1} \\ \widehat{\boldsymbol{y}}_{\tau+T-\Theta+2} \\ \vdots \\ \widehat{\boldsymbol{y}}_{\tau+T} \end{bmatrix} - \mathcal{T}_{\Theta-1} \begin{bmatrix} \widehat{\boldsymbol{u}}_{\tau+T-\Theta+1} \\ \widehat{\boldsymbol{u}}_{\tau+T-\Theta+2} \\ \vdots \\ \widehat{\boldsymbol{u}}_{\tau+T} \end{bmatrix} = \Omega_\Theta(A,C)\boldsymbol{x}_{\tau+T-\Theta+1}. \tag{2}$$

Utilizando el algoritmo de decodificación asociado al código bloque cuya matriz generadora es la matriz $\Omega_\Theta(A,C)$ recuperamos el estado $\boldsymbol{x}_{\tau+T-\Theta+1}$ y, una vez obtenido dicho vector, podemos calcular la secuencia de errores $\boldsymbol{e}_\tau, \boldsymbol{e}_{\tau+1}, \ldots, \boldsymbol{e}_{\tau+T-\Theta-1}$ utilizando el algoritmo de decodificación asociado al código bloque $\ker(\Phi_{T-\Theta+1}(A,B))$, de acuerdo con la siguiente ecuación

$$\Phi_{T-\Theta+1}(A,B) \begin{bmatrix} \widehat{\boldsymbol{u}}_\tau \\ \widehat{\boldsymbol{u}}_{\tau+1} \\ \vdots \\ \widehat{\boldsymbol{u}}_{\tau+T-\Theta} \end{bmatrix} - \boldsymbol{x}_{\tau+T-\Theta+1} + A^{T-\Theta+1}\boldsymbol{x}_\tau = \Phi_{T-\Theta+1}(A,B) \begin{bmatrix} \boldsymbol{e}_\tau \\ \boldsymbol{e}_{\tau+1} \\ \vdots \\ \boldsymbol{e}_{\tau+T-\Theta} \end{bmatrix} \tag{3}$$

y así recuperar la secuencia original, ya que $\boldsymbol{u} = \widehat{\boldsymbol{u}} - \boldsymbol{e}$. Recordemos que esta secuencia tendrá un peso menor o igual que λ. Como conocemos $\boldsymbol{x}_{\tau+T-\Theta+1}$ podemos utilizarlo como estado inicial para decodificar el siguiente bloque.

3. Criptosistema basado en el esquema de McEliece

Tomamos las matrices (A,B,C,D) de forma que el sistema dado por la expresión (1) representa un código convolucional óptimo y construimos la matriz de descripción local de trayectorias $\mathcal{T}_{T-\Theta}$. Si en un instante τ tenemos que $\boldsymbol{x}_\tau = \boldsymbol{0}$, entonces

$$\begin{bmatrix} \boldsymbol{y}_\tau \\ \boldsymbol{y}_{\tau+1} \\ \vdots \\ \boldsymbol{y}_{\tau+T-\Theta} \end{bmatrix} = \mathcal{T}_{T-\Theta} \begin{bmatrix} \boldsymbol{u}_\tau \\ \boldsymbol{u}_{\tau+1} \\ \vdots \\ \boldsymbol{u}_{\tau+T-\Theta} \end{bmatrix}. \tag{4}$$

Para poder utilizar la ecuación (4) cuando codificamos una secuencia, tenemos que asegurar que comenzamos en el estado $\boldsymbol{x}_\tau = \mathbf{0}$ en cada iteración. Es decir, que el vector código de la iteración anterior condujo el sistema al estado cero. Para ello necesitamos que nuestros vectores código sean vectores de peso finito.

Teorema 1 (Proposición 2.4 de [10]) $\left\{ \begin{bmatrix} \boldsymbol{y}_t \\ \boldsymbol{u}_t \end{bmatrix} \right\}_{t=0}^{\gamma}$ *representa un vector código de peso finito si y sólo si*

$$\left[\begin{array}{c|c} O & \Phi_{\gamma+1}(A,B) \\ \hline -I & \mathcal{T}_\gamma \end{array}\right] \left[\begin{array}{c} \boldsymbol{y}_0 \\ \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_\gamma \\ \hline \boldsymbol{u}_0 \\ \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_\gamma \end{array}\right] = \mathbf{0}.$$

Para conseguir esta condición transformaremos el mensaje original en palabras del código $\ker(\Phi_{T-\Theta+1}(A,B))$. Para ello, dividiremos nuestro mensaje en bloques de tamaño $k(T-\Theta+1)-\delta$ y multiplicaremos cada uno de estos bloques $\boldsymbol{m}_0$, $\boldsymbol{m}_1$, ... por la matriz F cuyas columnas constituyen una base de $\ker(\Phi_{T-\Theta+1}(A,B))$.

Además, necesitamos comprobar si se cumplen las condiciones necesarias 1, 2, 3, 4 y 5 que nos permiten aplicar el algoritmo de decodificación propuesto en [8]. Asímismo, deben cumplirse las condiciones sobre los errores. En nuestro caso, al suponer que no hay errores debidos al canal, sino que es el emisor el que los introduce, tenemos controlado el número de errores que se producen y, por tanto, podemos aplicar el algoritmo.

La clave privada del criptosistema viene dada por (P, Q, F, A, B, C, D), donde P es una matriz de permutación de tamaño $n(T-\Theta+1)$, F es la matriz de tamaño $k(T-\Theta+1)\times(k(T-\Theta+1)-\delta)$ cuyas columnas forman una base de $\ker(\Phi_{T-\Theta+1}(A,B))$ y Q es una matriz invertible de orden $k(T-\Theta+1)-\delta$. La clave pública del criptosistema es (E,λ), donde $E = P\begin{bmatrix} \mathcal{T}_{T-\Theta} \\ I_{k(T-\Theta+1)} \end{bmatrix} FQ$, es una matriz de tamaño $\alpha\times\beta$ con $\alpha = n(T-\Theta+1)$ y $\beta = k(T-\Theta+1)-\delta$.

3.1. Cifrado

La idea principal es que construimos un código convolucional y estructuramos el mensaje a enviar de manera que aplicamos el esquema de McEliece de forma similar al caso de un código bloque. Así, para cifrar un mensaje, el emisor lo divide en bloques de longitud β y en el instante i calcula $\mathfrak{v}_i = E\boldsymbol{m}_i$. Luego añade un vector de errores $\mathfrak{e}$ tal que $w(\mathfrak{e}) \leq \lambda$ y, por tanto, $\widetilde{\mathfrak{v}}_i = \mathfrak{v}_i + \mathfrak{e}$ es la palabra enviada.

3.2. Descifrado

El receptor recupera el mensaje siguiendo los mismos pasos que en el criptosistema de McEliece. El receptor recibe la palabra $\widetilde{\mathfrak{v}}_i$ y como la matriz de permutación P viene dada

en la clave privada, puede calcular

$$\widehat{\mathfrak{v}}_i = \begin{bmatrix} \widehat{\boldsymbol{v}}_{i(T-\Theta+1)} \\ \widehat{\boldsymbol{v}}_{i(T-\Theta+1)+1} \\ \vdots \\ \widehat{\boldsymbol{v}}_{i(T-\Theta+1)+T-\Theta-1} \\ \widehat{\boldsymbol{v}}_{i(T-\Theta+1)+T-\Theta} \end{bmatrix} = P^{-1}\widetilde{\mathfrak{v}}_i = P^{-1} \begin{bmatrix} \widetilde{\boldsymbol{v}}_{i(T-\Theta+1)} \\ \widetilde{\boldsymbol{v}}_{i(T-\Theta+1)+1} \\ \vdots \\ \widetilde{\boldsymbol{v}}_{i(T-\Theta+1)+T-\Theta-1} \\ \widetilde{\boldsymbol{v}}_{i(T-\Theta+1)+T-\Theta} \end{bmatrix} = \begin{bmatrix} \mathcal{T}_{T-\Theta} \\ I_{(T-\Theta+1)k} \end{bmatrix} FQ\boldsymbol{m}_i$$

$$+ P^{-1}\mathfrak{e} = \begin{bmatrix} \mathcal{T}_{T-\Theta} \\ I_{(T-\Theta+1)k} \end{bmatrix} \widetilde{\boldsymbol{u}} + P^{-1} \begin{bmatrix} \boldsymbol{e}_{i(T-\Theta+1)} \\ \boldsymbol{e}_{i(T-\Theta+1)+1} \\ \vdots \\ \boldsymbol{e}_{i(T-\Theta+1)+T-\Theta-1} \\ \boldsymbol{e}_{i(T-\Theta+1)+T-\Theta} \end{bmatrix} = \begin{bmatrix} \widehat{\boldsymbol{y}}_{i(T-\Theta+1)} \\ \widehat{\boldsymbol{y}}_{i(T-\Theta+1)+1} \\ \vdots \\ \widehat{\boldsymbol{y}}_{i(T-\Theta+1)+T-\Theta-1} \\ \widehat{\boldsymbol{y}}_{i(T-\Theta+1)+T-\Theta} \\ \widehat{\boldsymbol{u}}_{i(T-\Theta+1)} \\ \widehat{\boldsymbol{u}}_{i(T-\Theta+1)+1} \\ \vdots \\ \widehat{\boldsymbol{u}}_{i(T-\Theta+1)+T-\Theta-1} \\ \widehat{\boldsymbol{u}}_{i(T-\Theta+1)+T-\Theta} \end{bmatrix}.$$

Ahora, utilizando el altoritmo de decodificación, puede calcular $\widetilde{\boldsymbol{u}} = FQ\boldsymbol{m}$ y, como también conoce las matrices F y Q, puede calcular $\boldsymbol{m}$.

Nótese que en nuestro caso particular no necesitamos haber decodificado de forma correcta ninguna secuencia anteriormente. El propósito de esa condición es conocer un estado inicial $\boldsymbol{x}_\tau$ a partir del cual comenzar a decodificar. Pero nosotros sabemos que siempre $\boldsymbol{x}_0 = \boldsymbol{0}$ y, por lo tanto, podemos empezar la decodificación desde el instante 0. Después de decodificar un bloque siempre podemos calcular los estados que se suceden a continuación y así comenzar la siguiente iteración desde cualquiera de ellos.

4. Ejemplo

En esta sección consideramos el cuerpo finito $\mathbb{F}_{11}$ de 11 elementos y, para evitar confusiones, representaremos el elemento 10 como X.

Tomemos el código convolucional representado por las matrices

$$(A,B,C,D) = \left(\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ X & X & X & X & X & X \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} X & 0 & 2 & 6 & 3 & 8 \\ 0 & 3 & 4 & 8 & 2 & 0 \\ 9 & 5 & 6 & 3 & 1 & 7 \\ 7 & 0 & 9 & X & 4 & 7 \\ 0 & 9 & 0 & 9 & 0 & 9 \\ 3 & 0 & 6 & 5 & 7 & 3 \\ 8 & 3 & 1 & 9 & X & 6 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \\ 3 \\ 5 \\ 1 \\ 9 \\ 4 \end{bmatrix} \right).$$

En este caso $\delta = 6$, $k = 1$ y $n - k = 7$. Los valores de los parámetros $T = 10$ y $\Theta = 3$ satisfacen las condiciones necesarias para la aplicación del algoritmo. Además $\lambda = 1$.

Tomaremos P una matriz de permutación de orden 64 (que por razones de espacio omitimos) y $Q = \begin{bmatrix} 8 & 1 \\ 9 & X \end{bmatrix}$. Como matriz F cuyas columnas forman una base de

$\ker(\Phi_8(A,B))$ consideramos, en este caso,

$$F = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ X & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}'.$$

Por lo tanto

$$\begin{aligned} E =& P \begin{bmatrix} \mathcal{T}_7 \\ I_8 \end{bmatrix} FQ \\ =& [0X412298609X588917785395140988 8X392X0988917989X6128917X6080384X6X \\ & 280587X658X1511X38719193358X17124X07XX3X1611X803904042X687419268]'. \end{aligned}$$

Para cifrar el mensaje $\boldsymbol{m} = 172348$ primero lo dividimos en bloques de tamaño $k(T - \Theta + 1) - \delta = 2$. Obtenemos los bloques $\boldsymbol{m}_0 = [1,7]'$, $\boldsymbol{m}_1 = [2,3]'$ y $\boldsymbol{m}_2 = [4,8]'$ y ciframos cada uno de ellos utilizando la matriz E.

$$\begin{aligned} \mathfrak{v}_0 =& [\boldsymbol{v}_0, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_7]' = E\boldsymbol{m}_0 \\ =& [3043372681267442081 42X6406124242922442728554 2060X84122X995941240]', \\ \mathfrak{v}_1 =& [\boldsymbol{v}_8, \boldsymbol{v}_9, \ldots, \boldsymbol{v}_{15}]' = E\boldsymbol{m}_1 \\ =& [6086634152413884052849180124848474488434 5XX84010958244977X782480]', \\ \mathfrak{v}_2 =& [\boldsymbol{v}_{16}, \boldsymbol{v}_{17}, \ldots, \boldsymbol{v}_{23}]' = E\boldsymbol{m}_2 \\ =& [5550696399645776647749906196707106883616 1X0765263X24515481070165]'. \end{aligned}$$

Ahora el emisor puede introducir un máximo de $\lambda = 1$ errores de forma aleatoria en cada uno de estos bloques. Así, las palabras enviadas después de añadir los errores (que hemos marcado en rojo) son

$$\begin{aligned} \widetilde{\mathfrak{v}}_0 =& [\widetilde{\boldsymbol{v}}_0, \widetilde{\boldsymbol{v}}_1, \ldots, \widetilde{\boldsymbol{v}}_7]' \\ =& [3043372681267442081 42X6406124242922442728558 2060X84122X995941240]', \\ \widetilde{\mathfrak{v}}_1 =& [\widetilde{\boldsymbol{v}}_8, \widetilde{\boldsymbol{v}}_9, \ldots, \widetilde{\boldsymbol{v}}_{15}]' \\ =& [6086634152413884052849180124848 144884345XX84010958244977X782480]', \\ \widetilde{\mathfrak{v}}_2 =& [\widetilde{\boldsymbol{v}}_{16}, \widetilde{\boldsymbol{v}}_{17}, \ldots, \widetilde{\boldsymbol{v}}_{23}]' \\ =& [5550696399645776647749906196707106883616 1X0765263X64515481070165]'. \end{aligned}$$

Para proceder al descifrado, el primer paso es multiplicar cada bloque recibido por la matriz P^{-1}. De este modo queda

$$\begin{aligned} \widehat{\mathfrak{v}}_0 =& [\widehat{\boldsymbol{v}}_0, \widehat{\boldsymbol{v}}_1, \ldots, \widehat{\boldsymbol{v}}_7]' = P^{-1}[\widetilde{\boldsymbol{v}}_0, \widetilde{\boldsymbol{v}}_1, \ldots, \widetilde{\boldsymbol{v}}_7]' \\ =& [426X27824410968869014429405X268303317150426X7109522222222444 8442]', \\ \widehat{\mathfrak{v}}_1 =& [\widehat{\boldsymbol{v}}_8, \widehat{\boldsymbol{v}}_9, \ldots, \widehat{\boldsymbol{v}}_{15}]' = P^{-1}[\widetilde{\boldsymbol{v}}_8, \widetilde{\boldsymbol{v}}_9, \ldots, \widetilde{\boldsymbol{v}}_{15}]' \\ =& [8419435488207155170288 4780X94156066232X084193207X444144448888884]', \\ \widehat{\mathfrak{v}}_2 =& [\widehat{\boldsymbol{v}}_{16}, \widehat{\boldsymbol{v}}_{17}, \ldots, \widehat{\boldsymbol{v}}_{23}]' = P^{-1}[\widetilde{\boldsymbol{v}}_{16}, \widetilde{\boldsymbol{v}}_{17}, \ldots, \widetilde{\boldsymbol{v}}_{23}]' \\ =& [61351940769501X9946085583509841560641715042359 60X666666617777776]'. \end{aligned}$$

Los errores se han trasladado ahora a las posiciones marcadas en rojo.

Analicemos las posibles situaciones:

- Supongamos que los errores han ocurrido en la parte de secuencia $\widehat{\boldsymbol{u}}$ o $\widehat{\boldsymbol{y}}$ que interviene en el cálculo del estado $\boldsymbol{x}_{\tau+T-\Theta+1}$ en la ecuación (2). Como el emisor nunca introducirá un número de errores superior al número que el código bloque asociado a la matriz $\Omega_\Theta(A,C)$ puede recuperar, siempre podremos recuperar correctamente el estado $\boldsymbol{x}_{\tau+T-\Theta+1}$.
- Supongamos que los errores se producen en la paridad asociada a la parte de secuencia $\widehat{\boldsymbol{u}}$ que aparece en la ecuación (3). Esa parte de la secuencia no influye en el proceso de decodificación y tampoco nos interesa su recuperación, puesto que sólo queremos recuperar el mensaje original y no su paridad.
- Supongamos que los errores se producen en la parte de secuencia $\widehat{\boldsymbol{u}}$ que aparece en la ecuación (3). Esto sí afectará al cálculo de los errores.

En nuestro ejemplo sólo el primer bloque recibido se engloba en este último caso. Por lo tanto, sólo procederemos a la decodificación de este tramo de secuencia. Según la ecuación (2),

$$\begin{bmatrix} \widehat{\boldsymbol{y}}_8 \\ \widehat{\boldsymbol{y}}_9 \\ \widehat{\boldsymbol{y}}_{10} \end{bmatrix} - \mathcal{T}_2 \begin{bmatrix} \widehat{\boldsymbol{u}}_8 \\ \widehat{\boldsymbol{u}}_9 \\ \widehat{\boldsymbol{u}}_{10} \end{bmatrix} = \Omega_3(A,C)\boldsymbol{x}_8,$$

es decir, $\mathbf{0} = \Omega_3(A,C)\boldsymbol{x}_8$, y el algoritmo de decodificación para el código cuya matriz generadora es $\Omega_3(A,C)$ recupera correctamente el estado $\boldsymbol{x}_8 = \mathbf{0}$.

Para obtener el vector de errores sustituimos en la ecuación (3) y obtenemos

$$\begin{aligned}\Phi_8(A,B)\begin{bmatrix} 2 & 4 & 4 & 4 & 8 & 4 & 4 & 2 \end{bmatrix}' - \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}' + A^8 \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}' \\ = \Phi_8(A,B)\begin{bmatrix} \boldsymbol{e}_0 & \boldsymbol{e}_1 & \boldsymbol{e}_2 & \boldsymbol{e}_3 & \boldsymbol{e}_4 & \boldsymbol{e}_5 & \boldsymbol{e}_6 & \boldsymbol{e}_7 \end{bmatrix}',\end{aligned}$$

es decir

$$\begin{bmatrix} 0 \\ 0 \\ 4 \\ 7 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & X & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & X & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & X & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & X & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & X & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & X & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{e}_0 \\ \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \\ \boldsymbol{e}_3 \\ \boldsymbol{e}_4 \\ \boldsymbol{e}_5 \\ \boldsymbol{e}_6 \\ \boldsymbol{e}_7 \end{bmatrix}.$$

Aplicando ahora la decodificación por síndrome del código cuya matriz de control es $\Phi_8(A,B)$, tenemos que $\boldsymbol{e} = \begin{bmatrix} 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \end{bmatrix}'$. Entonces

$$\begin{aligned}\widetilde{\boldsymbol{u}} = \widehat{\boldsymbol{u}} - \boldsymbol{e} &= \begin{bmatrix} 2 & 4 & 4 & 4 & 8 & 4 & 4 & 2 \end{bmatrix}' - \begin{bmatrix} 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \end{bmatrix}' \\ &= \begin{bmatrix} 2 & 4 & 4 & 4 & 4 & 4 & 4 & 2 \end{bmatrix}'.\end{aligned}$$

Pero $\widetilde{\boldsymbol{u}} = FQ\boldsymbol{m}_0$ y como la matriz FQ representa un cambio de base, podemos recuperar $\boldsymbol{m}_0 = [1,7]'$.

Utilizando ahora como estado inicial el estado $\boldsymbol{x}_8 = \mathbf{0}$ podemos aplicar de nuevo el mismo procedimiento y seguir recuperando el resto de la secuencia $\boldsymbol{m}_1 = [2,3]'$ y $\boldsymbol{m}_2 = [4,8]'$.

Agradecimientos

Este trabajo ha sido parcialmente subvencionado por distintos proyectos. En concreto, el trabajo de Joan-Josep Climent y Virtudes Tomás ha sido subvencionado por el proyecto MTM2008-06674-C02-01, mientras que el de Victoria Herranz y Carmen Perea ha sido subvencionado por el proyecto MTM2008-06674-C02-02; ambos del Ministerio de Ciencia e Innovación del Gobierno de España. Además, la investigación de Victoria Herranz y Carmen Perea también ha sido subvencionada por el proyecto SA029A08 de la Junta de Castilla y León. Finalmente, la investigación de Virtudes Tomás ha sido financiada con una ayuda del Vicerrectorado de Investigación, Desarrollo e Innovación de la Universitat d'Alacant destinada a la formación de doctores.

Referencias

[1] B. M. Allen. *Linear Systems Analysis and Decoding of Convolutional Codes*. Tesis Doctoral, Department of Mathematics, University of Notre Dame, Indiana, USA, junio 1999.

[2] A. Canteaut y F. Chabaud. A new algorithm for finding minimum-weight words in a linear code: application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. *IEEE Transactions on Information Theory*, **44(1)**: 367–378 (1998).

[3] J.-J. Climent, V. Herranz y C. Perea. A first approximation of concatenated convolutional codes from linear systems theory viewpoint. *Linear Algebra and its Applications*, **425**: 673–699 (2007).

[4] J.-J. Climent, V. Herranz y C. Perea. Linear system modelization of concatenated block and convolutional codes. *Linear Algebra and its Applications*, **429**: 1191–1212 (2008).

[5] D. Engelbert, R. Overbeck y A. Schmidt. A summary of McEliece-type cryptosystems and their security. Cryptology ePrint Archive, Report 2006/162, 2006. `http://eprint.iacr.org/`.

[6] H. Janwa y O. Moreno. McEliece public key cryptosystems using algebraic-geometric codes. *Designs, Codes and Cryptography*, **8**: 293–307 (1996).

[7] R. J. McEliece. A public-key cryptosystem based on algebraic coding theory. DNS Progress Report 42–44, Jet Propulsion Laboratory, 1978.

[8] J. Rosenthal. An algebraic decoding algorithm for convolutional codes. *Progress in Systems and Control Theory*, **25**: 343–360 (1999).

[9] J. Rosenthal, J. Schumacher y E. V. York. On behaviors and convolutional codes. *IEEE Transactions on Information Theory*, **42(6)**: 1881–1891 (1996).

[10] J. Rosenthal y E. V. York. BCH convolutional codes. *IEEE Transactions on Information Theory*, **45(6)**: 1833–1844 (1999).

[11] J. van Tilburg. On the McEliece public-key cryptosystem. En S. Goldwasser (editor), *Advances in Cryptology – CRYPTO'88*, volumen 403 de *Lecture Notes in Computer Science*, páginas 119–131. Springer-Verlag, Berlin, 1988.

[12] E. V. York. *Algebraic Description and Construction of Error Correcting Codes: A Linear Systems Point of View*. Tesis Doctoral, Department of Mathematics, University of Notre Dame, Indiana, USA, mayo 1997.