Morfismos, Vol. 20, No. 1, 2016, pp. 25–42

Autenticación mediante conocimiento nulo en base a ecuaciones cuadráticas *

José Luis Juan Herrera García Guillermo Morales Luna
Feliú Sagols Troncoso

Resumen

Presentamos una alternativa a los métodos actuales de autenticación con conocimiento nulo, basada ésta en el método aceite-vinagre no-equilibrado (*Unbalanced Oil and Vinegar*). Con esto, se tiene un método alterno a los actuales, basados en diversos problemas computacionalmente difíciles.

2010 Mathematics Subject Classification: 11T71.
Keywords and phrases: Ecuaciones cuadráticas de varias variables, anillo de polinomios, ideales, llave publica, llave privada, método UOV, conocimiento nulo.

1 Introducción

Proponemos un método de autenticación de conocimiento nulo, donde absolutamente no hay información adicional que se pueda fugar hacia cualquier verificador, debido a que éste, antes de presentar un desafío al probador, es decir, a quien se autentica, conoce ya la respuesta que debe recibir: el desafío consiste en construir esa respuesta precisamente. Utilizamos el conocido método aceite-vinagre no-equilibrado [1], como base para generar un conjunto de polinomios que plantean un sistema de ecuaciones que se resuelve eficientemente por el probador. Tal conjunto de polinomios es propiamente la llave pública del probador. La llave

*El artículo está basado en la tesis de maestría del primer autor. Se realizó bajo la co-dirección de Guillermo Morales Luna y de Feliú Sagols Troncoso, ambos profesores del Cinvestav en los Departamentos de Computación y Matemáticas respectivamente. La tesis se presentó en el Departamento de Computación del Cinvestav en 2015.

privada del mismo, es un conjunto de polinomios y una transformación afín, que le permiten resolver eficientemente el desafío que el verificador le impone. En el proceso de autenticación, el verificador pide la solución de un polinomio generado como combinación lineal aleatoria de la llave pública. Como el probador conoce la solución de todos los polinomios podrá resolver también el nuevo polinomio y superar el desafío. El proceso se repite tantas veces como sea necesario hasta satisfacer las demandas del verificador.

2 Trabajos previos

El tema que nos ocupa, está basado en lo intratable de resolver ecuaciones en varias variables cuadráticas. Sin embargo existen otros problemas de identificación por ejemplo Kernels Permutados (PK) [2], Síndrome de Decodificación binario (SD) [3] [4], Ecuaciones Lineales Restringidas (CLE) [5], Perceptrones Permutados (PP) [6] [7] y q-ary SD [8]. Estos esquemas, se basan en la dificultad de una instancia aleatoria de cada problema cuya versión de decisión asociada se sabe es NP-Completa.

En el año 2011, se publicó el artículo [9], donde se propone un esquema de identificación para llave pública, basado en el problema de resolver un sistema cuadrado de varias variables, empleando además el concepto de conocimiento nulo. Por sistema cuadrado los autores se refieren a un sistema polinomial de ecuaciones en varias variables donde cada polinomio es la suma de una forma cuadrática y una transformación lineal. La solución x a un sistema cuadrado $y = F(x)$ forma la llave privada (secreto) y el vector y es la llave pública. Luego los autores consideran la forma polar asociada $G(x, y)$ de $F(x)$: $G(x, y) = F(x+y)-F(x)-F(y)$ que es una función bilineal porque precisamente en cada componente de G se eliminaron los términos lineales provenientes de F quedando una forma cuadrática.

El usar sistemas cuadrados ofrece muchas ventajas porque se pueden implementar eficientemente, se pueden utilizar como funciones de un sentido (*one-way functions*) y se ha demostrado que resolver sistemas cuadrados multivariable es un problema NP-completo [10]. Ahora bien, esto de ninguna manera significa que resolver sistemas cuadrados en un conjunto (infinito) que generemos es un problema NP-completo, podríamos generar sistemas donde sólo los términos lineales tengan coeficientes distintos de cero. Resolver los sistemas en tal conjunto definitivamente no es un problema NP-completo. Para que esto ocurra, debe-

mos generar sistemas cuadrados aleatorios. Es decir, los coeficientes en el conjunto generado, vistos como variables aleatorias continuas deben ser independientes e identicamente distribuidas. Así, para establecer el protocolo de conocimiento nulo basado en este enfoque se hace lo siguiente:

1. Se genera un vector $F(x)$ de polinomios aleatorios cuyas variables corresponden a las entradas en x.
2. $x = r_0 + r_1$ (separar la llave secreta en dos partes)
3. $y = F(r_0 + r_1) = G(r_0, r_1) + F(r_0) + F(r_1)$
4. $r_0 = t_0 + t_1$ (separar r_0 en dos partes)
5. $F(r_0) = e_0 + e_1$ (separar adicionalmente a $F(r_0)$)
6. Y como consecuencia: $y = G(t_0, r_1) + e_0 + F(r_1) + G(t_1, r_1) + e_1$

Así, el protocolo de autenticación de conocimiento nulo se basa en el hecho de que es imposible conocer en tiempo polinomial la llave privada x si sólo se conocen dos elementos en el conjunto $\{(t_0, e_o), (t_1, e_1), r_1\}$. Los detalles aparecen en los Algoritmos 1 y 2.

En la breve descripción que haremos en este párrafo, se utilizan conceptos criptográficos básicos no definidos a propósito por ser irrelevantes en el resto del trabajo. Éstos pueden ser consultados en [11]. En la línea 4 del Algoritmo 1, *Comm* se refiere a una función compromiso (*commitment*), misma que se puede realizar con base en una función picadillo resistente a colisiones. Esta función compromiso es estadísticamente de encubrimiento, es decir, si las duplas (u, v) y (u', v') se eligen uniformemente, entonces $Comm(u, v)$ y $Comm(u', v')$ son estadísticamente indistinguibles. Es decir, el compromiso de u prácticamente no revela información de x aún para un verificador muy poderoso. Así, esta misma función de compromiso es computacionalmente obligada, ya que la probabilidad de que $Comm(u, v) = Comm(u', v')$ es despreciable.

En [12] se evoluciona la propuesta anterior [9] generalizando esta construcción para polinomios de grado d, i.e., se diseñan en este caso dos esquemas de conocimiento nulo de un conjunto de ecuaciones de polinomios de grado d. Uno de los esquemas es óptimo en el número de cálculos que se ejecutan y el otro esquema es óptimo en el número de bits que se envían.

Algoritmo 1 Algoritmo para identificación con prueba de conocimiento nulo

Entrada: $x, F(x)$

Salida: Aceptación o rechazo de un *probador*

// **Actividades del *probador***:

Aleatoriamente seleccionar r_0, t_0, e_0

// Generar compromisos:

Calcular: $r_1 \leftarrow x - r_0, t_1 \leftarrow r_0 - t_0, e_1 \leftarrow F(r_0) - e_0$

Con función compromiso calcular: $c_0 = Comm(r_1, G(t_0, r_1) + e_0)$, $c_1 = Comm(t_0, e_0), c_2 = Comm(t_1, e_1)$

Enviar c_0, c_1, c_2 al *verificador*

// **Actividades del *verificador***:

Seleccionar alguno de los 3 c_i que recibió e indica al *probador* cuál seleccionó

// **Actividades del *probador***: el *probador* responde al *verificador*, con los siguientes datos, según el compromiso c_i seleccionado:

if Seleccionó c_0 **then**

    Envía $\sigma = (r_0, t_1, e_1)$

else if Seleccionó c_1 **then**

    Envía $\sigma = (r_1, t_1, e_1)$

else

    Seleccionó c_2, envía $\sigma = (r_1, t_0, e_0)$

end if

Algoritmo 2 Algoritmo para identificación con prueba de conocimiento nulo (continuación del Algoritmo 1)

14: El verificador recibe σ
    // **Actividades del *verificador***: con base en la σ recibida y al c_i seleccionado:
15: **if** Había seleccionado c_0 **then**
16: **if** $c_1 == Comm(r_0 - t_1, F(r_0) - e_1)$ **and** $c_2 == Comm(t_1, e_1)$ **then**
17: OK
18: **end if**
19: **else if** Había seleccionado c_1 **then**
20: **if** $c_0 == Comm(r_1, y - F(r_1) - G(t_1, r_1) - e_1)$ **and** $c_2 == Comm(t_1, e_1)$ **then**
21: OK
22: **end if**
23: **else**
24: **if** $c_0 == Comm(r_1, G(t_0, r_1) + e_0)$ **and** $c_1 == Comm(t_0, e_0)$ **then**
25: OK
26: **end if**
27: **end if**
28: **if** OK **then return** Acepta al *probador* como uno válido
29: **elsereturn** Rechaza al *probador*.
30: **end if**

Los trabajos que se han mencionado [9] y [12] tienen los mismos objetivos que el trabajo que estamos presentando, tuvimos conocimiento de ellos gracias a uno de los evaluadores anónimos de nuestro artículo. Sin haber hecho un estudio comparativo profundo sobre las diferencias con respecto a nuestro método consideramos que podríamos llegar a tener desventajas aunque a la fecha no hemos podido detectar una forma efectiva de rompimiento. Nuestro enfoque es más simple desde el punto de vista computacional. En contraste, los métodos reportados en [9] y [12] son robustos porque los coeficientes de los sistemas polinomiales se eligen de manera aleatoria y esto, de acuerdo con [10], garantiza la intratabilidad del problema de que un intruso descubra la llave privada. Como veremos más adelante, nosotros pretendemos alcanzar cierta "aleatoriedad" simulada mediante la aplicación de transformaciones lineales aleatorias sobre los coeficientes, pero sólo tenemos evidencias empíricas de que éstas funcionan de manera apropiada.

3 Conceptos preliminares

3.1 Antecedentes básicos

Sea $\mathbb{K}$ un campo finito. Entonces $\mathbb{K}[\mathbf{X}] = \mathbb{K}[X_1, \ldots, X_n]$ es un anillo de polinomios en n variables. Un subconjunto $I \subset \mathbb{K}[\mathbf{X}]$ es un ideal de $\mathbb{K}[\mathbf{X}]$ si se cumple que:

1. $0 \in I$
2. Si $f, g \in I$ entonces $f + g \in I$
3. Si $f \in I$ y $h \in \mathbb{K}[\mathbf{X}]$, entonces $fh \in I$

Sea $F = \{f_1, \ldots, f_m\}$ un conjunto de polinomios en $\mathbb{K}[\mathbf{X}]$. Se define:

$$\langle F \rangle = \langle f_1, \ldots, f_m \rangle = \left\{ \sum_{i=1}^{m} h_i f_i | h_1, \ldots, h_m \in \mathbb{K}[\mathbf{X}] \right\} \tag{1}$$

y se dice que $\langle F \rangle$ es el ideal de $\mathbb{K}[\mathbf{X}]$ generado por $f_1, \ldots, f_m$. La variedad algebraica de I es el conjunto de puntos $\mathbf{x} = (x_1, \ldots, x_n)$ que hacen que todos los polinomios $f_1, \ldots, f_m$ se anulen, es decir, al evaluarse en $\mathbf{x}$ todos dan el valor cero. Sea $B = \{X_1^2 - X_1, \ldots, X_n^2 - X_n\}$. Claramente, los puntos en la variedad algebraica de $\langle B \rangle$ han de tener entradas 0 o 1 únicamente. Los ideales de la forma $\langle F \cup B \rangle$ son llamados *ideales* 0-1. Este será el tipo de ideales a usar en la presente propuesta.

3.2 Método de autenticación

Sea $F = \{f_1, \ldots, f_m\} \subset \mathbb{K}[\mathbf{X}]$ un conjunto de polinomios cuadrados en n variables, y sea $I = \langle F \cup B \rangle$ el ideal 0-1 que genera. En [10] se demuestra que en un sistema polinomial cuadrado con coeficientes elegidos de manera aleatoria, encontrar el valor de las n variables para obtener un punto en la variedad algebraica de I es un problema NP-Completo, y lo es también el problema de localizar $\mathbf{x} \in \mathbb{K}^n$ tal que $\mathbf{y} = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))$ para un vector $\mathbf{y} \in \mathbb{K}^m$ dado de antemano. Diremos que este último problema es de la *localización de una imagen inversa* para la lista de polinomios.

El juego de autenticación con conocimiento nulo que proponemos involucra a un *probador* y a un *verificador*. El probador selecciona un conjunto de polinomios en $\mathbb{K}[\mathbf{X}]$ que genera a un ideal 0-1 tal que le sea sencillo resolver el problema de la localización de imágenes inversas debido a que conoce la manera en que se construyó el conjunto de polinomios. El conjunto generador es su *llave pública* y el procedimiento de construcción es su *llave privada*. El protocolo de autenticación queda determinado de la siguiente manera:

1. El probador genera una serie de polinomios que forman un ideal 0-1 y éstos conforman la llave pública.

2. El verificador definirá un vector con los valores que desea cumpla cada uno de los polinomios de la llave pública (por ejemplo, si desea que los polinomios cumplan con la variedad algebraica, entonces el verificador generará un vector con m ceros donde m es el número de polinomios). Este vector se lo manda al probador, en calidad de *desafío*, quien ha de encontrar la solución $\mathbf{x}$ del problema de la localización de imágenes inversas.

3. El verificador selecciona ahora k polinomios de la llave pública para realizar una combinación lineal con ellos, incluyendo también los valores del vector que deben cumplir esos polinomios. Únicamente el polinomio generado es enviado al probador.

4. El probador auténtico, dado que conoce una forma eficiente de resolver el sistema de ecuaciones conformado por los polinomios públicos y el vector que recibió del verificador, encuentra la solución $\mathbf{x} \in \mathbb{K}^n$, y posteriormente, sustituye los valores necesarios en el polinomio que le envió el verificador, obteniendo así la evaluación de dicho polinomio en $\mathbf{x}$ y envía su valor al verificador.

Un *probador intruso* no podrá encontrar los valores de $\mathbf{x}$ eficientemente, sólo podrá *adivinar* el resultado con probabilidad 1/2. El valor encontra333do, se le envía al verificador.

5. El verificador compara el valor recibido, con el que él calculó, si el resultado no es igual, rechaza al probador, si el resultado es igual al que él tiene, repite r veces el ciclo (del paso 2 al paso 5).

Si bien, el problema de encontrar la variedad algebraica asociada al ideal generado por un sistema de polinomios es NP-completo, cuando los sistemas son generados por algún método propio tenemos que garantizar aleatoriedad para preservar la NP-completitud. Sin embargo, en la práctica esto último es difícil de satisfacer. Por esta razón, hay varios métodos efectivos para resolver tipos particulares de sistemas de ecuaciones polinomiales, [13, 14, 15, 16, 17, 18], los cuales se conocen como *ataques algebraicos*. Así, es muy importante garantizar que efectivamente los coeficientes del sistema empleado se han elegido de manera aleatoria o garantizar de alguna otra forma que el problema es en verdad NP-completo.

4 Ecuaciones cuadráticas de varias variables

Un sistema de ecuaciones polinomiales de varias variables se basa en lo siguiente:

Sea $n \in \mathbb{N}$ el número de variables, $m \in \mathbb{N}$ el número de polinomios y $d \in \mathbb{N}$ el grado del sistema planteado de ecuaciones. Las variables $X_1, \ldots, X_n$ han de evaluarse sobre un campo finito $\mathbb{K}$ y la variable X_0 por convención se toma con el valor $1 \in \mathbb{K}$. Para n y d dados, ν_n^d denotará al conjunto de sucesiones de $d+1$ términos, no-decrecientes con valores enteros entre 0 y n inclusive. Sea

$$u = (u_0, u_1, \ldots, u_d) \in \nu_n^d. \tag{2}$$

Para $1 \leq i \leq m$, sea

$$p_i(X_1, \ldots, X_n) := \sum_{u \in \nu_n^d} \gamma_{i,u} \prod_{j=1}^{d} X_{u_j} \in \mathbb{K}[\mathbf{X}] \tag{3}$$

y sea $P = \{p_1, \ldots, p_m\}$

Para $d = 2$, se tiene el caso de *ecuaciones cuadráticas de varias variables*. De (2) vemos que u será un vector perteneciente al conjunto

ν_n^d, donde cada vector u será una pareja, ordenada, de números (índices) entre 0 y n. Esto producirá tres casos:

1. Para $u = (0, 0)$, entonces el correspondiente monomio según la ecuación (3) es

$$\prod_{j=1}^{n} X_{u_j} = X_0 \cdot X_0 = 1 \cdot 1 = 1,$$

   dado que X_0, como se mencionó antes, se toma como 1. En este caso, se obtiene solamente la constante 1.

2. Si $u = (0, k)$, con $k \geq 1$, es decir el primer índice toma el valor de cero, entonces el correspondiente monomio según la ecuación (3) es

$$\prod_{j=1}^{n} X_{u_j} = X_0 \cdot X_k = 1 \cdot X_k = X_k,$$

   esto es, el monomio que se obtiene involucra sólo una variable.

3. Cuando $u = (i, j)$, con $1 \leq i \leq j \leq n$, entonces el correspondiente monomio según la ecuación (3) es de segundo orden:

$$\prod_{j=1}^{n} X_{u_j} = X_1 X_2.$$

Considerando los puntos anteriores se puede ver que en el caso de ecuaciones cuadráticas de varias variables tendremos la suma de términos cuadrados (producto de dos variables x), con términos lineales (una sola variable x) y una constante. Por otra parte, los coeficientes $\gamma_{i,u}$ de la ecuación (3) pueden ser distinguidos para cada caso de los mencionados antes; denotemos éstos γ, β y α. Entonces, los polinomios serán de la siguiente forma:

$$p_i(X_1, \ldots, X_n) := \sum_{1 \leq j \leq k \leq n} \gamma_{i,j,k} X_j X_k + \sum_{j=1}^{n} \beta_{i,j} X_j + \alpha_i \tag{4}$$

donde $1 \leq i \leq m$, $\gamma_{i,j,k}, \beta_{i,j}, \alpha_i \in \mathbb{K}$. Generalmente a $\gamma_{i,j,k}$ se le llama *coeficiente cuadrado*, a $\beta_{i,j}$ *coeficiente lineal* y a α_i *coeficiente constante*. En el caso de $\mathbb{K} = GF(2)$, es suficiente que cada variable aparezca con grado a lo más 1, ya que, para cada i, $X_i^2 = X_i$.

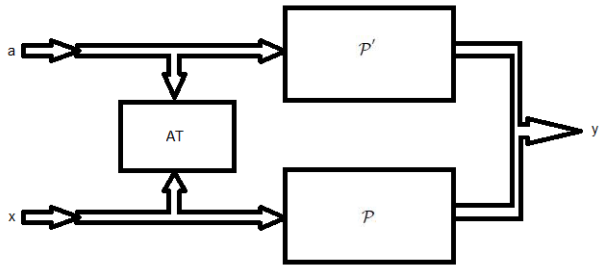

Figura 1: Método Aceite-Vinagre No-Equilibrado

5 Método Aceite-Vinagre No-Equilibrado

El método presentado en este documento, está basado en el uso de ecuaciones cuadráticas de varias variables. En [19] se encuentra un análisis detallado de este tipo de ecuaciones y su dificultad para resolverlas. Emplearemos en particular el método conocido como *Aceite-Vinagre No-Equilibrado* (AVNE) (*Unbalanced Oil and Vinegar*) descrito en [1]. La Fig. 1 muestra el esquema general de este método.

- El conjunto de variables se dividirá en dos tipos de variables, las *vinagre* y las *aceite*.
- Sean $n, v, o \in \mathbb{N}$, respectivamente, el número de variables en las ecuaciones cuadráticas, el número de variables vinagre y el número de variables aceite: $n = v + o$.
- Sea $m \in \mathbb{N}$ el número de ecuaciones que se tendrán en cada sistema de ecuaciones. Por definición de este método, $m = o$, por lo que $v = n - m$.
- Sea P un conjunto de m polinomios cuadrados en n variables, se denominará $\mathbf{X} = (X_1, \ldots, X_n)$.
- Sea P' un conjunto de m polinomios cuadrados en $n = v + o$ variables. Se nombrará a estas n variables como

  $$\mathbf{A} = (A_1, \ldots, A_v, A_{v+1}, \ldots, A_n)$$

  donde las variables vinagre son $A_1, \ldots, A_v$ y las variables aceite son $A_{v+1}, \ldots, A_n$.

- Para cada $\mathbf{x} \in \mathbb{K}^n$, sea $\mathbf{y} = (p(\mathbf{x}))_{p \in P}$ la lista de valores que toman los m polinomios en P, evaluados en $\mathbf{x}$. Se dirá que P es *invertible* si conociendo $\mathbf{y}$, se puede determinar $\mathbf{x}$. Lo mismo se puede decir para P' y en ese caso invertir P' será encontrar los valores de las n variables $\mathbf{A}$ cuando se conocen los valores que toman las m ecuaciones que forman P'.

Como n es mayor que m y se trata de un sistema de polinomios cuadrados, un sistema de ecuaciones $\{p(\mathbf{X}) = y |\ p \in P, \mathbf{y} \in \mathbb{K}^m\}$ no es invertible fácilmente. Por otra parte, el sistema de ecuaciones P' será fácilmente invertible por la distinción que se hace de las $n = v + o$ variables y la forma como se trata a éstas.

Para explicar porqué P' es fácilmente invertible, se explicará primero el bloque AT que es una transformación afín.

Se forma una matriz de orden $n \times n$, no singular, M y un vector $\mathbf{w}$ de n elementos, aleatoriamente, los que conformarán el *secreto* que se debe guardar. La transformación afín se define como:

$$\forall \mathbf{x} \in \mathbb{K}^n : \quad S(\mathbf{x}) = M \cdot \mathbf{x} + \mathbf{w} \tag{5}$$

Asimismo, dado que la matriz M es invertible, se puede encontrar $\mathbf{x}$ en cada ecuación $\mathbf{y} = S(\mathbf{X})$:

$$\mathbf{x} = S^{-1}(\mathbf{y}) = M^{-1}(S(\mathbf{x}) - \mathbf{w}) = M^{-1}(\mathbf{y} - \mathbf{w}). \tag{6}$$

Por otra parte, los polinomios que forman P' se generarán de la siguiente manera:

$$p_i(A_1, \ldots, A_n) = \sum_{j=1}^{n-m} \sum_{k=1}^{n} \gamma_{i,j,k} A_j A_k + \sum_{j=1}^{n} \beta_{i,j} A_j + \alpha_i \tag{7}$$

Por esta construcción, se puede observar, que las variables A_j de la primera sumatoria doble, son todas vinagre, en tanto que las variables A_k son cualesquiera, es decir las variables vinagre son las únicas que podrán aparecer multiplicadas por otras vinagre, mientras que las variables aceite **NO** aparecerán multiplicadas por otra aceite. De esto se concluye que los polinomios en P', son cuadrados sólo con las variables vinagre.

Debido a esta última observación, el esquema AVNE se construye de la siguiente manera:

- Se genera aleatoriamente un conjunto de m polinomios en n variables de P', cumpliendo con las características antes mencionadas.

- Se genera la matriz M y el vector $\mathbf{w}$ también aleatoriamente pero asegurando que la matriz M sea no singular y por lo tanto invertible.

- Se generan ahora los m polinomios en n variables de P. Para esto, se obtienen las variables a_i en función de las variables X_i, usando la transformación afín, ver ecuación (5). Esto es, cada variable a_i es igual a la suma de los productos de los elementos de cada renglón de la matriz M por las variables X_i: $a_i = \sum_{j=1}^{n} M_{i,j} X_j$. Como un ejemplo y suponiendo que $n = 4$ y que el primer renglón de M es $1, 0, 1, 1$ entonces $a_1 = X_1 + X_3 + X_4$.

  Posteriormente, estas variables a_i (que en este momento están representadas en función de las variables X_i), se sustituyen en el sistema de ecuaciones de P' y después de realizar su reducción simbólica, se obtiene la salida $\mathbf{y}$ (ver Fig. 1). Como esta $\mathbf{y}$ debe ser igual a la salida que entregue el sistema de ecuaciones P entonces estas m ecuaciones formarán P.

Con este esquema, si se tienen los valores de $\mathbf{x} = X_1, \ldots, X_n$, se sustituyen dichos valores en P y se obtiene la salida $\mathbf{y}$ (observar que $\mathbf{y}$ tiene un tamaño de $m = o$ valores). Alternativamente, los valores $\mathbf{x}$ se pasan por la transformación afín y se genera el vector $\mathbf{a} = a_1, \ldots, a_n$, mismo que se sustituye en los polinomios P' obteniéndose la misma salida $\mathbf{y}$.

Por otra parte, y esto es lo que permite invertir P' fácilmente, si se proporciona la salida $\mathbf{y}$ buscando encontrar las $\mathbf{x}$ que generen dicha salida, entonces en el sistema de ecuaciones P', se sustituyen valores aleatorios en las variables vinagre y quedan entonces m ecuaciones *lineales* con $o = m$ incógnitas (ya que todas las variables vinagre se sustituyen por los valores numéricos generados aleatoriamente y por la forma en que se generó P' las variables aceite nunca aparecen multiplicadas por ellas mismas). Se resuelve el sistema de ecuaciones que se generó y con esto se conocerán las $n = v + o$ variables que permitirán usar la transformación inversa afín, para generar las variables $\mathbf{x}$ que entreguen el valor solicitado de $\mathbf{y}$.

En todo este proceso es muy importante precisar el papel de la matriz M y el vector $\mathbf{w}$. Su objetivo es transformar la estructura del sistema cuadrado que en principio es muy simple de resolver (conjunto P') en un conjunto que pretende tener coeficientes aleatorios (conjunto P). Como ya se ha mencionado en varios puntos de este trabajo, lo que garantizaría

la intratabilidad de localizar la llave privada para un intruso es que los coeficientes en los polinomios de P sean realmente aleatorios. Esto se logra en la medida que la aleatoriedad de M y $\mathbf{w}$ genere instancias difíciles para localizar la variedad algebráica asociada al ideal generado por P. Así, la efectividad del método sólo se ha podido establecer de manera empírica. Algunos experimentos realizados se detallan en las conclusiones del trabajo.

6 Conocimiento nulo

Aplicando el principio de los protocolos de conocimiento nulo, en este caso, se busca que el probador demuestre que puede resolver el sistema de ecuaciones formado por los polinomios de la llave pública igualados al vector generado por el verificador, sin que el probador revele nada adicional a lo que el verificador ya conoce.

En esta propuesta, los m polinomios en n variables que forman la llave pública, se resuelven fácilmente por el probador, dado que éstos se construyeron por el método aceite-vinagre no-equilibrado, como se mencionó antes y cuando el verificador le envía un nuevo polinomio que se generó por una combinación lineal de dichos polinomios, entonces el probador puede sustituir los valores que tiene de $\mathbf{x}$ para encontrar el valor al que evalúa el polinomio enviado. Este valor será igual al que tiene el verificador y por lo tanto éste aceptará este resultado.

Las tres propiedades que debe satisfacer cualquier prueba de conocimiento nulo, se satisfacen por la presente propuesta [20]:

1. Completitud (completeness): Dado que un probador auténtico, sabe como resolver el sistema de ecuaciones planteado por un verificador también auténtico, entonces, cuando éste último envía un nuevo polinomio producto de la combinación lineal de algunos polinomios originales, el probador podrá encontrar el valor al que es igual dicho polinomio, con solo sustituir $\mathbf{x}$ en el polinomio enviado y con 100% de probabilidad puede enviar al verificador un resultado correcto.

2. Coherencia (soundness): Si el verificador es auténtico, pero el probador no lo es, entonces este último, no puede resolver eficientemente el nuevo polinomio enviado por el verificador y considerando que estamos trabajando con ideales 0-1 se tiene 1/2 de probabilidad de enviar la respuesta correcta. Si se realizan r ciclos

de este protocolo, la probabilidad disminuye a $1/2^r$ y por lo tanto el probador no podrá burlar consistentemente al verificador.

3. Conocimiento nulo: El probador nunca envía al verificador, los valores calculados de $\mathbf{x}$, sólo envía el valor al que debe evaluar el polinomio generado por el verificador y cuyo resultado ya es conocido por éste, con lo que el probador nunca revela algún conocimiento nuevo al verificador.

7 Robustez del protocolo de autenticación por conocimiento nulo

Dado que el esquema de autenticación de conocimiento nulo perfecto utiliza como base al método AVNE, dependemos primeramente en este método, para que este protocolo de autenticación sea robusto. El método AVNE, hasta donde tenemos conocimiento el día de hoy, no ha sido roto mientras el número de variables vinagre sea el doble de las de aceite: $v = 2o$. En [21] se detalla el ataque que sufrió AVNE cuando $v = o$ y en [1] se analiza cuando

$$v \geq \frac{o^2}{2}$$

dando como resultado también un esquema inseguro. Entonces, sólo hay que apegarse a usar $v = 2o$, lo que permanece como seguro hasta ahora.

El número total de casos que tenemos para realizar las combinaciones lineales (NC) es:

$$NC = \binom{m}{1} + \binom{m}{2} + \cdots + \binom{m}{m} = \sum_{k=1}^{m} \binom{m}{k} \tag{8}$$

Esto porque de los m polinomios que forman la llave pública, podemos escoger uno de m, o bien dos de m y así sucesivamente hasta poder escoger m de los m polinomios.

La Ecuación 8, se puede reducir a:

$$NC = 2^m - 1 \approx 2^m \text{ para } m \text{ grandes.}$$

Proponemos como m grande a $m \geq 16$, ya que con eso la diferencia entre 2^m y $2^m - 1$ es $\leq 1.5 \times 10^{-3}\%$.

Por otra parte, para nuestro caso de estudio, $v = 2o$ y $m = o$, por lo que $n = o + v = 3o = 3m$.

Entonces, un parámetro de seguridad de 128 bits implica tener un sistema de 128 polinomios y para eso, $n = 3 \cdot 128 = 384$, i.e., debemos generar una llave pública de 128 polinomios en 384 variables. Observamos que este parámetro de seguridad hace que el número de variables de la llave pública se mueva linealmente.

8 Conclusión

Dado que la criptografía de llave pública, se basa fundamentalmente en la dificultad de resolver la factorización de números enteros muy grandes (RSA) o bien la dificultad de resolver el problema del logaritmo discreto (ElGamal, curvas elípticas) este trabajo, presenta una alternativa a dichos esquemas: el uso de sistema de ecuaciones de *polinomios* para encontrar los valores de las variables, que produzcan un resultado deseado. La razón por la que el esquema vinagre-aceite equilibrado se ha roto es precisamente porque hay fuertes correlaciones en los coeficientes producidos por el modelo. Para el esquema no-equilibrado no existe una prueba de que el esquema no se pueda romper, pero tampoco se ha logrado hacer de manera consistente. Pruebas experimentales que hemos realizado sugieren que los coeficientes del sistema P producidos por el esquema al variar aleatoriamente la matriz M y el vector w minimizan sus correlaciones cruzadas cuando el número de variables aceite es igual a un tercio del total de variables y lo maximizan en un medio.

Por otra parte, la propuesta del protocolo de conocimiento nulo de este documento, muestra que el verificador nunca aprende nada más de lo que él ya conoce, dado que él conoce la respuesta que espera del probador incluso antes que envíe el polinomio que debe resolver este último.

Es importante considerar que el sistema empleado en este trabajo, requiere por razones de seguridad, que el número de variables aceite sea aproximadamente un tercio del total de variables. Así, si el total de variables es 384, se tendrán 128 ecuaciones en 384 variables, pero además, habría 2^{128} posibles valores que el *verificador* podría pedir al *probador* para autenticarse. Si suponemos que en cada sesión para autenticarse, se repite el ciclo 50 veces (probabilidad de acierto $1/2^{50} = 8.8 \times 10^{-16}$ para un probador no auténtico) , esto nos generaría $2^{128}/50 = 6.8 \times 10^{36}$ posibles sesiones de autenticación. Si cada nanosegundo se realizara una

sesión nueva de autenticación (representando esto una serie de autenticaciones poco realista por la rapidéz con la que se estarían realizando) y el *verificador* llevara un control de que valores a pedido cumplan los polinomios de la llave pública entonces con los valores anteriores, éstos se agotarían en poco más de 215.8×10^{18} años, momento en que habría que cambiar de dicha llave pública!

José Luis Juan Herrera-García
Departamento de Computación,
CINVESTAV-IPN,
Av. I.P.N. 2508
México, D.F.
jherrera@computacion.cs.cinvestav.mx

Guillermo Morales-Luna
Departamento de Computación,
CINVESTAV-IPN,
Av. I.P.N. 2508
México, D.F.
gmorales@cs.cinvestav.mx

Feliú Sagols Troncoso
Departamento de Matemáticas,
CINVESTAV-IPN,
Av. I.P.N. 2508
México, D.F.
fsagols@math.cinvestav.mx

Referencias

[1] A. Kipnis, J. Patarin, and L. Goubin, "Unbalanced oil and vinegar signature schemes," *EUROCRYPT99*, 1999.

[2] A. Shamir, "An Efficient Identification Scheme Based on Permuted Kernels (Extended Abstract)," in *CRYPTO 1989. LNCS*, vol. 435, pp. 606–609, Springer, Heidelberg, 1990.

[3] J. Stern, "A New Identification Scheme Based on Syndrome Decoding," in *CRYPTO 1993. LNCS*, vol. 773, pp. 13–21, Springer, Heidelberg, 1994.

[4] J. Stern, "A New Paradigm for Public Key Identification," in *IEEE Transactions on Information Theory*, pp. 13–21, 1996.

[5] J. Stern, "Designing Identification Schemes with Keys of Short Size," in *CRYPTO 1994. LNCS*, vol. 839, pp. 164–173, Springer, Heidelberg, 1994.

[6] D. Pointcheval, "A New Identification Scheme Based on the Perceptrons Problem," in *EUROCRYPT 1995. LNCS*, vol. 950, pp. 319–328, Springer–Verlag, Heidelberg, 1995.

[7] D. Pointcheval and G. Poupard, "A New NP-Complete Problem and Public-key Identification," in *Des. Codes Cryptography 28*, pp. 5–31, 2003.

[8] P. Cayrel, P. Véron, and S. E. Y. Alaoui, "A Zero-Knowledge Identification Scheme Based on the q-ary Syndrome Decoding Problem," in *SAC 2010. LNCS*, vol. 6544, pp. 171–186, Springer, Heidelberg, 2011.

[9] K. Sakumoto, T. Shirai, and H. Hiwatari, "Public-Key Identification Schemes Based on Multivariate Quadratic Polynomials," in *Advances in Cryptology - CRYPTO 2011* (P. Rogaway, ed.), vol. 6841 of *Lecture Notes in Computer Science*, pp. 706–723, Springer Berlin Heidelberg, 2011.

[10] G. Bard, *Algorithms for linear and polynomial systems of equations over finite fields with applications to cryptanalysis.* PhD thesis, Faculty of the Graduate School of the University of Maryland.

[11] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography.* CRC press, 2011.

[12] V. Nachef, J. Patarin, and E. Volte, "Zero-Knowledge for Multivariate Polynomials," in *Progress in Cryptology - LATINCRYPT 2012* (A. Hevia and G. Neven, eds.), vol. 7533 of *Lecture Notes in Computer Science*, pp. 194–213, Springer Berlin Heidelberg, 2012.

[13] N. Courtois, A. Klimov, J. Patarin, and A. Shamir, "Efficient algorithms for solving overdefined systems of multivariate polynomial equations.," *In Bart Preneel, editor, Advances in Cryptology-EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques. Bruges, Belgium, May 14-18, 2000, Proceeding, volume 1807 of Lecture Notes in Computer Science, pages 392-407. Springer, 2000.*, 2000.

[14] N. Courtois and J. Pieprzyk, "Cryptanalysis of block ciphers with overdefined systems of equations.," *Cryptology e-print archive, report 2002/044, 2002. http:// e-print.iacr.org/2002/044.*, 2002.

[15] X. S. Gao and Z. Huang., "Efficient characteristic set algorithms for equation solving in finite fields and application in analysis

of stream ciphers.," *Cryptology ePrint Archive, Report 2009/637, 2009. http://eprint.iacr.org/2009/637.*, 2009.

[16] J.-C. Faugere., "A new efficient algorithm for computing grobner bases without reduction to zero (f4).," *Journal of Pure and Applied Algebra, 139:61-88.*, 2002.

[17] H. Raddum and I. Semaev., "New technique for solving sparse equation systems.," *Cryptology ePrint Archive, Report 2006/475, 2006. http://eprint.iacr.org/2006/475.*, 2006.

[18] X. Tang and Y. Feng., "A new efficient algorithm for solving systems of multivariate polynomial equations.," *Cryptology ePrint Archive, Report 2005/312, 2005. http://eprint.iacr.org/2005/312.*, 2005.

[19] C. Wolf and B. Preneel, "Taxonomy of public key schemes based on the problem of multivariate quadratic equations," *Cryptology ePrint Archive, Report 2005/077,http://eprint.iacr.org/*, 2005.

[20] O. Goldreich, *Foundations of Cryptography. Basic Tools.* Cambridge, UK: Cambridge University Press, 1st ed., 2004.

[21] A. Kipnis and A. Shamir, "Cryptanalysis of the oil and vinegar signature scheme.," *CRYPTO98, LNCS 1462, pp. 257-267.*, 1998.