Privacidad en Motores de Búsqueda con un Protocolo Multi-usuario con Atacantes Internos

Cristina Romero-Tris, Jordi Castellà-Roca, Alexandre Viejo
Universitat Rovira i Virgili, UNESCO Chair in Data Privacy
Departament d'Enginyeria Informàtica i Matemàtiques
Av. Països Catalans 26, E-43007 Tarragona, Spain
Email de contacto: cristina.romero@urv.cat

***Abstract*—Los motores de búsqueda son herramientas que permiten encontrar información en Internet. Sin embargo, pueden también representar una amenaza para la privacidad de sus usuarios, ya que almacenan y analizan la información personal que los usuarios revelan al hacer sus consultas. Para evitar este problema de seguridad, es necesario proporcionar mecanismos que protejan a los usuarios de los motores de búsqueda.**

Este artículo propone un protocolo que protege la privacidad del usuario no sólo frente al motor de búsqueda, sino también frente a usuarios internos deshonestos. El esquema presentado mejora el rendimiento de propuestas similares en términos de carga computacional y comunicación.

***Index Terms*—privacidad, motores de búsqueda,**

I. Introducción

La búsqueda de términos es una actividad frecuente para la mayoría de usuarios de Internet. Los motores de búsqueda son herramientas que permiten recuperar la información que el usuario necesita de entre toda la información que se encuentra en la red. Existen distintos motores de búsqueda en el mercado, como por ejemplo Google, Bing, Yahoo, etc.

Durante el proceso, el motor de búsqueda construye un perfil del usuario basado en las consultas que envía. Por ejemplo, en su Centro de Privacidad [1], Google informa de que sus servidores registran automáticamente las consultas que hacen sus usuarios. Estos "registros" incluyen no sólo la consulta del usuario, sino también su dirección IP, el tipo de navegador, el lenguaje del navegador, la fecha y la hora de la consulta y una referencia a una o más cookies capaces de identificar unequívocamente el navegador del usuario.

La creación de perfiles es una amenaza para la privacidad de los usuarios de los motores de búsqueda. Los registros almacenados por los motores de búsqueda contienen datos sensibles que pueden ser usados para revelar información de un individuo concreto y comprometer su derecho a la privacidad.

Algunos incidentes en el pasado han demostrado que los motores de búsqueda no son capaces de proteger la privacidad de sus usuarios. Por ejemplo, en 2006 AOL publicó un archivo que contenía veinte millones de búsquedas generadas por sus usuarios [2]. Este incidente tuvo serias consecuencias, ya que muchas consultas contenían información que permitía identificar a los usuarios.

Incidentes como este ponen de manifiesto que los usuarios no deberían confiar ciegamente en las empresas que manejan los motores de búsqueda. Por lo tanto, es necesario proporcionar alternativas que impidan a los motores de búsqueda conocer la información privada de los usuarios.

II. Trabajo previo

El problema de conseguir hacer búsquedas privadas en Internet es una cuestión ya estudiada en la literatura. Una de las aproximaciones existentes consiste en utilizar un protocolo muli-usuario. En este tipo de protocolos, inicialmente se forma un grupo de usuarios donde un usuario le pide a otro componente del grupo que envíe su consulta al motor de búsqueda en su lugar y le devuelva el resultado.

En [3], los autores proponen un protocolo multi-usuario llamado Useless User Profile (UUP). La ideA básica del sistema es que hay un nodo central que agrupa a los usuarios en grupos dinámicos donde intercambian sus consultas de forma segura. Al final, cada usuario envía una consulta que pertenece a otro miembreo del grupo, no la suya propia. De esta manera, el perfil que el motor de búsqueda posee de cada usuario está distorsionado. El tiempo de respuesta que este protocolo consigue es de 5.2 segundos. Este tiempo mejora significativamente propuestas previas. Sin embargo, el protocolo UUP sufre un inconveniente: no proporciona seguridad en presencia de usuarios internos deshonestos. Esto significa que un atacante interno puede conocer qué consulta pertenece a qué miembro del grupo.

Los autores de [4] utilizan un escenario similar al propuesto en [3]. Sin embargo, defienden que el nivel de seguridad de [3] no es suficiente. Por lo tanto, modifican el protocolo UUP para hacerlo seguro frente a algunos ataques. Sin embargo, el inconveniente de su propuesta es que utiliza herramientas criptográficas costosas (*i.e.* doble cifrado) que hacen que el tiempo de espera no sea aceptable. De hecho, los autores indican que la ejecución de su protocolo es el doble de costosa que el protocolo propuesto en [3].

A. Contribution and Plan of this Paper

En este artículo, presentamos un nuevo protocolo multi usuario que protege la privacidad de los usuarios frente a los motores de búsqueda y frente a atacantes internos. Respecto a sistemas existentes similares, nuestro protocolo aumenta el nivel de seguridad de [3], y requiere menos recursos de computación y comunicación que [4].

La sección III presenta los conceptos previos y las herramientas que utiliza el protocolo. La sección IV describe el escenario y los requisitos de privacidad. El protocolo se encuentra detallado en la sección V. Las secciones VI y VII analizan la privacidad y el rendimiento del protocolo respectivamente. Finalmente, la sección VIII contiene las conclusiones del artículo.

III. Conceptos previos y Notación

A. n-out-of-n Threshold ElGamal Encryption

En la mayoría de protocolos multi-usuario, es necesario que algunas de las operaciones criptográficas se realicen conjuntamente entre varios usuarios. En el cifrado *n-out-of-n threshold ElGamal*, n usuarios tienen una clave pública distribuida y, mientras que su correspondiente clave secreta α se divide en n partes α_i, con la particularidad de que ningún usuario aislado conoce el secreto completo. Usando este protocolo, un cierto mensaje m puede cifrarse mediante la clave pública y, mientras que el cifrado solo puede realizarse si los n usuarios colaboran. Para más detalles sobre la generación de claves, el cifrado y el descrifrado, ver [5].

B. ElGamal Re-masking

La operación llamada re-masking consiste en realizar ciertas modificaciones sobre un mensaje cifrado de manera que el texto del mensaje no cambie, pero la apariencia del mensaje cifrado sea distinta.

Dado un mensaje cifrado bajo ElGamal $E_y(m, r)$, el re-masking se realiza calculando [6]:

$$E_y(m, r) \cdot E_y(1, r')$$

Para un $r' \in \mathbb{Z}_q^*$ escogido aleatoriamente y donde $\cdot$ representa el producto escalar por componentes (Un mensaje cifrado con ElGamal puede verse como un vector de dos componentes). El texto cifrado resultante corresponde al mismo texto en claro m.

C. Redes Benes Optimizadas de tamaño arbitrari -Optimized Arbitrary Size (OAS) Benes-

Una red de permutación Benes [7] es un grafo dirigido con N entradas y N salidas. La notación utilizada es $PN^{(N)}$. Esta red es capaz de realizar cualquier posible permutación de N elementos.

Una red Benes se compone de un conjunto de conmutadores 2x2. Estos conmutadores tienen una señal de control binaria $b \in \{0, 1\}$ que determina su estado interno y, en consecuencia, la salida. Los dos posibles estados de un conmutador 2x2 están representados en la Figura 1. La fórmula para calcular el número mínimo de conmutadores de una red es:

$$S(N) = \begin{cases} (N-1) + 2 * S(\frac{N}{2}) & if\ N\ is\ even \\ 2 * \lfloor \frac{N}{2} \rfloor + S(\lceil \frac{N}{2} \rceil) + S(\lfloor \frac{N}{2} \rfloor) & if\ N\ is\ odd \end{cases}$$

$$Where\ S(1) = 0,\ S(2) = 1,\ S(3) = 3$$

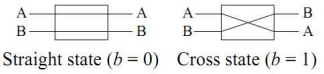

Fig. 1. Estados de un conmutador 2 x 2

Para más detalles sobre la construcción de las redes Benes, consultar [8].

1) OAS Benes multi usuario.: Las OAS Benes se pueden utilizar para realizar permutaciones conjuntas. Esto significa que los conmutadores de la OAS Benes se pueden distribuir entre un grupo de n usuarios que quieran realizar una permutación de N entradas. Sin embargo, esto se debe hacer de tal manera que ningún usuario aislado conozca la permutación total entre las entradas y las salidas.

De acuerdo con [8], una permutación segura (donde no hay ningún usuario que conozca la permutación total) requiere como mínimo t OAS Benes $PN^{(N)}$, donde t depende del número mínimo de usuarios honestos que el sistema requiere. Los t OAS Benes $PN^{(N)}$ están divididos equitativamente en n etapas adyacentes. De esta manera, la etapa i (para $i \in 1, \ldots, n$) se asigna al usuario i. Dado que la construcción de la OAS Benes es mecánica, los usuarios pueden crearla sin ningún tipo de colaboración entre ellos, y sin la ayuda de otra entidad.

Con el fin de obtener una permutación segura, la condición que debe cumplirse es que los usuarios honestos controlen, al menos, $S(N)$ conmutadores. Se denota como λ el número mínimo de usuarios honestos que el sistema requiere. Proponemos la siguiente fórmula para calcular el número de OAS Benes que requiere un esquema de n usuarios, N entradas, y λ usuarios honestos.

$$t = \left\lceil \frac{n \cdot \left\lceil \frac{S(N)}{\lambda} \right\rceil}{S(N)} \right\rceil$$

D. Prueba de Equivalencia de Textos en Claro -Plaintext Equivalence Proof (PEP)-

PEP [9] es una prueba de conocimiento nulo (ZKP) basada en una variante del algoritmo de Schnorr [10]. El propósito de esta prueba es demostrar que dos textos cifrados diferentes contienen el mismo mensaje en claro .

Dos textos cifrados con ElGamal $(c1_a, c2_a) = (g^{r_a}, m_a \cdot y^{r_a})$ and $(c1_b, c2_b) = (g^{r_b}, m_b \cdot y^{r_b})$ para algún r_a, $r_b \in \mathbb{Z}_q^*$ son equivalentes si $m_a = m_b$. Para conocer los detalles del argoritmo del protocolo PEP, ver [9].

E. PEP Disyuntivo -Disjunctive PEP (DISPEP)-

DISPEP [9] es una extensión del protocolo PEP. En este caso, un usuario demuestra que dos textos cifrados diferentes contienen el mismo texto en claro.

Sean $(c1_a, c2_a) = (g^{r_a}, m_a \cdot y^{r_a})$ y $(c1_b, c2_b) = (g^{r_b}, m_b \cdot y^{r_b})$ dos mensajes cifrados bajo ElGamal. Entonces, uno de ellos es una versión (tras una operación de re-masking) de otro mensaje cifrado $(c1, c2) = (g^r, m \cdot y^r)$ para algún r_a, r_b, $r \in \mathbb{Z}_q^*$ si $m_a = m$ o $m_b = m$. Para conocer los detalles del argoritmo del protocolo DISPEP, ver [9].

IV. Modelo del Sistema

A. Entities

El protocolo se ejecuta en un escenario con tres entidades:

- *Usuarios*. Los individuos que envían las consultas al motor de búsqueda. Suponemos que en nuestro escenario hay usuarios honestos y deshonestos. La motivación de los usuarios honestos es proteger su privacidad. La motivación de los usuarios deshonestos es conocer las consultas de los usuarios honestos.
- *Nodo central*. Es la entidad que organiza a los usuarios en grupos. Su objetivo principal es distribuir la información que los usuarios necesitan para ponerse en contacto con los demás miembros del grupo.
- *Motor de búsqueda*. Es el servidor que contiene la base de datos. Como se mencionó anteriormente, los motores de búsqueda no tienen ninguna motivación para proteger la privacidad de sus usuarios.

B. Requisitos de privacidad

Con el fin de garantizar la privacidad de los usuarios, el sistema debe cumplir los siguientes requisitos:

- Los usuarios no pueden vincular ninguna consulta con el usuario que la generó.
- El nodo central no puede vincular ninguna consulta con el usuario que la generó.
- El motor de búsqueda no es capaz de construir un perfil fiable de ningún usuario.

V. Descripción Protocolo

El protocolo está compuesto por cuatro fases que los usuarios ejecutan secuencialmente.

A. Configuración del Grupo

Cada usuario que desee enviar una consulta al motor de búsqueda, debe ponerse en contacto con el nodo central. Cuando el nodo central recibe n peticiones, se crea un grupo $\{U_1, \ldots, U_n\}$. Entonces, a los n usuarios se les notifica que pertenecen al mismo grupo. Los usuarios reciben un mensaje con el tamaño del grupo (n) y la posición que a cada componente ha sido asignada al azar ($i = 1, \ldots, n$). Cada posición se asocia con la dirección IP y el puerto donde el usuario está escuchando. Esta información permite a los usuarios establecer un canal de comunicación entre ellos. A partir de ese momento, el nodo central ya no es necesario.

B. Distribución de la Red de Permutación

Como se indica en la sección III-C1, t redes OAS Benes son necesarias para realizar una permutación seguro. El número de entradas de las redes es igual al número de usuarios $N = n$, que es también el mismo que el número de consultas. En cuanto al número de usuarios honestos, el parámetro se fija siempre en $\lambda = 2$. La razón de esta elección requiere un análisis de la privacidad y, por tanto, más adelante se detalla en la Sección VII-A2.

Conociendo los parámetros n, N y λ, los usuarios calculan el valor de t usando la fórmula definida en la Sección III-C1.

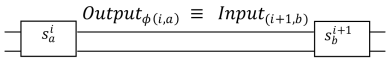

Fig. 2. Correlación entre las salidas de un conmutador y las entradas del siguiente

La construcción de las t OAS Benes $PN^{(n)}$ es mecánica. Esto significa que los usuarios no necesitan intercambiar información. Mientras conozcan el valor de los parámetros n y t, conocen la distribución de los conmutadores en las t redes OAS Benes $PN^{(n)}$. Por lo tanto, pueden dividírsela equitativamente en n fases adyacentes.

De acuerdo con las posiciones asignadas en la fase anterior, el usuario U_i es responsable de los conmutadores que corresponden a la etapa i-ésima. Cada etapa está formada por d conmutadores, donde $d = \frac{t}{n} \cdot S(n)$, en promedio.

Denotamos s_l^i el l-ésimo conmutador del i-ésimo usuario para $i = 1, \ldots, n$ and $l = 1, \ldots, d$. También definimos una función $\Phi(i, l)$ que, dada una salida de un conmutador, devuelve la entrada del siguiente conmutador a la que está conectada. El resultado se obtiene de acuerdo a la disposición de los conmutadores en la red de permutación. La Figura 2 ilustra el funcionamiento de esta función.

C. Generación de la Clave de Grupo.

1) Los usuarios $\{U_1, \ldots, U_n\}$ acuerdan un gran número primo p, donde $p = 2q + 1$ y q es un primo también. A continuación, eligen un elemento $g \in \mathbb{Z}_q^*$ de orden q.
2) Para generar la clave de grupo, cada usuario U_i realiza los siguientes pasos:
   a) Genera un número aleatorio $a_i \in \mathbb{Z}_q^*$.
   b) Calcula su parte de la clave pública $y_i = g^{a_i} \bmod p$.
   c) Hace un broadcast de un compromiso al valor que ha calculado $h_i = \mathcal{H}(y_i)$, donde $\mathcal{H}$ es una función de un solo sentido.
   d) Envía y_i a los demás miembros del grupo.
   e) Comprueba que $h_j = \mathcal{H}(y_j)$ para $j = (1, \ldots, n)$.
   f) Calcula la clave de grupo a partir de los valores recibidos:
   $y = \prod_{1 \le j \le n} y_j = g^{a_1} \cdot g^{a_2} \cdot \ldots \cdot g^{a_n}$ End enumerate End enumerate

D. Recuperación anónima de las consultas

Para $i = 1, \ldots, n$, cada usuario U_i realiza las operaciones siguientes:

1) U_i genera un valor aleatorio r_i y emplea la clave de grupo y para cifrar su consulta m_i:
$$E_y(m_i, r_i) = (c1_i, c2_i) = c_i^0$$
2) U_i envia c_i^0 a los otros miembros U_j, para $\forall j \neq i$.
3) Para cada conmutador s_l^i ($l = (1, \ldots, d)$) con dos entradas c_{i-1}^{2l-1} y c_{i-1}^{2l} recibidas de U_{i-1} (las entradas para los conmutadores de U_1 son las consultas cifradas $\{c_1^0, \ldots, c_n^0\}$):
   a) U_i aplica una operación de re-masking sobre los mensajes cifrados c_{i-1}^{2l-1} y c_{i-1}^{2l}. Obtiene una nueva

versión e_{i-1}^{2l-1} y e_{i-1}^{2l} usando el algoritmo de re-masking descrito en la sección III-B.

b) U_i escoge aleatoriamente $b_{i,l} \in \{0,1\}$ para determinar el estado del conmutador s_l^i como en la Figura 1. De acuerdo a este estado, obtiene el conjunto de mensajes cifrados en diferente orden $e_{i-1}^{\pi(2l-1)}$ y $e_{i-1}^{\pi(2l)}$.

c) U_i hace un broadcast de $\{c_{\Phi(i,2l-1)}, c_{\Phi(i,2l)}\} = \{e_{i-1}^{\pi(2l-1)}, e_{i-1}^{\pi(2l)}\}$

d) Considerando que:

$$c_{i-1}^{2l-1} = E_y(m_1, r_1), \quad c_{i-1}^{2l} = E_y(m_2, r_2)$$

$$e_{i-1}^{\pi(2l-1)} = E_y(m_1', r_1'), \quad e_{i-1}^{\pi(2l)} = E_y(m_2', r_2')$$

U_i debe probar que $e_{i-1}^{\pi(2l-1)}$ y $e_{i-1}^{\pi(2l)}$ son una nueva versión (tras varias operaciones de re-masking y permutaciones) de c_{i-1}^{2l-1} y c_{i-1}^{2l}. Esto equivale a probar las dos siguientes igualdades: T

I. $(m_2 = m_2') \vee (m_2 = m_1')$.
Para probarlo, es necesario utilizar el protocolo DISPEP descrito en la Sección III-E.

II. $m_1 \cdot m_2 = m_1' \cdot m_2'$.
U_i calcula $c = E_y(m_1 \cdot m_2, r_1 + r_2)$ y $c' = E_y(m_1' \cdot m_2', r_1' + r_2')$, y utiliza el protocolo PEP (Sección III-D) para probar que c y c' contienen el mismo texto en claro.

El resto de usuarios U_j ($\forall j \neq i$) verifican las pruebas.

4) Asignamos la notación $\{c_1, \ldots, c_n\}$ a lista de mensajes cifrados resultado de las operaciones de re-masking y permutación. En este punto del protocolo, cada usuario U_i posee estos n valores. Entonces, el usuario U_i descifra el valor c_i que corresponde a una consulta m^i generada por algún miembro del grupo. Es necesario destacar que debido a las operaciones de re-masking y permutaciones, probablemente m^i no corresponde a m_i (la consulta generada por U_i).
El descifrado de una consulta c_i requiere que los n usuarios participen enviando su parte de la clave secreta al usuario U_i. De esta manera, U_i recibe $(c1_i)^{\alpha_j}$ de U_j, para $j = (1, \ldots, n)$ y $j \neq i$. También U_i calcula $(c1_i)^{\alpha_i}$. Finalmente, U_i descifra m^i calculando:

$$m^i = \frac{c2_i}{c1_i^{\alpha_i}(\prod_{j \neq i} c1_i^{\alpha_j})}$$

VI. Análisis de la Privacidad

En esta sección se analiza el comportamiento del protocolo de acuerdo a los requisitos de privacidad descritos en la sección IV-B. Básicamente, estos se requiere que, al final del protocolo, ninguna consulta puede relacionarse con el usuario que la ha generado.

El sistema se analiza en presencia de las tres entidades deshonestas que pueden participar en el protocolo: usuarios deshonestos, nodo central deshonesto y motor de búsqueda deshonesto.

A. Usuario Deshonesto

El uso del criptosistema ElGamal en el protocolo garantiza que un usuario deshonesto no puede saber si dos textos cifrados diferentes darán lugar al mismo texto en claro tras el descrifrado.

Por lo tanto, cada vez que una consulta cifrada c_i atraviesa un conmutador, se aplica un re-masking y una permutación, y el atacante sólo puede intentar vincular el resultado con c_i al azar, con una probabilidad de éxito de $1/2$. Esta probabilidad disminuye exponencialmente por cada conmutador que cruza la consulta cifrada.

En el caso de un atacante que sólo conoce las entradas y las salidas de la red de permutación, las operaciones de re-masking y permutación le impiden encontrar la relación directa entre ellos. Por lo tanto, dado un usuario en particular, la probabilidad de que vincule correctamente una consulta con una de las entradas de la red de permutación es de $1/n$.

B. Nodo Central Deshonesto

El nodo central es el que crea los grupos de usuarios. Esta entidad sólo participa en la fase inicial del protocolo, antes de que los usuarios intercambien cualquier mensaje. Puesto que no conoce las posteriores comunicaciones entre los usuarios, el nodo central no es capaz de vincular ninguna consulta al usuario que la originó

C. Motor de Búsqueda Deshonesto

El objetivo del motor de búsqueda es recoger las consultas de los usuarios para construir sus perfiles. En el protocolo propuesto, el motor de búsqueda sólo participa en la última fase. El motor de búsqueda recibe las consultas de cada miembro del grupo y devuelve los resultados.

El motor de búsqueda puede vincular cada consulta con el usuario que la envió e incluir esa información en su perfil. Dado que un usuario U_i no enviasu consulta, sino la consulta de otro participante, su perfil se distorsiona. Por lo tanto, tras varias ejecuciones del protocolo, el perfil de U_i que el motor de búsqueda posee es inútil.

VII. Análisis del rendimiento

El objetivo de esta sección es analizar el rendimiento de nuestra propuesta y comparar los resultados con otras propuestas similares. Concretamente, nuestra propuesta se compara con dos sistemas similares: el esquema propuesto por [3] y el esquema que se presenta en [4]. Dado que el trabajo presentado en [4] no contiene simulaciones ni una estimación en un entorno real del tiempo que tarda un usuario en recibir los resultados para su consulta (retardo de la consulta), hemos decidido analizar los protocolos de forma teórica. Con este objetivo, a continuación se analiza el protocolo respecto al tiempo de cálculo requerido y al número de mensajes que necesitan ser intercambiados.

A. Selección de parámetros

Hay tres parámetros del sistema que se deben definir: el tamaño del grupo (n), la longitud de la clave, y el número de OAS Benes (t).

1) Tamaño del grupo y longitud de la clave.: En el protocolo propuesto, la privacidad de los usuarios frente al motor de búsqueda aumenta con el tamaño del grupo. Esto significa que cuantos más usuarios por grupo, más privacidad obtienen los miembros.

Sin embargo, en la práctica, el tamao del grupo está limitado por el tiempo que los usuarios deben esperar para crear el grupo. Para minimizar el retardo de la consulta, la creación del grupo debe ser rápida. De acuerdo con [3], Google responde a 1157 consultas por segundo. Las consultas siguen una distribución de Poisson. Esto permite calcular la probabilidad de formar un grupo de n usuarios en un cierto tiempo. Después de varias pruebas con $n = 3$, $n = 4$, $n = 5$ y $n = 10$, los autores de [3] concluyen que $n = 3$ es el tamaño de grupo más realista. Como se indica en [3], la probabilidad de formar un grupo de $n = 3$ usuarios en una centésima de segundo es casi 1.

Por esta razón, en el análisis de rendimiento siguiente se presentan los resultados obtenidos para $n = 3$ usuarios. Para una comparación más completa, también se muestran los resultados para un tamaño de grupo de $n = 4$ y $n = 5$ usuarios.

En cuanto a la longitud de la clave, de acuerdo con [3] y [11], una longitud de clave de 1024 bits se considera computacionalmente segura.

2) Número mínimo de redes OAS Benes.: El número mínimo de redes OAS Benes (t) se calcula según la fórmula definida en la sección III-C1. Esta fórmula depende del tamaño del grupo (n), el número de entradas (N) y el número mínimo de usuarios honestos (λ).

La selección del tamao del grupo (n) se ha explicado en la sección anterior. El número de entradas es igual al tamao del grupo ($N = n$), debido a que las entradas son las consultas que cada usuario genera. No obstante, el número mínimo de usuarios honestos requiere un análisis adicional.

Nuestro esquema debe ser capaz de proporcionar privacidad en las peores condiciones posibles. Es decir, cuando el número de usuarios deshonestos es grande en comparación con el número de usuarios honestos. Sin embargo, cuanto menor sea el parámetro λ, más redes OAS Benes son necesarias y crece el retardo de la consulta. Por lo tanto, el valor de λ debe minimizar el retardo de la consulta sin sacrificar la privacidad de los usuarios.

El valor mínimo para el número de usuarios honestos es de $\lambda = 1$. Sin embargo, este valor no garantiza la privacidad de los usuarios. Como se indica en la sección VI-B, en un escenario con un único usuario honesto y $n - 1$ usuarios deshonestos, incluso si la permutación es segura, la privacidad de los usuarios honestos se pierde. Una coalición de $n - 1$ usuarios deshonestos puedan identificar fácilmente cuál de las n consultas pertenece al usuario honesto.

El siguiente valor mínimo posible es $\lambda = 2$. Es el peor escenario en el que nuestro esquema puede proporcionar privacidad. En este caso, los $n-2$ usuarios deshonestos tienen una probabilidad de 0.5 de conocer la consulta que pertenece a cada usuario honesto.

En resumen, podemos fijar el parámetro $\lambda = 2$, como el número mínimo de usuarios honestos que nuestro protocolo requiere.

TABLE I
TIEMPO MEDIO REQUERIDO POR USUARIO PARA REALIZAR LAS EXPONENCIACIONES MODULARES

Castellà et al. [3]	$(3n + 3) \cdot \tau_{1024}$
Lindell et al. [4]	$6n \cdot \tau_{1024} + 5n \cdot \tau_{2048} - \tau_{1024} + 2 \cdot \tau_{2048}$
Nuestro Esquema	$\left(n + 3 + \frac{25 \cdot t \cdot S(n)}{n}\right) \cdot \tau_{1024}$

B. Análisis del tiempo de cálculo

A continuación, se analiza el tiempo de cálculo necesario en la esquema de [3], [4] y nuestra propuesta. Más en concreto, nos centramos en la cantidad de exponenciaciones modulares que cada usuario debe realizar en cada ejecución del protocolo.

Hay algunas partes del protocolo de [4] que emplean un doble cifrado. Esto significa que algunas exponenciaciones modulares se llevan a cabo utilizando un módulo 2048 bits, en lugar de utilizar un módulo de 1024 bits como hacen [3] y nuestra propuesta. A fin de comparar el tiempo requerido por una exponenciación de 1024 bits y una de 2048 bits, ejecutamos una simulación que realiza ambas operaciones. La simulación reveló que, en las mismas condiciones, una exponenciación modular de 1024 bits tarda 22 ms en promedio, mientras que una exponenciación modular de 2048 bits requiere 172 ms en promedio.

La tabla I muestra el tiempo de cálculo teórico necesario para las exponenciaciones modulares en cada protocolo. La τ_{1024} representa el tiempo necesario para hacer una exponenciación modular de 1024 bits. La τ_{2048} representa el tiempo necesario para hacer una exponenciación modular de 2048 bits
.

La Figura 3 muestra los tiempos para un tamaño de grupo de 3, 4 y 5 usuarios. Los resultados indican que [3] obtiene el menor tiempo de cálculo. Esto sucede porque [3] no utiliza ningún mecanismo para proteger a los participantes frente a usuarios deshonestos. Puesto que [4] utiliza cifrados dobles y nuestra propuesta utiliza pruebas de conocimiento nulo, los tiempos de cálculo son más altos. Sin embargo, los resultados indican que nuestra propuesta mejora el protocolo de [4]. Por ejemplo, para $n = 3$ usuarios, nuestra propuesta requiere aproximadamente un segundo más de tiempo de cálculo que [3], mientras que [4] necesita 3 segundos más que [3].

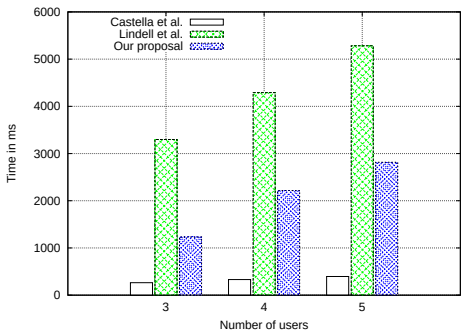

Fig. 3. Comparación de las exponenciaciones modulares por usuario

TABLE II
NÚMERO MEDIO DE MENSAJES ENVIADOS POR CADA USUARIO

Castellà et al. [3]	$3n - 1 - \frac{2}{n}$
Lindell et al. [4]	$4n - 2 - \frac{2}{n}$
Nuestro esquema	$4n - 4$

C. Análisis del número de mensajes

Con el fin de analizar el rendimiento del protocolo, otra medida es el uso de la red. La tabla VII-C refleja el número de mensajes que cada usuario envía en cada ejecución del protocolo.

La Figura 4 representa el número de mensajes enviados cuando 3, 4 o 5 usuarios ejecutan de forma conjunta el protocolo. Aunque el número de mensajes es similar en las tres propuestas, los resultados indican que el número de mensajes enviados en [3] es menor que en [4] y en nuestra propuesta. Los resultados también indican que nuestra propuesta requiere menos el envío de mensajes que [4].

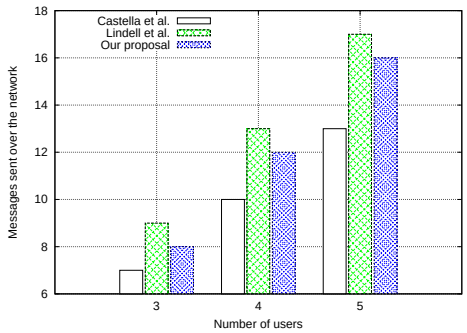

Fig. 4. Comparación de mensajes enviados por usuario en cada protocolo

VIII. CONCLUSIONES

Los usuarios frecuentemente revelan información personal en las consultas que envían a los motores de búsqueda. Los motores de búsqueda almacenan esta información y la utilizan para mejorar los resultados de las búsquedas y para proporcionar publicidad personalizada a cada usuario. Este artículo propone un protocolo que protege la privacidad de los usuarios frente a la creación de perfiles por parte de los motores de búsqueda.

Se ha realizado un análisis de privacidad y otro de rendimiento al protocolo propuesto. El análisis de privacidad muestra que los usuarios esán protegidos frente al motor de búsqueda y frente a atacantes internos. Respecto al rendimiento, el protocolo mejora las propuestas similares que proporcionan el mismo nivel de privacidad.

AGRADECIMIENTOS

Los autores están con la UNESCO Chair in Data Privacy, pero son únicamente responsables por lo expresado en este artículo, que no reflejan necesariamente la posición de la UNESCO ni comprometen a la organización. Este trabajo ha sido en parte financiado por el Ministerio Español de Ciencia e Innovación a través de los proyectos TSI2007-65406-C03-01 “E-AEGIS”, CONSOLIDER CSD2007-00004 “ARES” y PT-430000-2010-31 “Audit Transparency Voting Process”, y por el Gobierno de Cataluña bajo el 2009 SGR 1135.

REFERENCES

[1] Google Privacy Center, 2011. http://www.google.com/privacy
[2] M. Barbaro, T. Zeller, “A Face is Exposed for AOL Searcher No. 4417749”, *New York Times*, August 2006.
[3] J. Castellà-Roca, A. Viejo, J. Herrera-Joancomartí, “Preserving user’s privacy in web search engines”, *Computer Communications* vol. 32, no. 13-14, pp. 1541–1551, 2009.
[4] Y, Lindell, E. Waisbard, “Private web search with malicious adversaries”, *Proceedings of the 10th international conference on Privacy enhancing technologies – PETS’10*, pp. 220–235, 2010.
[5] Y. Desmedt, Y. Frankel, “Threshold cryptosystems”, *Advances in Cryptology – CRYPTO’89*, Lecture Notes in Computer Science, vol. 335, pp. 307–315, 1990.
[6] M. Abe, “Mix-networks on permutation networks”, *Advances in Cryptology – Asiacrypt’99*, Lecture Notes in Computer Science, vol. 1716, pp. 258-273, 1999.
[7] A. Waksman, “A Permutation Network”, *Journal of the ACM*, vol. 15, no. 1, pp. 159–163, 1968.
[8] W. H. Soo, A. Samsudin, A. Goh, “Efficient Mental Card Shuffling via Optimised Arbitrary-Sized Benes Permutation Network”, *Proceedings of the 5th International Conference – ISC 2002*, vol. 2433, pp. 446–458, 2002.
[9] M. Jakobsson, A. Juels, “Millimix: mixing in small batches”, *DIMACS Technical report 99-33*, 1999.
[10] C. P. Schnorr, “Efficient Signature Generation by Smart Cards”, *Journal of Cryptology*, vol. 4, pp. 161–174, 1991.
[11] Recommendation for Key Management, Special Publication 800–57 Part 1, NIST, 2007.