Youssef BENKARYOUH

Un nuevo sistema P2P basado en perfiles sintéticos de usuarios para la protección de la privacidad en motores de búsqueda manteniendo la calidad de servicio

TRABAJO FIN DE MÁSTER

Dirigido por Dr. Jordi Castellà-Roca y Dr. Alex Viejo

UNIVERSITAT ROVIRA I VIRGILI

Tarragona

2014

Agradecimientos

Primero y antes que nada, dar gracias a *Dios*, por estar conmigo en cada paso que doy, por fortalecer mi corazón e iluminar mi mente y por haber puesto en mi camino a aquellas personas que han sido mi soporte y compañía durante todo el periodo de mis estudios.

Esta tesis la dedico especialmente a la memoria de mi querido padre Sr. Mohamed Benkaryouh. La dedico igualmente a todos los miembros de mi familia, especialmente a mi madre Sra. Saida Assou por su apoyo moral y preocupación, y a mi hermano Sr. Khalid por sus consejos y ayuda siempre que la necesite.

A continuación, quisiera agradecer a todas las personas que me han apoyado de lejos o de cerca para poder realizar esta tesis.

Un agradecimiento especial a mis directores de la tesis, al Dr. Jordi Castellá-Roca, que ha sido un honor hacer un trabajo de fin de estudios con él por la segunda vez; le agradezco su orientación, ayuda, confianza y sobre todo el apoyo moral. Y al Dr. Alex Viejo por su gran ayuda en la revision de la tesis y sus comentarios para producir la versión final de esta tesis.

Agradecer también a la Srta. Cristina Romero-Tris por su colaboración, y al Dr. Jesús A. Manjón por proporcionar las máquinas para poder hacer las simulaciones requiridas en esta tesis.

Finalmente, quisiera agradecer a todos mis profesores y mis compañeros de clase por los buenos momentos que hemos pasado juntos y a todos mis amigos que siempre han sido mi soporte.

UNIVERSITAT ROVIRA I VERGILI

Resumen

Escola Técnica Superior d'Enginyeria
Departament d'Enginyeria Informàtica i Matemàtiques

por Youssef Benkaryouh

Los motores de búsqueda (por ejemplo ***Google***, ***Yahoo***, ***Bing***, etc.) registran información relacionada con las consultas realizadas por los usuarios, lo que permite crearles perfiles que mejoran la calidad del servicio ofrecido (resultados personalizados, sugerencias, correcciones, etc.). La información que contiene el perfil creado por el motor de búsqueda para el usuario, le puede identificar de manera única y así relacionar su identidad con consultas sensibles y confidenciales, y por lo tanto se compromete su derecho de privacidad. Actualmente, existen varias propuestas en la literatura que tratan el problema de privacidad de los usuarios de motores de búsqueda ofreciendo diferentes alternativas que permiten solucionar este problema. Los esquemas basados en la colaboración de usuarios (en redes ***P2P*** o sociales), llamados ***multi-party***, utilizan consultas generadas por usuarios reales para ofuscar los verdaderos intereses del usuario, y así solucionan el problema de detectar las consultas generadas automáticamente utilizadas en los esquemas ***single-party*** que se basan en la ejecución individual del protocolo de privacidad. Aunque los esquemas ***multi-party*** aportan mejoras con respecto a los esquemas ***single-party*** a nivel del tipo de consultas utilizadas, lo que mejora el nivel de privacidad, todavía no se consigue una buena calidad de servicio en estos esquemas. Por esta razón, se propone un nuevo sistema ***P2P*** que permite que los usuarios se agrupen en diferentes categorías en función de su perfil y puedan ejecutar un protocolo para proteger su privacidad manteniendo una buena calidad de servicio al mismo tiempo.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Internet ha dado lugar a una gran cantidad de servicios que han facilitado mucho el día a día de nuestra vida.

Según un estudio reciente del ***Netcraft*** (Febrero 2014) [1], existen más de 920.102.000 sitios web, con un incremento de 58 millones de sitios web en el mes de enero 2014. Ese gran número da una idea sobre la enorme cantidad de información que existe en Internet.

Los motores de búsqueda, conocidos por ***Web Search Engine*** (***WSE***), facilitan encontrar un cierto dato dentro de este gran volumen de información.

Los ***WSE***s son ampliamente usados para buscar información a partir de una o más palabras clave. El resultado obtenido tiene la forma de una lista de enlaces conocida por ***Search Engine Result Pages*** (***SERP***).

La rapidez de los ***WSE***s como ***Google***, ***Yahoo*** o ***Bing***, en encontrar la información, ha aumentado la popularidad de estas utilidades hoy en día hasta que la navegación en Internet ha sido muy difícil sin utilizarlas.

Los ***WSE***s ofrecen un servicio de búsqueda fácil para utilizar por cualquier tipo de usuarios de cualquier edad, ya que no se requiere tener un buen nivel en informática o ni siquiera en ortografía. La mayoría de los ***WSE***s soportan la corrección ortográfica de palabras introducidas por el usuario; mientras escribiendo los términos a buscar o después de validar la búsqueda. Se ofrece también una lista de sugerencias que contiene términos relacionados con el interés del usuario mientras introduciendo los términos deseados, o bien autocompletar los términos introducidos.

Las mejoras y las novedades ofrecidas, han dado lugar a una gran competencia entre los ***WSE***s para ser el motor de búsqueda más utilizado. Ser el más utilizado, quiere decir tener una gran

cantidad de usuarios que desde el punto del ***WSE*** son también clientes de diferentes productos del mismo. Esta competencia para tener el máximo posible de usuarios se justifica por los beneficios económicos de los anuncios publicitarios; que son la fuente principal de estos beneficios, es decir que con más usuarios se obtienen más beneficios. Por ejemplo, ***Google*** ha ganado alrededor de 30 mil millones de dólares en 2010 como beneficios de publicidad [2].

La clave del éxito de ***Google*** es ofrecer publicidad relevante para el usuario; *"Si alguien está interesado en alpinismo y busca información al respecto, le mostramos información sobre promociones para ir a escalar, sobre equipo de montaña o sobre un blog con artículos que tratan el tema"*, explican en [2]. Esto quiere decir que los anuncios que se muestran son relacionados a lo que el usuario está viendo o está buscando.

Para alcanzar la relevancia en la publicidad y ofrecer un mejor servicio para los usuarios, los ***WSE***s recaban la máxima cantidad posible de información para personalizar anuncios y al mismo tiempo personalizar los resultados de las consultas. Por consiguiente, los ***WSE***s crean perfiles a los usuarios basados sobre consultas anteriores realizadas por los mismos. Según la política de la privacidad de ***Google*** [3] por ejemplo, además del texto de la consulta, se pueden almacenar también de forma automática, la dirección ***IP***, tipo de dispositivo, tipo de navegador, idioma del navegador, fecha y hora de la consulta y la ***URL*** de la referencia. Este conjunto de información se le llama registro de consultas, ***Query Logs***. Una vez guardados, los ***Query Logs*** se procesan y se analizan para construir perfiles de usuarios y calcificarlos en función de sus intereses.

Los perfiles creados se emplean con distintas fines. Una de ellas es la búsqueda avanzada. Estos perfiles son una fuente muy valiosa de información por varias razones. En primer lugar, y tal como se ha mencionado antes, se pueden utilizar para mejorar los resultados que los usuarios obtienen para una consulta. Por ejemplo, asumamos que un usuario siempre busca información con palabras clave demasiado cortas para contener número suficiente de términos útiles para discriminar entre los documentos ambiguos. Por esta razón, los ***WSE***s explotan los perfiles de usuarios para resolver las consultas ambiguas [4]. Como ejemplo, consideremos la palabra ***Ratón***. Este término puede referirse al material informático o bien al animal. Con el perfil del usuario, el ***WSE*** puede inferir el sentido correcto en el que el usuario está empleando el término. Por ejemplo, si el usuario ha enviado previamente consultas como ***teclado*** o ***disco duro***, el ***WSE*** asumirá que el usuario está interesado en informática. En consecuencia, los primeros resultados mostrados por el ***WSE*** se referirán al ***Ratón***, el material informático. Según un estudio de ***SeoReacherchers*** [5] sobre la distribución del porcentaje de los ***clicks*** en las listas de resultados, (***SERP***), las primeras cinco posiciones reciben el 88 % de los clicks, y las tres primeras reciben el 79 %. De aquí se puede entender la enorme importancia de la posición de los resultados y su relación con la búsqueda avanzada.

Otras aplicaciones de perfiles de usuarios para mejorar la experiencia del usuario se describen en [6].

Una de las mejores descripciones para describir el precio a pagar por recibir servicios de ***WSE***s es una cita leída en ***metaFilter*** [7] : *"Si no estás pagando por el servicio, entonces tu eres el producto que se está vendiendo"*. A pesar de que el perfil del usuario ayuda a encontrar de manera eficiente los contenidos, también puede poner en peligro su privacidad. En algunos casos, las consultas enviadas contienen información que identifica de manera única a un usuario (nombres, número del ***DNI***, etc.). En otros casos, las consultas pueden contener información sensible (enfermedades, religión, orientación sexual, etc.). Según [6], el principal riesgo del almacenamiento de los ***Query logs***, es pasar esta información a una tercera parte (por ejemplo anunciantes, medios de comunicación, etc.).

Según el funcionamiento de los ***WSE***s, se puede deducir que hay un compromiso entre el nivel de la privacidad y la calidad de los servicios. De un lado, los ***WSE***s ofrecen mejor servicio para usuarios con perfiles completamente exactos, pero con un nivel bajo o nulo de privacidad. Y de otro lado, la privacidad del usuario está más protegida sin perfil, pero los servicios que recibe pueden ser ineficientes.

1.1. Objetivos

Este trabajo de investigación tiene como objetivo hacer un estudio sobre las propuestas de la protección de la privacidad de los usuarios de los ***WSE***s.

A continuación proponer un esquema de privacidad que ofrezca una buena protección de la privacidad y que permita personalizar los resultados mediante la construcción de perfiles sintéticos de usuarios.

El esquema de privacidad se desarrollará, y evaluará mediante un simulador que utilizará consultas reales de usuarios.

Finalmente, se analizarán los resultados obtenidos por las simulaciones.

1.2. Justificación

La protección de la privacidad es un tema que se ha tratado anteriormente en varios trabajos de investigación. Estos trabajos tienen como objetivo principal, mejorar el nivel de protección de la privacidad.

Generalmente este trabajo va en la misma línea, pero además nos interesa mantener un compromiso aceptable entre la protección de la privacidad y la calidad de servicios ofrecidos por los ***WSE***s.

Los trabajos realizados en esta área ofrecen varias alternativas para proteger la privacidad del usuario. En algunos se utiliza la técnica de ofuscación mediante generar consultas sintéticas que pueden ser detectadas como falsas. Este problema se ha solucionado en otros trabajos donde se utilizan consultas reales para ofuscar los intereses del usuario. Las consultas reales que se utilizan son de otros usuarios que están conectados en una red (***P2P*** o sociales). Aunque no se pueden detectar las consultas falsas en este caso, la calidad de servicio que se obtiene es muy ineficiente ya que no se consideran los intereses de los usuarios.

En este trabajo de investigación se implementa un sistema ***P2P*** para proteger la privacidad de los usuarios manteniendo buena calidad de servicios ofrecidos por ***WSE***s.

1.3. Estructura de la memoria

En esta sección se describen generalmente las diferentes partes que formaran este trabajo de investigación.

En el capítulo 2, se presenta el estado del arte, donde se han estudiado trabajos anteriores que tratan el mismo tema. En el capítulo 3 se explica la nueva propuesta, y luego se describe el funcionamiento del sistema y las simulaciones realizadas en el capítulo 4, y después se evalúan los resultados en el capítulo 5. Y finalmente se dan las conclusiones y el trabajo futuro en el capítulo 6.

Capítulo 2

Estado del arte

Cada usuario estima diferentemente el valor de su privacidad, por lo tanto, se requiere un nivel diferente tanto de privacidad como calidad de servicio según la estimación del usuario. Tal como se había mencionado en 1.2, hay varios trabajos que tratan este problema, y que dan varias alternativas como solución. Estas alternativas se pueden clasificar en dos categorías: en la primera se agrupan los esquemas que ofrecen una privacidad perfecta sin tener en cuenta la calidad de servicio, y en la segunda los que protegen la privacidad del usuario y además permiten al ***WSE*** ofrecer un cierto grado de servicios avanzados al usuario. En la Figura 2.1 se muestra una clasificación de los diferentes esquemas de protección de la privacidad.

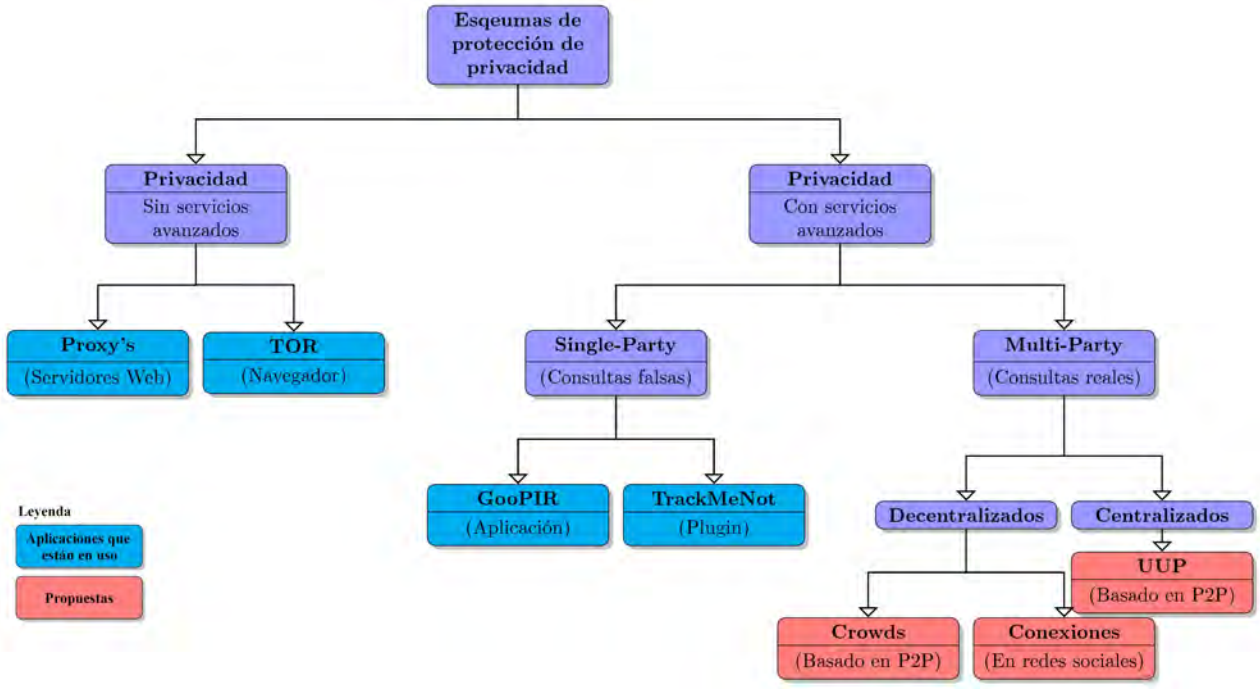

FIGURA 2.1: Clasificación de esquemas de protección de la privacidad.

2.1. Privacidad perfecta sin servicios avanzados

La privacidad perfecta que se ofrece en este caso es gracias a no permitir a los ***WSE***s que generen ningún perfil al usuario. Entre estos esquemas se encuentran los ***proxy's*** anónimos.

En [8] se define el ***proxy*** como un servidor intermedio que acepta conexiones desde un iniciador I y las reenvía hacia un respondedor R. Este servidor transfiere las conexiones de I a R sin revelar la identidad de I a R. De hecho, R ve el servidor intermedio como iniciador de conexiones.

Hay varios ejemplos que se pueden citar, ***DuckDuckGo*** [9], ***Ixquick*** [10], ***PageWash*** [11], ***Start Page*** [12], ***Yippy*** [13], etc. En este caso, los ***WSE***s no conocen el origen exacto de la consulta ya que se envía por el usuario mediante los ***proxy's***. El problema principal de esta técnica, es que el perfil del usuario se puede construir en el ***proxy***. La solución de este problema es la utilización de la técnica de enrutamiento de cebolla (***Onion Routing Technique***), que se basa en la utilización de una red ***proxy***, donde los Routers actúan como anonimizadores. El proyecto ***TOR*** [14] es un ejemplo que explota esta técnica.

No obstante, si no se confía en los ***WSE***s porqué se debería confiar en los ***proxy's***, y además, perdiendo la calidad del servicio.

2.2. Privacidad protegida con servicios avanzados

Esquemas que permiten que los ***WSE***s ofrezcan servicios avanzados hasta un cierto grado, permitiendo-los obtener un perfil destrozado del usuario. El usuario consigue que los ***WSE***s tengan un perfil ofuscado ejecutando un protocolo con la finalidad de añadir un cierto ruido. La idea es enviar consultas falsas para enmascarar las consultas reales del usuario. De hecho, con esta técnica algunos usuarios se pueden identificar, pero sus intereses siguen siendo ocultos. Dentro de esta categoría, los esquemas se clasifican según el número de usuarios que participan en el protocolo:

2.2.1. Esquemas single-party

En estos esquemas el usuario protege su privacidad por sí mismo sin colaboración de otras partes. ***TrackMeNot*** [15] y ***GooPIR*** [16] son dos ejemplos prácticos que pertenecen a la categoría ***single-party***. Estos dos esquemas se basan en la generación de consultas automáticas que se mezclan con las consultas reales del usuario.

GooPIR es una aplicación que se basa en el protocolo de ofuscación de consultas. La consulta que se envía al ***WSE*** es el resultado de una mezcla de la consulta real generada por el usuario y

otras palabras falsas. Este sistema está basado en un concepto llamado *h(K)-private information retrieval*, donde k es un entero no negativo que representa el número de las consultas que hace el usuario (una consulta real y $k-1$ falsas), y $h(\cdot)$ es una función tal que $h(k) \geq 0$. El protocolo utilizado en ***GooPIR*** proporciona *h(K)-private information retrieval* porqué el intruso ve la consulta (q_0) del usuario, como una variable aleatoria (Q_0) con una entropía de ***Shannon*** de $h(Q_0) \geq (k)$. Esto quiere decir que el intruso ve la consulta como una variable (Q_0) que puede tomar varios valores. De esta manera, el intruso no puede determinar de forma inequívoca la consulta real que el usuario ha generado.

Las palabras falsas que se mezclan con la consulta real se obtienen a partir de un tesauro que proporciona las palabras de acuerdo con sus frecuencias relativas. La frecuencia se obtiene basándose en la apariencia en una colección de textos (periódicos por ejemplo) o a partir de la frecuencia de salida cuando se busca la palabra en el ***WSE***. Las frecuencias se utilizan para determinar las palabras a mezclar con la consulta real antes de enviarla al ***WSE***. Las palabras con la misma frecuencia que la consulta real proporcionan una entropía alta.

Aunque de momento no hay ninguna propuesta que demuestra que es posible distinguir entre las consultas reales y las palabras falsas, ***GooPIR*** tiene el inconveniente del uso de palabras en vez de frases como consultas falsas.

El ***TrackMeNot*** es un ***plugin*** para el ***Mozilla Firefox*** que genera consultas dinámicas utilizando el ***RSS***. Para este esquema se utilizan tanto palabras como frases como consultas, y se envían de manera periódica al ***WSE*** con el objetivo de enmascarar los intereses reales del usuario. La idea es producir suficiente ruido alrededor de los patrones reales de las búsquedas del usuario de manera que sea imposible distinguir entre las consultas que el ***TrackMeNot*** produzca y las que los humanos hagan.

El usuario puede personalizar varios parámetros que afectan el funcionamiento del ***TrackMeNot***. Por ejemplo, puede seleccionar el ***WSE*** (***AOL***, ***Bing***, ***Google***, ***Yahoo***, etc.) al cual se enviaran las consultas. El ***TrackMeNot*** permite ajustar la frecuencia con la cual se envían las consultas al ***WSE***, y también elegir el origen del ***RSS*** que se utiliza para producir las consultas falsas.

Uno de los problemas del ***TrackMeNot*** es que envía consultas aleatorias al ***WSE*** cuando el usuario no está muy activo también. Esto puede ser una amenaza para la privacidad del usuario ya que el ***WSE*** puede dividir las consultas en las que se han realizado durante las horas de trabajo (según la zona horario del usuario) y las que se han realizado fuera de este horario. Probablemente, todas las consultas que se han realizado fuera de las horas del trabajo han sido enviadas por el ***TrackMeNot***.

Además, es muy probable que las consultas realizadas por el ***TrackMeNot*** no estén relacionadas al nivel semántico; es decir que mientras los usuarios envían consultas secuenciales sobre el mismo

tema, las consultas enviadas por el ***TrackMeNot*** tienen diferentes temas aleatorios. Basándose sobre el tiempo de realizar las consultas y las características semánticas de las mismas, el ***WSE*** puede detectar las consultas que han sido realizadas por el ***TrackMeNot***. De hecho, hay trabajos [17] y [18] que demuestran que es posible distinguir entre consultas reales y automáticas con una probabilidad de error muy baja (alrededor de $0,02\,\%$).

2.2.2. Esquemas multi-party

En los esquemas ***single-party***, la privacidad del usuario se amenaza por el peligro de detección de las consultas sintéticas ya que se generan automáticamente en la mayoría de los casos, lo que facilita su detección mediante análisis semánticos. Los esquemas ***multi-party*** superan este problema utilizando consultas generadas por usuarios reales para que sea más difícil detectar las consultas falsas. Los protocolos que pertenecen a este tipo de esquemas, tienen como objetivo esconder las acciones de un usuario entre acciones de otros. Una vez conectado, el usuario solicita a otro que le envíe su consulta al ***WSE*** y que le devuelva el resultado después. En consecuencia, los individuales esconden sus identidades en el grupo, y por lo tanto, se obtiene un único perfil para todo el grupo. Los esquemas ***multi-party*** se diferencian principalmente en la manera con la cual se agrupan los usuarios.

En la propuesta [19] se propone el protocolo del perfil inútil del usuario (***UUP***[1]); para cada usuario que quiere realizar una consulta, no se envía la propia consulta si no una consulta de otro usuario en su lugar. Como resultado para este comportamiento, el ***WSE*** no puede generar un perfil real para un determinado usuario. Utilizando herramientas criptográficas, la privacidad se protege por el hecho que los usuarios no saben a quién pertenece cada consulta. El sistema requiere dos componentes: un nodo central y una aplicación cliente. El nodo central escucha las consultas de los usuarios, y cuando recibe n consultas, crea otro grupo y se envían las direcciones ***IP*** y los números de puertos correspondientes a cada miembro del grupo. A continuación, los miembros del grupo establecen las conexiones entre ellos y comienzan a comunicarse sin la interacción con el nodo central. Finalmente, a cada usuario se le asigna al azar una consulta generada por otro miembro del grupo. Y después, cada usuario envía la consulta al ***WSE*** y difunde los resultados. En consecuencia, cada miembro del grupo recibe los resultados de la búsqueda para la consulta que ha generado. El esquema se ha probado en condiciones reales y los resultados muestran que se produce un retardo de $5,2$ segundos con un grupo de tres usuarios y una longitud de clave de $1024 bits$.

Los trabajos presentados en [20], [21] y [22] se basan sobre el protocolo ***UUP***, adaptándolo para diferentes propósitos. Por ejemplo, [20] introduce una modificación en la cual cada miembro del

[1] En Inglés: the Useless User Profile

grupo envía las n consultas generadas al ***WSE*** (cada una se ha generado por un miembro diferente). Esto significa que un usuario siempre envía su propia consulta junto con otras generadas por otros con la intención de ofuscar la suya. La ventaja es que los resultados que los devuelve el ***WSE*** no necesitan ser difundidos y por lo tanto, se consigue un retardo menor que el protocolo ***UUP*** ($3,2$ segundos con un grupo de tres usuarios y una longitud de clave de $1024bits$).

Por otro lado, [21] y [22] sostienen que el protocolo ***UUP*** no es seguro contra los usuarios deshonestos internos. Por lo tanto, se proponen algunas modificaciones con la finalidad de proporcionar más resistencia contra los ataques internos. Para este propósito, [21] aumenta la seguridad del protocolo mediante el uso de encriptaciones dobles, que introducen un retraso alto. Mientras en [22], los autores han diseñado un sistema que emplea redes de permutación y pruebas de conocimiento cero (***ZKP***s[2]). Los resultados muestran que se supera a [21], a nivel de computación y comunicación.

Las propuestas [19], [20], [21] y [22] comparten los siguientes inconvenientes:

- La utilización de un nodo central puede provocar un embotellamiento, ya que cada vez que un usuario quiera enviar una consulta, tendrá que contactar con el nodo central.
- Cuando el nodo central haya recibido n consultas (donde n es el tamaño del grupo), se crea un nuevo grupo. El proceso de crear un nuevo grupo introduce un retardo importante.
- El perfil del grupo que se obtiene, puede no encajar los intereses reales de los usuarios. Al utilizar el esquema ***FIFO*** en crear los grupos, no se consideran los intereses de los usuarios del grupo. Por esta razón, es muy probable que no coincidan los perfiles obtenidos con los intereses reales de los usuarios.
- Estos esquemas no consideran el compromiso entre el nivel de privacidad adquirido con la calidad de servicio recibida.

Algunos de los problemas que genera la centralización del sistema se han solucionado en algunos trabajos que emplean redes descentralizadas.

La propuesta [23] presenta un sistema llamado ***"Crowds"*** en el cual los usuarios esconden sus acciones dentro de acciones de otros. El sistema agrupa a los usuarios en grupos grandes donde las consultas se envían al ***WSE*** a nombre de los otros miembros. Los usuarios se representan en el sistema por procesos llamados ***"Jondos"***. Los ***"Jondos"*** se asignan a una ***"Crowd"*** con otros ***"Jondos"*** mediante un proceso administrativo llamado ***"Blender"***. El ***"Blender"*** se encarga de informar a los nuevos ***"Jondos"*** de otros miembros de la ***"Crowd"*** y también de informar a todos los ***"Jondos"*** cuando un usuario se une o se va de la ***"Crowd"***. Además, cada

[2]En Inglés: Zero Knowledge Proofs

"Jondo" tiene un enlace directo con cada otro ***"Jondo"*** de la red. La comunicación a través del enlace se cifra por una clave que la conocen únicamente los dos ***"Jondos"*** (criptografía de punto a punto). Cuando un usuario quiere hacer una consulta, establece un camino aleatorio a través de la red. Para ello, el usuario escoge aleatoriamente un ***"Jondo"*** y le envía la consulta. Este ***"Jondo"*** lanza una moneda con una probabilidad de reenvío a otro ***"Jondo"*** de $p_f > 0,50$. Dependiendo de resultado del lanzamiento de la moneda, se selecciona otro ***"Jondo"*** al azar o se envía la consulta al ***WSE***. De esta manera, la consulta se acabará enviando al ***WSE*** por algún ***"Jondo"*** y el resultado se devuelve al ***"Jondo"*** que ha generado la consulta siguiendo el camino inverso. La seguridad del sistema se basa en el hecho de que cuando un ***"Jondo"*** recibe una consulta, no sabe si el ***"Jondo"*** que se la ha enviado ha sido el ***"Jondo"*** que ha generado la consulta o sólo se la ha reenviado. Se tiene que tener en cuenta que hay un compromiso entre la probabilidad de reenvío p_f y la longitud del camino elegido y, por lo tanto, el retardo de consulta. Con más jondos que la consulta visite, mayor será su retardo e inferior será el rendimiento para el usuario.

Los inconvenientes del esquema propuesto en [23] son:

- El uso de cifrado punto a punto y de una red donde los nodos están completamente conectados entre ellos requiere que cada nodo almacene tantas claves simétricas como nodos que están en la red. Además de eso, cada salto requiere una encriptación y una des-encriptación. Esto introduce una cierta sobrecarga al sistema.
- Al igual que en las redes anónimas, ***"Crowds"*** sólo protege el transporte de los datos. Los usuarios se encargan de ocultar la información privada.
- La personalización de resultados sólo es posible si los miembros de la ***"Crowd"*** comparten los mismos intereses.
- Este esquema es débil contra el ataque de predecesor [24] : Para atacar ***"Crowds"***, los atacantes pueden simplemente unirse a la ***"Crowd"*** y esperar a que los caminos sean reformados. Cada atacante puede registrar a su predecesor después de cada reformación de camino. Consideremos un usuario U que quiere hacer varias consultas. Debido a la distribución al azar de las consultas entre todos los nodos de la red, U enviará casi todas sus consultas para el resto de los usuarios. Como resultado, varias reformas van a suceder y U aparecerá en todas ellas. Por lo tanto, los atacantes se registrarán U mucho más a menudo que cualquier otro nodo. Después de un gran número de reformas de trayectoria, será claro que U es el usuario que está generando las consultas.

El esquema que se presenta en [25] emplea las conexiones dentro de las redes sociales para distorsionar los perfiles de usuario. De esta manera, un determinado usuario que quiere realizar una consulta puede enviarla directamente al ***WSE*** o reenviarla a un vecino (un amigo en la red

social). El vecino que recibe la consulta tiene las mismas opciones: enviarla al ***WSE*** o reenviarla a uno de sus vecinos. Eventualmente, alguien enviará la consulta al ***WSE*** y los resultados se envían de vuelta al usuario original siguiendo el camino inverso que la consulta ha tomado. Con la finalidad de equilibrar la carga y mantener la privacidad, dos funciones se definen en el sistema. La primera función Φ determina si un usuario debe enviar una consulta al ***WSE*** o pasarla a un vecino; en el segundo caso, Φ determina a qué vecino se debería reenviar la consulta. El objetivo de la función Φ es distribuir de manera igual las consultas para que el ***WSE*** no pueda vincular una consulta al usuario real que la generó. La segunda función α funciona como un esquema basado en la reputación, evaluando el egoísmo de cada usuario y castigar a los usuarios egoístas. Este proceso es similar al proceso introducido por ***"Crowds"***. Sin embargo, en [25], un determinado usuario está conectado solamente con un grupo limitado de usuarios (sus amigos en la red social). Otro punto interesante de esta propuesta es que los grupos de usuarios ya están generados y apoyados por una red social existente (por ejemplo, ***Facebook***). Además, los usuarios que comparten el mismo grupo, se supone que son amigos en la vida real. Esto aumenta la homogeneidad del grupo en términos de intereses compartidos. Por lo tanto, este esquema genera perfiles distorsionados que aún permiten a los usuarios obtener una calidad de servicio adecuada del ***WSE***. Además de eso, este sistema mejora la ex solución en términos de retardo de la consulta.

En el esquema presentado en [26] se propone un protocolo ***P2P*** que explota las redes sociales con el objetivo de proteger la privacidad de los usuarios de ***WSE***s. El artículo presenta una nueva versión del protocolo presentado en [25] con algunas mejoras. En primer lugar, los autores han rediseñado la función Φ con la finalidad de aumentar el nivel de privacidad obtenido por los usuarios. En la nueva versión, esta función distribuye equitativamente las consultas en un camino de longitud dos. Esto significa que no se consideran sólo a los vecinos directos, sino también a los vecinos de vecinos. En consecuencia, el origen de una consulta está mejor ocultado que en [25], ya que en la nueva versión el origen está ocultado dentro de un grupo con una longitud de trayectoria de dos. La segunda contribución de [26] con respecto a [25], es que se propone una nueva métrica que estima la privacidad lograda por los usuarios: el nivel de exposición del perfil (***PEL***[3]). Esta métrica mide el porcentaje de la información acerca de un usuario que se puede extraer de su perfil ofuscado y se calcula como:

$$PEL = \frac{I(\varphi, \varphi')}{H(\varphi)} \cdot 100$$

Donde φ es el conjunto de consultas que el usuario que quiere enviar al ***WSE***, φ' es el conjunto de consultas que el usuario acaba enviando, $H(\varphi)$ es la entropía de φ, y $I(\varphi, \varphi')$ es la información mutua entre ambos conjuntos. Como resultado, ***PEL*** mide el porcentaje de información de

[3]En Inglés: the Profile Exposure Level

usuario que se difunde cuando φ' está revelado. El esquema se probó utilizando consultas reales extraídas desde el registro de consultas publicado por ***AOL***. Los resultados muestran que el promedio del tiempo de respuesta es de $4205,4ms$ ligeramente más alto que el [25]. Sin embargo, el nivel de privacidad alcanzado es mejor: casi el 90 % de los usuarios exponen menos del 20 % del perfil real.

Tal como se ha visto, en los esquemas presentados en [23], [25] y [26], no se requiere un nodo central ya que usan una red de usuarios. Los tres esquemas siguen la misma idea: los usuarios están dentro de la red donde están enviando consultas al ***WSE*** mediante otros usuarios. Cuando un usuario quiere enviar una consulta, establece un camino aleatorio a través de la red. De forma aleatoria, se escoge un vecino y se le envía la consulta. Después, el vecino que ha recibido la consulta decide seguir reenviándola o enviarla al ***WSE***. De hecho, la privacidad se protege en este caso por medio de que el vecino cuando recibe una consulta no sabe si pertenece al usuario que se la ha enviado, o sólo ha sido reenviada. La diferencia principal entre estas tres propuestas es el tipo de la red utilizada para conectar usuarios: mientras que [23] utiliza su propia red propuesta, [25] emplea una red social ya desarrollada (***Windows Live Messenger***, ***Facebook***, etc.), y [26] es una extensión de [25], pero es especialmente desarrollada para redes sociales donde cada usuario sabe cuantos amigos tiene cada uno de sus contactos.

A pesar de las mejoras presentadas en [23], [25] y [26] con respecto a las propuestas anteriormente citadas, todavía no se consigue un compromiso aceptable entre el nivel de privacidad y la calidad de servicio. Mientras [23] no considera la conservación de los intereses del usuario en el perfil de grupo, [25] y [26] suponen que los amigos en las redes sociales comparten los mismos intereses. Sin embargo, esta suposición se podría argumentar hasta un cierto nivel, pero se ha de tener en cuenta que las consultas pueden recorrer un largo camino antes de enviarlas al ***WSE***.

Capítulo 3

Nueva propuesta

A medida que un determinado usuario utiliza un ***WSE*** para encontrar ciertos datos de su interés, el ***WSE*** va registrando las consultas que hace el usuario con la finalidad de crearle un perfil. El perfil que se crea en el ***WSE*** para el usuario es idéntico a su perfil real ya que las consultas llegan directamente al ***WSE*** sin intervención de ningún intermediario (Figura 3.1).

FIGURA 3.1: Envío tradicional de consultas al WSE.

En la sección 2, se han revisado varios trabajos que intentan solucionar el problema de que el perfil que tiene el ***WSE*** sea idéntico al perfil real del usuario. Los esquemas ***single-party*** ofrecen una alternativa práctica para proteger la privacidad, pero la utilización de consultas sintéticas hace que se revelen los intereses del usuario mediante análisis semánticos que detectan las consultas falsas. Este problema se ha solucionado en los esquemas ***multi-party*** utilizando consultas generadas por usuarios reales, difícilmente detectables. Hasta la fecha, los esquemas ***multi-party*** no ofrecen buena calidad de servicio por no considerar los intereses de los usuarios (trabajos basados en redes ***P2P***) o por asumir que los usuarios tienen intereses similares (trabajos basados en redes sociales).

En la nueva propuesta se propone un sistema ***P2P*** anónimo y dinámico que protege más la privacidad. Los sistemas basados en las redes sociales sufren contra ataques internos ya que los

miembros se conocen en la vida real en la mayoría de los casos, lo que hace que la posibilidad de identificar consultas de usuarios sea alta. En este caso, aunque haya protección de privacidad en frente del ***WSE***, se pierde entre los usuarios. Los sistemas que se basan en arquitecturas ***P2P*** estáticas sufren contra el mismo ataque ya que a lo largo de tiempo se pueden deducir los intereses de cada usuario.

La nueva propuesta consiste en la agrupación de los usuarios según sus intereses. Los usuarios se conectan a grupos que presentan las diferentes categorías a las que una consulta pueda pertenecer. El usuario escoge al azar otro usuario del mismo grupo y le envía la consulta. El usuario que recibe la consulta tiene la opción de reenviarla a otro usuario o enviarla al ***WSE*** (Figura 3.2). En este caso, aunque el ***WSE*** registre las consultas que hace un determinado usuario, el perfil resultante contiene información dentro del intervalo de interés del usuario pero no amenaza de ninguna manera a su privacidad. De este modo, los perfiles que se crean en el ***WSE*** se parecen a los perfiles reales pero no se pueden relacionar con las identidades de los usuarios.

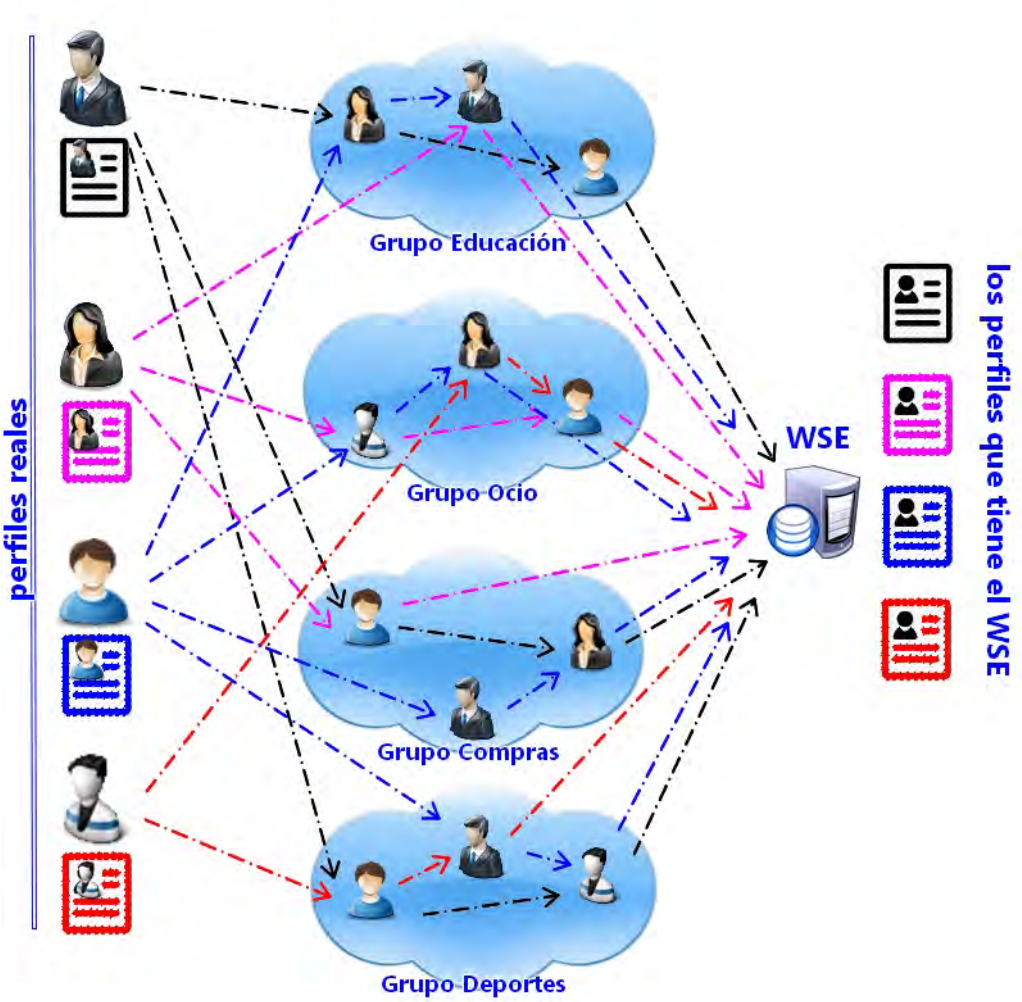

FIGURA 3.2: Nueva propuesta de envío de consultas.

Como objetivo principal de esta propuesta, se quiere conseguir una arquitectura ***P2P*** que permita a los usuarios que se agrupen en diferentes categorías según sus perfiles. En esta sección, se explica detalladamente la arquitectura de la red ***P2P*** no estructurada de manera que los usuarios puedan conseguir la agrupación en categorías en función del perfil. Para ello, se definen a continuación la estructura del perfil considerado y la topología de la red.

3.1. El vector perfil

Para definir el vector que caracteriza el perfil del usuario, el sistema utiliza las categorías definidas en el ***Open Directory Project*** [27]. El ***ODP*** se puede considerar como clasificación muy aceptada de las diferentes categorías que existen de páginas Web. La clasificación se hace a varios niveles donde cada nivel es una especificación del nivel anterior. Por ejemplo, la consulta ***"URV"*** estaría clasificada según ***ODP*** en las siguientes categorías (de general a más especificado): ***"World: Català: Ensenyament: Universitats: Universitat Rovira i Virgili"*** . Por cuestión de simplicidad, L se refiere al nivel de la categoría, así la categoría ***"World"***se encuentra a nivel $L = 1$, la categoría ***"Català"***a nivel $L = 2$, etc. Asimismo, con C_L se refiere al conjunto de categorías que se encuentran en un nivel, y s el número de categorías que contiene el nivel. En consecuencia,

$C_1 = \{c_1, \cdots, c_{16}\} = \{Arte, Negocios, Noticias, Ocio, Referencia, Regional, Ciencia, Compras, Sociedad, Deportes, Mundo\}$.

Se define también $Q_i = \{q_1, \cdots, q_k\}$ como el conjunto de consultas generadas por el usuario i, y $E_i^L = \{c_1, \cdots, c_k\}$ como el conjunto de categorías de nivel L a las que pertenecen las consultas de Q_i. Para obtener la categoría a la que pertenece cada consulta, se utiliza el método de análisis textual, como el propuesto en [28]. Este método aplica, para cada consulta, un análisis morfo-sintáctico y semántico, cuyo resultado final es la categoría ***ODP*** a la que pertenece la consulta.

Finalmente, se define el perfil P_i^L de un usuario i en el nivel L como un vector, donde cada posición corresponde al peso w de una categoría del nivel L, $P_i^L = \{w_{c_1}, \cdots, w_{c_s}\}$. Cada peso w es el número de apariciones de esa categoría en E_i^L.

Por ejemplo, consideremos que nuestro sistema está fijado para $L = 1$. Consideremos también un usuario a que ha generado $k = 3$ consultas $Q_a = \{q_1, q_2, q_3\}$, con categorías $E_a^1 = \{Arte, Deporte, Art\}$. El perfil resultante sería:

$$P_i^L = \{2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0\}$$

Este perfil es dinámico, es decir, a medida que el usuario vaya enviando más consultas, mayor información contendrá su perfil. Supongamos que el usuario envía otra consulta q_1 de arte, y otra q_2 de negocios, su perfil será:

$$P_i^L = \{3, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0\}$$

3.2. Estructura de la red

Los usuarios se conectan entre ellos usando una red ***P2P*** no estructurada formada por varios ***clusters***. El número de los ***clusters*** se determina a partir del número de categorías del nivel L. Esto quiere decir, que para $L = 1$ por ejemplo, tendremos una red ***P2P*** formada por 16 ***clusters***.

Dentro de cada ***cluster***, habrá un nodo llamado ***superpeer*** que se encarga de mantener una lista de direcciones de los nodos que están conectados en este ***cluster***. Dependiendo de su perfil, un nodo puede estar conectado en uno o varios ***clusters***.

Para acceder a la red, un servidor llamado ***bootstrap*** se encarga de mantener y proporcionar la lista de ***superpeers*** (direcciones ***IP***) a los nuevos nodos. A continuación, se describe detalladamente las formas posibles de cambiar la estructura de la red.

3.2.1. Nueva conexión de un nodo

La red se construye gradualmente a medida que los nodos se conectan al sistema. Los pasos que se siguen para añadir un nuevo nodo al sistema son:

1. Consideremos un nuevo usuario i que quiere enviar una consulta q_i. En primer lugar, el sistema extrae la categoría c_j a la que pertenece q_i y crea el perfil inicial del usuario $P_i^L = \{0_{c_1}, \cdots, 1_{c_j}, \cdots, 0_{c_s}\}$.
2. El usuario i se conecta al ***bootstrap*** y obtiene una lista de ***superpeers*** que existen en el sistema. En este paso, existen dos posibilidades:
   a) Que no hayan ***superpeers*** para la categoría c_j. En este caso i se convierte en el nuevo ***superpeer*** de esa categoría, y notifica al ***bootstrap*** enviándole un mensaje. El ***bootstrap*** guarda la ***IP*** de i en la lista de ***superpeers*** asociándola con c_j.
   b) Que haya un ***superpeer*** para la categoría c_j. En este caso, i envía una petición al ***superpeer*** de c_j para ser incluido en su ***cluster***. El ***superpeer*** responde enviando una lista con direcciones ***IP*** de todos los nodos que pertenecen a c_j, y después incluye

la ***IP*** de i en esta lista. La lista de nodos que recibe i del ***superpeer*** se utiliza para ejecutar el protocolo de privacidad que se explica posteriormente en este documento.

3.2.2. Modificación del perfil de un nodo

Existen dos situaciones en las que el perfil del usuario pueda ser modificado:

1. Al realizar una nueva consulta de una categoría de su perfil con peso $w_c > 0$. En este caso, se trata de una categoría ya existente en su perfil, lo que ocurre es incrementar el peso w_c en 1.
2. Al realiza una nueva consulta de una categoría de su perfil cuyo peso $w_c = 0$. En este caso, se cambia el peso a $w_c = 1$, y el usuario tiene que conectarse a un nuevo ***cluster***. Utilizando la lista de ***superpeers*** proporcionada por el ***bootstrap***, el usuario vuelve a ejecutar una de las dos opciones del paso anterior.

3.2.3. Desconexión de un nodo

Una vez que haya una desconexión de un nodo del sistema, surge un cambio en la estructura de la red, y dependiendo del rol del usuario desconectado en el sistema, los siguientes cambios deben realizarse de:

1. Usuario ***superpeer***: antes de su desconexión, se asume que este usuario debería enviar dos mensajes. Un mensaje con una lista de las ***IP***s de los nodos del ***cluster***, que se envía a un nodo escogido de manera aleatoria para que sea el nuevo ***superpeer***. El otro mensaje se envía al servidor ***bootstrap*** para informarle de la ***IP*** del nuevo ***superpeer***.
2. Usuario normal: en este caso, sólo se tiene que enviar una notificación al ***superpeer*** responsable de este nodo para ser borrado de la lista de los nodos conectados. Para no cargar la red enviando notificaciones a todos los nodos del ***cluster*** para informar la desconexión, los otros nodos borrarán al usuario de sus listas en el caso que intenten conectarse con él y reciban un mensaje de error indicando que ya se ha desconectado.

3.3. Protocolo de privacidad

Después de explicar la estructura de la red en la sección anterior, en esta sección se asume que cada usuario que esté ya conectado en el sistema, ya posee de una lista de las direcciones ***IP*** de uno o varios ***clusters*** a los que pertenece su perfil.

Tal como se ha mencionado antes, el protocolo de privacidad tiene como finalidad la protección de las consultas enviadas al ***WSE***. Como idea general del protocolo, las consultas no se envían directamente al ***WSE*** por los usuarios que las generan, sino se envían a otro usuario (nodo) de la red. Este usuario se escoge aleatoriamente dentro del ***cluster*** al cuál corresponde la categoría de la consulta. El usuario escogido puede aceptar o rechazar la consulta. Si la rechaza, el usuario buscará a otro usuario del ***cluster*** que se la acepte. Un usuario que acepta una consulta tiene dos opciones: enviarla al ***WSE*** o reenviarla a otro nodo del ***cluster***. La decisión se toma a base del historial de las consultas enviadas al ***WSE***. En el caso que el nodo decida reenviar la consulta a otro nodo, éste último elige una de las dos opciones hasta que la consulta sea finalmente enviada al ***WSE***. Los resultados obtenidos siguen el camino inverso hasta llegar al nodo que generó la consulta. A continuación se describen detalladamente las fases del protocolo:

3.3.1. Inicialización del protocolo

Una vez conectado, se asume que un nodo i del sistema tiene dos perfiles:

1. Un perfil real P_i^L , construido a partir de las categorías de las consultas que genera, tal como se explica en 3.2.1.
2. Un perfil ofuscado Φ_i^L, construido a partir de las categorías de las consultas que envía al ***WSE***. Este perfil tiene la misma estructura del perfil real.

Para mantener la utilidad del perfil ofuscado, el protocolo de la privacidad intenta conseguir que los dos perfiles se parezcan al máximo posible. Para controlar el nivel de distorsión del perfil, se ha definido el parámetro de ofuscación z que presenta la diferencia máxima que puede existir entre el peso de una categoría en el perfil real $w_{c_j^P}$ y el peso de la misma categoría en el perfil ofuscado $w_{c_j^\Phi}$. Con otra expresión, se ha de cumplir la siguiente condición:

$$w_{c_j^\Phi} \leq w_{c_j^P} + z$$

Además, se define el parámetro ξ como probabilidad de aceptación. Variando en el rango $\{0, 1\}$, éste parámetro indica la posibilidad de que se acepte una consulta enviada por usuario a otro.

3.3.2. Envío de consultas

Esta fase del protocolo, se ejecuta cada vez que se genere una nueva consulta. Se asume que el usuario i genera una consulta q_i. En el caso que se cumpla la condición de que estén conectados por lo menos k usuarios en el ***cluster*** al cual pertenece la consulta, donde k es un parámetro

fijado por el sistema, la consulta pasa por varios pasos durante su camino al ***WSE***. Los pasos que se siguen son los que se describen a continuación:

1. i calcula la categoría c_j a la que pertenece q_i.
2. i incrementa en una unidad el peso $w_{c_j^P}$ de su perfil real.
3. i escoge al azar un nodo v del ***cluster*** que corresponde a c_j.
4. i y v ejecutan un protocolo de compromiso de bits para saber si v debería aceptar o rechazar la consulta.

   a) i escoge aleatoriamente un valor X de longitud suficiente y aplica una función de *hash* con seguridad de computación aceptable $H()$, y obtiene $x = H(X)$.

   b) i envía x a v.

   c) v escoge aleatoriamente un valor Y, aplica la misma función de *hash* y obtiene $y = H(Y)$.

   d) v envía y a i.

   e) i envía X a v, para que verifique si $x == H(X)$.

   f) igualmente v envía Y a i, para que verifique si $y == H(Y)$.

   g) si alguna de las verificaciones falla, los dos nodos se desconectan, y el nodo i reinicia esta fase eligiendo otro nodo v'.

   h) si las verificaciones se llevan a cabo con éxito, i y v calculan un hash que concatena X e Y : $H(X \parallel Y)$. El resultado de este *hash* se usa como entrada para un generador pseudo-aleatorio que genera un valor λ de manera aleatorio entre 0 y 1. Si $\lambda \geq \xi$, v acepta la consulta q_i. Si $\lambda < \xi$, v rechaza la consulta y el usuario i reinicia esta fase del protocolo eligiendo otro nodo v'.

5. Suponiendo que el nodo que finalmente acepta la consulta es v, éste calcula $w_{c_j^{\Phi_v}} + z$, y lo compara con $w_{c_j^{P_v}}$:

   a) Si $w_{c_j^{\Phi_v}} \leq w_{c_j^{P_v}} + z$, v envía la consulta al ***WSE***. Y además, incrementa una unidad el peso $w_{c_j^{\Phi_v}}$ de perfil ofuscado.

   b) Si $w_{c_j^{\Phi_v}} > w_{c_j^{P_v}} + z$, v escoge al azar otro nodo r del ***cluster*** que corresponde a c_j. En este punto, se vuelve a ejecutar el protocolo entre v y r a partir del paso 4.

En el caso que no se cumpla la condición que hayan por lo menos k usuarios conectados, el usuario que quiere hacer la consulta elige una opción entre esperar a que se conecten k usuarios o enviar la consulta al ***WSE***.

3.3.3. Recepción de los resultados

Cuando un nodo llega a enviar la consulta al ***WSE*** después de pasar por todos los pasos de la fase anterior, los resultados obtenidos del ***WSE*** comienzan el camino inverso hacia el origen de la consulta empezando por éste último nodo que envía la consulta al ***WSE***. En esta fase, ninguno de los nodos conoce el camino completo que ha seguido la consulta, simplemente reciben los resultados del nodo al que se la reenviaron, y los pasan al nodo del que la recibieron. Siguiendo este comportamiento, la consulta llega hasta el usuario que la generó.

Capítulo 4

Análisis del sistema

Para llevar a cabo el análisis del sistema de la propuesta, se ha implementado una aplicación para simular las consultas que enviaría cada usuario al ***WSE*** si utilizara este sistema. La aplicación permite también hacer varias estadísticas sobre las simulaciones, como el número de saltos que realiza una consulta antes de recibir la respuesta, y el tiempo que pasa viajando en la red hasta que vuelva a su origen. Durante el proceso de la simulación, se puede comparar en cada momento el perfil real con el perfil ofuscado de cada usuario.

Con la finalidad de analizar el funcionamiento del sistema y los resultados obtenidos por el simulador que emplea el protocolo de privacidad previamente definido, se han definido algunas métricas para evaluar los beneficios que obtienen los usuarios utilizando este sistema:

- El promedio de saltos: Los saltos presentan el número de nodos que una consulta ha visitado durante su viaje hacia el ***WSE*** en ambos sentidos.
- La proporción del perfil real conocida por el ***WSE***. Por esta proporción se refiere al porcentaje de las consultas propias que un usuario envía al ***WSE***. Por ejemplo, un usuario ha generado 100 consultas en total, pero de estas 100, sólo ha enviado 25 directamente al ***WSE*** y el resto se ha enviado a otros usuarios para que se encarguen de enviarlo al ***WSE***. Por lo tanto, la proporción del perfil real que conoce el ***WSE*** es $25/100 = 0,25$.
- La distancia entre el perfil real y el perfil ofuscado del usuario. Dado el perfil real del usuario $P_i^L = \{w_{c_1}^P, w_{c_2}^P, \cdots, w_{c_s}^P\}$, y el perfil ofuscado $\Phi_i^L = \{w_{c_1}^\Phi, w_{c_2}^\Phi, \cdots, w_{c_s}^\Phi\}$, la distancia entre ambos $d(P, \Phi)$ se calcula aplicando la formula:

$$d(P,\Phi) = \sqrt{(w_{c_1}^P - w_{c_1}^\Phi)^2 + (w_{c_2}^P - w_{c_2}^\Phi)^2 + \cdots + (w_{c_s}^P - w_{c_s}^\Phi)^2}$$

- El tiempo de respuesta, o el tiempo de vida de una consulta. Es el tiempo que pasa una consulta viajando en la red desde que se genere hasta que esté enviada al ***WSE*** y devuelta a su origen otra vez.

Los resultados que se estudian en la próxima sección están basados sobre consultas reales extraídas desde datos proporcionados por ***AOL***. En primer lugar, las consultas se ordenaron por las fechas en las que fueron realizadas, y luego se reciben por el simulador de acuerdo con el tic (segundo) correspondiente.

Las consultas extraídas desde los datos proporcionados por ***AOL*** tienen el formato mostrado en la Figura 4.1.

```
592896  3001225 hairball medicine   2006-03-01 00:01:03.0   0
712244  3613173 batman signal images    2006-03-01 00:01:04.0   12  http://www3.flickr.com
```

FIGURA 4.1: Ejemplo de registro de consultas AOL.

Debido al gran volumen de datos ofrecidos por ***AOL***, al gran número de archivos generados por el simulador (tres por cada usuario), al gran número de simulaciones que se tienen que hacer y a la necesidad de maquinas demasiado potentes para hacerlas, se han considerado consultas realizadas durante un día entero únicamente.

4.1. Descripción de los parámetros

Después de varias pruebas, se ha hecho la siguiente lista de parámetros que más afectan al comportamiento del simulador y del protocolo también:

- El número mínimo de usuarios (K): (Ver 3.3.2).
- El tiempo de inactividad (T_{inac}): el tiempo máximo permitido al usuario que quede inactivo en la red. Es decir, si durante x tics el usuario no realiza ningúna acción se desconecta automáticamente del sistema.
- La ofuscación (z): (Ver 3.3.1).
- Los intentos (I): número máximo de oportunidades para encontrar un usuario que acepte una consulta.
- La probabilidad de aceptación (ξ): (Ver 3.3.1).
- Porcentaje de egoístas (Ego): el porcentaje de usuarios que rechazan consultas que les envían otros.

Por cuestión de simplicidad, se han fijado algunos parámetros. En todas las simulaciones, el nivel de las categorías se ha fijado en $L = 1$. El parámetro K se ha fijado en $K = 3$, ya que si hay por lo menos tres usuarios no se puede acertar el origen de la consulta al 100 %. Para ofrecer un análisis más completo, se ha ido variando los valores de nivel de ofuscación, tiempo de inactividad, intentos, probabilidad de aceptación y porcentaje de egoístas. Concretamente, el sistema ha sido simulado para cuatro valores de ofuscación ($z = \{0; 2; 5; 10\}$), dos valores de tiempo de inactividad ($T_{inac} = \{10; 100\}$), tres valores de intentos ($I = \{3; 10; 100\}$), cuatro valores de probabilidad de aceptación ($\xi = \{0{,}25; 0{,}5; 0{,}75; 1\}$), y cuatro valores también del porcentaje de egoístas ($Ego = \{0{,}25; 0{,}5; 0{,}75; 1\}$).

4.2. Organización de las simulaciones

Combinando diferentes valores de los diferentes parámetros del sistema, se han tenido que hacer 480 simulaciones. Para facilitar el proceso, las simulaciones se han repartido por 30 grupos, llamados $configuraciones$, donde cada configuración contiene 16 simulaciones. En cada configuración, se fija el valor de tiempo de inactividad en 10 o 100, donde 10 es equivalente a la disponibilidad mínima y 100 a la disponibilidad máxima de un usuario en el sistema. Se fija también el número de intentos en 3, 10 o 100; donde 3 es equivalente a pocos intentos, 10 a suficientes intentos y 100 a ilimitados intentos para encontrar un usuario que acepte una consulta. El resto de los parámetros va cambiando en el rango que corresponde. En cada configuración se fija el valor del porcentaje de egoístas para todas las simulaciones.

4.3. Implementación del simulador

Para llevar a cabo estas simulaciones, se ha programado un simulador en ***Java***. Este simulador es una aplicación que está formada por varias clases, donde cada clase presenta un componente del sistema. Para poder hacer análisis y estadísticas, se ha decidido crear algunos atributos en cada clase que permiten almacenar la información necesaria para ello.

A continuación se explican las clases más importantes de la aplicación:

4.3.1. Clase *Consulta*

Es una estructura de datos que contiene varios atributos, donde algunos forman parte del protocolo de la privacidad y otros ayudan en el proceso de analizar y hacer las estadísticas.

Los atributos que forman parte del protocolo son:

- *Categoria*: a la cual pertenece la consulta.
- *TextoConsulta*: el texto de la consulta.
- *HashConsulta*: es un valor que lo calcula el propietario original de consulta, básicamente es el resultado de la función de hash ***SHA1*** con una entrada formada por la concatenación de tres variables: un número entero escogido al azar dentro de un intervalo suficiente grande para evitar colisiones, la identificación del usuario que hace el hash y el texto de la consulta. El hecho de poner un número aleatorio al principio de la cadena a la cual se aplica la función de resumen, evita la identificación de la consultas entre los usuarios, otra ventaja que tiene, es aún que un usuario reciba dos consultas con el mismo texto, este atributo permite gestionar la devolución de las respuestas.
- *Respuesta*: la respuesta obtenida del ***WSE***.
- *Estado*: es un ***Flag*** que ayuda saber si la consulta ha llegado al ***WSE*** o no.

Los atributos que se utilizan en el proceso de análisis son:

- *Camino*: permite saber todos los nodos por los cual ha pasado la consulta.
- *TiempoDeVida*: la totalidad de tiempo que pasa la consulta viajando en la red desde su lanzamiento por el propietario hasta llegar a ***WSE*** y volver otra vez a su origen. Este tiempo se incrementa también mientras se hacen intentos para encontrar un nodo que acepte la consulta.
- *Saltos*: el total de los nodos que han sido visitados por la consulta.
- *Info*: este atributo permite saber por qué razón la consulta ha sido enviada por el último usuario al ***WSE***, el valor de este atributo puede ser uno de la lista siguiente:{No hay suficientes usuarios para reenviar la consulta, consulta duplicada, se ha alcanzado el maximo de usuarios o para completar el perfil}.

4.3.2. Clase *HashMapConsulta*

Por necesidad de agrupar las consultas en listas, los usuarios han de tener una lista donde guardar las consultas que reciben y las que emiten. Esta clase proporciona una estructura de almacenamiento tipo clave-valor. Las claves son el atributo *HashConsulta* y el valor es objeto Consulta que corresponde.

4.3.3. Clase $Usuario$

Esta clase tiene varios atributos y funciones. La complejidad de esta clase es alta debido a que contiene también las funciones que ejecutan el protocolo de privacidad.

Los atributos de esta clase son:

- Id: identificador del usuario.
- $TiempoInactividad$: es una variable que incrementa cada tic que el usuario no realice ninguna acción. Esta variable va incrementando hasta alcanzar el tiempo máximo permitido de inactividad que exige el sistema, y entonces el usuario se desconecta automáticamente.
- $Egoista$: es un booleano que indica si el usuario es egoísta o no, esta variable no forma parte del protocolo de manera directa, se utiliza por necesidad de hacer las simulaciones. Esta variable se controla por el sistema y según el porcentaje de egoístas establecido para la simulación, un usuario puede ser egoísta si hay menos egoístas de los que deberían haber en un momento dado, o no si ya se ha alcanzado el porcentaje exigido.
- $ListaNegra$: es una lista donde se guardan los usuarios que rechazan las consultas cuando el resultado del compromiso de bits indica que la consulta debería ser aceptada pero se ha rechazado.
- $perfilReal$: es un vector de 17 posiciones, donde cada posición presenta una categoría del nivel $L = 1$. Normalmente sólo deberían ser 16, pero se ha añadido una posición más por el hecho de haber consultas sin categoría (por ejemplo una consulta tipo "*aaaaaaa*") que no tienen ningún sentido. Cada vez que se haga una consulta propia nueva, la posición correspondiente se incrementa en una unidad.
- $perfilWSE$: con las mismas características que el $perfilReal$, pero se utiliza para saber las consultas que ha enviado el usuario al ***WSE***. Con otras palabras, es el perfil que tiene el ***WSE*** de este usuario.
- $consultasP$: es una lista que no forma parte del protocolo, y sólo se utiliza en el proceso de escribir las estadísticas en un archivo de texto. Tiene las mismas características que el $perfilReal$ y el $perfilWSE$, pero se modifica solamente cuando el usuario envía una consulta propia al ***WSE***.
- $consultasA$: lo mismo que $consultasP$. Se modifica cuando el usuario envía una consulta de otro al ***WSE***.
- $userHash$: es un $HashMap$ donde se guarda la pareja clave-valor $(HashConsulta, Id_{usuario})$, es necesario para saber de quién se ha recibido la consulta, y por lo tanto devolver las respuestas de manera eficiente en el camino inverso.

- $userHashDup$: hay casos cuando se recibe una consulta que se había recibido previamente (el mismo $HashConsulta$), en este caso, se asume que el usuario debería aceptar la consulta y mandarla directamente al ***WSE*** para evitar ataques de identificación de consultas en la red. Este $HashMap$ evita modificar una pareja con la misma clave en $userHash$.

La clase $Usuario$ dispone también de varias funciones que permiten ejecutar el protocolo de la privacidad, y otras que son utilidades necesarias para hacer cálculos o escribir resultados en el disco duro.

A continuación se explica el rol de cada una de las funciones más importantes:

- $actualizarPerfil()$: esta función se encarga de actualizar los diferentes vectores del perfil ($perfilReal, perfilWSE, ConsultasP$ o $ConsultasA$). Recibe como parámetros el vector y la posición que serán actualizados.
- $aceptarConsulta()$: con esta función se decide aceptar o rechazar una consulta de otro usuario. Hay cuatro casos posibles:
  - Que el usuario actual sea egoísta: en este caso se rechaza automáticamente la consulta.
  - Que el usuario que quiere enviar la consulta esté en la lista negra: en este caso también se rechaza automáticamente.
  - Que la consulta sea duplicada (es decir ya está en la lista de consultas): en este caso se acepta para evitar ataques que identifican las consultas.
  - En el último caso, se decide a partir del resultado del compromiso de bits si la consulta será aceptada o rechazada.
- $enrutarConsulta()$: según el parámetro de ofuscación, si decide reenviar la consulta a otro usuario o enviarla a ***WSE***, dependiendo si se cumple la siguiente condición o no:

$$perfilWSE\,[cat] \leq (perfilReal\,[cat] + ofuscacion(z))$$

- $activo()$: devuelve un booleano que indica si el usuario está conectado o no.
- $nuevaConsulta()$: esta función determina la acción que hacer cuando se reciben consultas nuevas. Hay tres casos:
  - Consulta propia nueva: en este caso, la consulta aún no tiene asignado el $HashConsulta$, entonces se tiene que calcular, y para ello, primero se calcula un valor aleatorio dentro de un rango bastante grande, y luego se concatena con el id del usuario y el texto de la consulta y se pasa por una función de hash (***SHA1***). Después, se añade la consulta en la lista de consultas del usuario, y también se añade la pareja $(HashConsulta, id)$

en la lista $UserHash$ y por último se actualiza el $perfilReal$ del usuario utilizando la función $actualizarPerfil()$.

- Consulta recibida de otro usuario: la consulta recibida se añade a la lista de las consultas. Si es la primera vez que se recibe esta consulta se añade la pareja ($HashConsulta$, $idUsuarioAnterior$) en la lista $UserHash$, en caso contrario se añade en la lista de consultas duplicadas $UserHashDup$.
- Respuesta recibida: (se puede saber que es una respuesta mediante en el Estado de la consulta). En este caso solamente se añade a la lista de consultas para poder entregarla luego al usuario que se la ha enviado previamente.

- $enviarConsulta()$: si la función $aceptarConsulta()$ devuelve $true$, la consulta se envía al usuario que la ha aceptado. En caso contrario, si el usuario a quien se quería enviar la consulta, la tuviera que aceptar y el la ha rechazado, en este caso se añade el Id de este usuario a la $ListaNegra$.

- $enviarRespuesta()$: se utiliza para enviar las consultas que ya han sido enviadas al ***WSE***. Las consultas que se envían por esta función se encuentran el camino inverso, es decir, están viajando hacia donde se generaron la primera vez.

- $sevirConsultas()$: es la función más compleja de esta clase, se explicará por partes para que se pueda entender bien. En esta función se toman las varias decisiones, y se aplica el protocolo de la privacidad. Se recorre toda la lista de las consultas y según el caso se realiza la acción que corresponde. En esta lista hay dos tipos de consultas, consultas de camino hacia el ***WSE***, y otras en el camino inverso hacia el origen:

  - Consultas en camino hacia ***WSE***:

    Entre estas consultas hay consultas propias y consultas ajenas.

    Para las consultas propias:

    - Si la consulta está duplicada, es decir, es propia pero la ha reenviado otro usuario hacia el usuario actual (la consulta ha hecho un ciclo). En este caso el usuario está obligado a enviarla al ***WSE***. Se actualiza el vector del $perfilReal$, del $perfilWSE$ y de $consultaP$. Se establece el estado de la consulta a "OK" para indicar que ya se ha enviado al ***WSE*** y luego se establecen los atributos ($Camino$, $Info$, $Saltos$ y $TiempoDeVida$) de la consulta para poder hacer estadísticas más tarde. Al final, se escribe esta consulta y las estadísticas en los archivos que corresponden a este usuario.
    - Si la consulta es nueva, primero se verifica si el número de usuarios es mayor o igual a K (valor establecido por el sistema), en este caso, el usuario tiene un número determinado de intentos para encontrar un usuario que acepte la consulta. Si se agota el máximo de intentos sin que la consulta se acepte por otro

usuario o hay menos de K usuarios. El usuario tiene dos opciones: esperar hasta el siguiente tic o enviarla al ***WSE***. El usuario escoge una de las dos opciones al azar, y si se escoge la segunda opción, se tendrán que hacer las mismas actualizaciones y acciones que se hacen en el caso anterior.

Para las consultas ajenas:

- Si la consulta está duplicada: (la consulta ha hecho un ciclo). En este caso, el usuario está obligado a enviar la consulta al ***WSE***. Después, Se actualiza el vector del $perfilWSE$ y el vector de $consultaA$. Y también se establecen los atributos estadísticos de la consulta y se escriben las estadísticas en los archivos correspondientes del usuario.
- Si la consulta no está duplicada: según lo que devuelve la función $enrutarConsulta()$, se decide enviar la consulta al ***WSE*** si el usuario necesita consultas para completar su perfil, en este caso, se actualiza el vector del $perfilWSE$, el vector de $ConsultasA$ y los atributos estadísticos de la consulta, y Luego se escriben las estadísticas en los archivos correspondientes del usuario. En el caso que el perfil del usuario esté completado, se decide reenviar la consulta a otro usuario, en este caso el usuario tiene un número determinado de intentos para encontrar un usuario que acepte la consulta. Si el usuario llega a encontrar alguien que le acepte la consulta, se la envía mediante la función $enviarConsulta()$, y luego establece los atributos estadísticos de la consulta. En el caso contrario, el usuario se encuentra obligado a enviar la consulta al ***WSE*** aún que su perfil esté completado. En este caso, se actualiza el vector del $perfilWSE$, el vector de $ConsultasA$ y los atributos estadísticos de la consulta, y se escriben las estadísticas también en los archivos correspondientes del usuario.

- Consultas en el camino inverso:

  Son las consultas que tienen el Estado establecido a "OK". Para estas consultas hay dos casos también, consultas propias y consultas ajenas.

  Para las consultas propias:

  - Si la consulta está duplicada, es decir, se ha enviado previamente a otro usuario, pero le ha tocado a al usuario actual enviarla al ***WSE*** porqué la consulta ha hecho un ciclo, en este caso, se tiene que enviar la respuesta también al usuario de quien se recibió la consulta. Después, se borran las entradas correspondientes en las listas $UserHash$ y $UserHashDup$, y se elimina también la consulta desde la lista de las consultas. Finalmente, se escriben los resultados en los archivos correspondientes.
  - Si la consulta no está duplicada, sólo se borra la entrada del $UserHash$ y se escriben los resultados en los archivos correspondientes.

  Para consultas ajenas:

- Si la consulta está duplicada, se tiene que enviar la respuesta a los dos usuarios de quienes se recibió la consulta. Después, se borran las entradas correspondientes en las listas $UserHash$ y $UserHashDup$, y se elimina también la consulta desde la lista de las consultas. Finalmente, se escriben los resultados en los archivos correspondientes.
- Si la consulta no está duplicada, solamente se envía al usuario que corresponde. Y después, se hacen las actualizaciones necesarias.

- makeSHA1Hash(): se utiliza para calcular el resumen de una cadena utilizando el ***SHA1***.
- EscriberConsultas(): es la función que escribe en archivos, los diferentes resultados obtenidos.

4.3.4. Clase *UsuarioHashMap*

Esta función facilita la agrupación de los usuarios en grupos (***clusters***).

4.3.5. Clase *Grupo*

Proporciona funciones para gestionar entrada y salida de usuarios.

4.3.6. Clase *Principal*

Carga y lee los parámetros globales de las simulaciones.

4.3.7. Clase *Simulacion*

En esta clase se crean los hilos para llevar a cabo el proceso de simulación y también se crean los hilos que se encargan de analizar los resultados y proporcionar las estadísticas en forma de gráficas.

4.4. Funcionamiento del simulador

El simulador tiene como entrada un archivo de texto que contiene las consultas ordenadas por tiempo. Este archivo se va leyendo línea por línea hasta el final, donde cada línea es un conexión nueva al sistema en el caso de un usuario nuevo, o una modificación del perfil si el usuario que hace la consulta ya existe en el sistema. De mientras, el protocolo de privacidad se va ejecutando por los usuarios hasta que se sirvan todas las consultas. Como salida, el simulador genera tres archivos[1] para cada usuario, el primero contiene sus propias consultas con las respuestas que incluyen: la consulta, el total de los saltos, el camino completo que ha seguido en ida y vuelta, el tiempo que ha pasado viajando en la red, y la razón por la cual ha sido enviada al ***WSE*** (No hay nadie que la quiere aceptar, o se han agotado los intentos, o para completar el perfil, etc.). El segundo archivo contiene todas las consultas que envía el usuario al ***WSE*** con los mismos datos que el archivo anterior. El tercero contiene información para poder comparar los dos perfiles del usuario.

Después de simular todas las simulaciones de una configuración, el simulador genera un archivo de estadísticas para cada una de las diez-i-seis simulaciones, este archivo tiene cinco columnas, donde la primera presenta el identificador del usuario, la segunda presenta el promedio de saltos, la tercera presenta el porcentaje del perfil real que tiene el ***WSE***, la cuarta presenta la distancia entre los dos perfiles del usuario, y la última presenta el promedio del tiempo que han quedado las consultas del usuario viajando en la red. Una vez calculadas estas estadísticas, la aplicación genera gráficas que representan estas estadísticas al nivel de la configuración.

[1] Los datos que contienen estos archivos no son visibles para otros usuarios, se registran solamente para facilitar el estudio del sistema. La única información que sabe un usuario de una consulta que ha recibido, a parte del texto, es el usuario que se la ha enviado.

Capítulo 5

Estudio de los resultados

En esta sección, se estudian los diferentes resultados obtenidos. Las 480 simulaciones que se han hecho, generan 90 graficas para estudiar el comportamiento del sistema de manera general, es decir, sin distinguir entre usuarios honestos y usuarios egoístas. Se generan también 72 graficas para estudiar el comportamiento de los usuarios honestos, y otras 72 graficas para estudiar el comportamiento de los usuarios egoístas.

Además de la gran cantidad de estas gráficas que hacen que el estudio sea difícil, la cantidad de los parámetros que utiliza el sistema lo hace aún más difícil. Para estudiar un sistema con 5 parámetros, y evaluar como afecta cada uno de ellos con respecto a otros, se necesita una matriz de 5 dimensiones. Esto quiere decir que el estudio de forma manual se debería olvidar como opción a elegir.

Para encontrar una alternativa que permita estudiar los resultados de manera eficiente, se ha tomado como punto de partida el objetivo de implementar este sistema, que es alcanzar un compromiso aceptable entre el nivel de privacidad proporcionado y la calidad de servicios obtenida. Es decir, lo que se quiere obtener es:

- Un promedio bajo de saltos.
- Una proporción baja, del perfil del usuario, conocida por el ***WSE***.
- Una distancia mínima entre el perfil real y el perfil ofuscado del usuario.
- Un tiempo[1] bajo de respuesta.

[1]El tiempo se ha medido con una unidad "propia" del sistema. El motivo por no utilizar el reloj del sistema operativo es porqué devuelve valores al orden de milis y son muy similares, la cosa que hace que la comparación sea bastante difícil.

Para dar prioridad al nivel de privacidad en frente de la calidad del servicio, se tiene que mantener el orden propuesto de estos tres objetivos.

Según la práctica, el caso ideal sería:

- Un promedio de saltos entre 4 y 6; con menos saltos se supone que la consulta la generó el usuario de quién se ha recibido.
- Una proporción de 0 %; el ***WSE*** no tiene información sobre el perfil real del usuario.
- Una distancia igual a 0; el perfil que tiene el ***WSE*** es parecido al perfil real del usuario.

La metodología que se ha seguido en el estudio de los resultados es la misma que se utilizó en [25], que consiste en buscar las situaciones similares o que se acercan al caso ideal, y luego estudiar cómo afecta el cambio del valor de algunos parámetros. En este caso, hay 5 parámetros (K, T_inac, z, I y Ego (Ver 4.1)) que determinan el comportamiento del sistema. Entre estos parámetros, hay 2 que no se pueden controlar por el sistema (el tiempo que un usuario esté conectado y la cantidad de usuarios egoístas en la red), en cambio los otros 3 pueden tomar valores fijos y el sistema los puede controlar. La idea es encontrar la combinación de estos 3 parámetros controlables para alcanzar el caso ideal en 2 situaciones: cuando los usuarios están disponibles solamente durante poco tiempo, y cuando la duración de disponibilidad es larga. Después de identificar estas combinaciones, se evalúa la resistencia contra el incremento del número de usuarios egoístas en la red, es decir, hasta qué grado se mantiene el compromiso obtenido entre el nivel de privacidad y la calidad de servicios que se alcanzan con el sistema bajo condiciones ideales.

A continuación se estudian todos los casos posibles obtenidos mediante la combinación de los varios valores de los parámetros del sistema. Se muestran solamente las gráficas relacionadas con los casos similares al caso ideal.

5.1. Un sistema sin egoístas

Para estudiar el comportamiento ideal del sistema en la ausencia de los usuarios egoístas, se han hecho 96 simulaciones agrupadas en 6 configuraciones. Para cada configuración (16 simulaciones), se definen dos categorías de pruebas. Una categoría de pruebas con tiempo mínimo de disponibilidad de usuarios, que se ha fijado en 10*tics*, y otra categoría con tiempo de disponibilidad máximo de 100*tics*.

Para cada categoría se ha estudiado como afecta cada uno de los tres parámetros controlables por el sistema a nivel de promedio de saltos y tiempo, porcentaje de perfil conocido por el ***WSE*** y la distancia entre el perfil real y el perfil ofuscado.

Para esta sección el estudio global del comportamiento del sistema lleva a las mismas conclusiones que el estudio del comportamiento parcialmente en el caso de usuarios honestos ya que no hay usuarios egoístas.

5.1.1. Disponibilidad mínima de usuarios

5.1.1.1. Promedio de saltos

Las figuras (5.1, 5.2 y 5.3) muestran la variación del promedio de saltos que se ve afectado de manera directa por la variación del parámetro de ofuscación, más concretamente, con más ofuscación menos saltos hacen las consultas. Y esto se puede explicar de la menara siguiente: cuando más grande es el parámetro de ofuscación, se necesitan más consultas para que los usuarios completen su perfil. Se observa también que no hay una gran diferencia entre la Figura 5.1 y la Figura 5.2; y esto es debido a que no se requieren más de 10 intentos para encontrar un usuario que acepte una consulta enviada o reenviada por otro, o sea 10 intentos son suficientes.

Al reducir el número de intentos a 3, se puede ver la bajada a nivel del promedio de saltos (Figura 5.3). En este caso también, se ve la influencia de la probabilidad de aceptación; a menos aceptación se hacen menos saltos porqué los usuarios se encuentran obligados a enviar las consultas al ***WSE*** en vez de reenviarlas a otros usuarios una vez alcanzados los 3 intentos permitidos.

El tiempo de vida de las consultas se afecta de manera directa por la probabilidad de aceptación; cuando es alta, se requieren menos intentos para encontrar usuarios que acepten las consultas y viceversa.

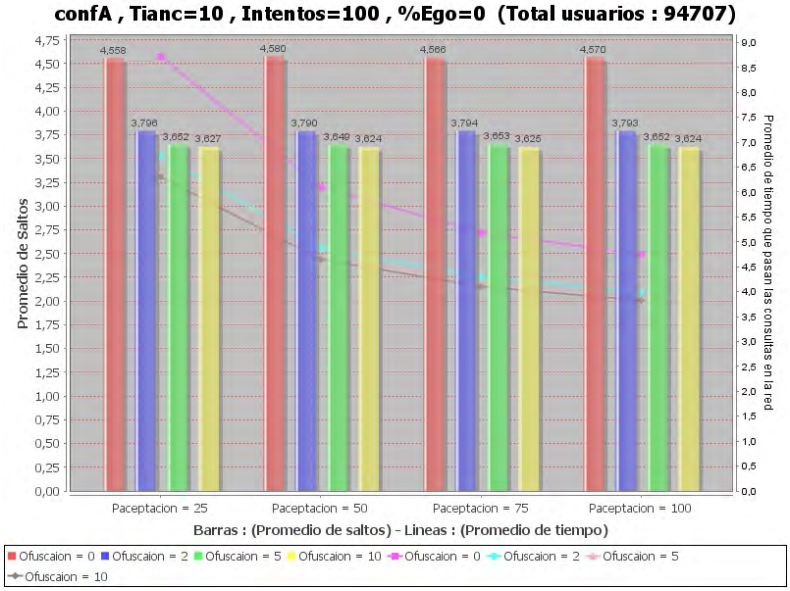

FIGURA 5.1: Promedio de saltos y tiempo de respuesta (Configuración A).

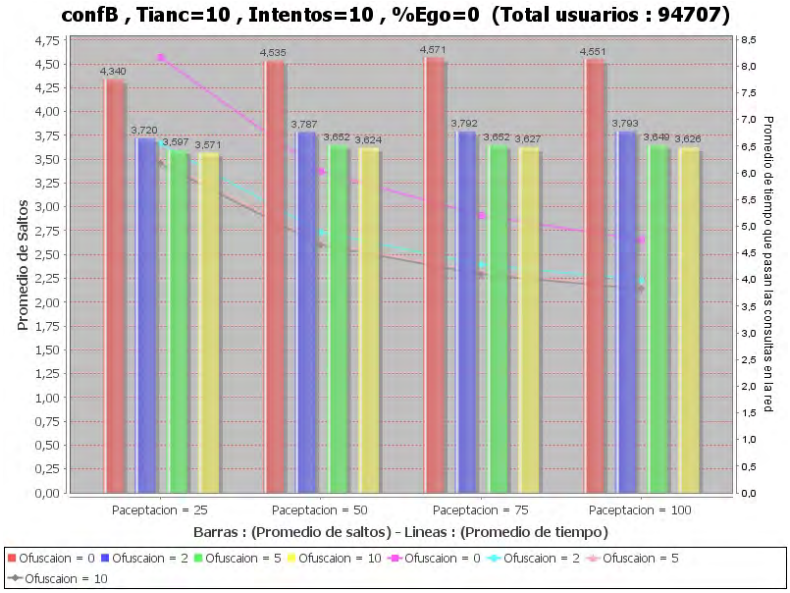

FIGURA 5.2: Promedio de saltos y tiempo de respuesta (Configuración B).

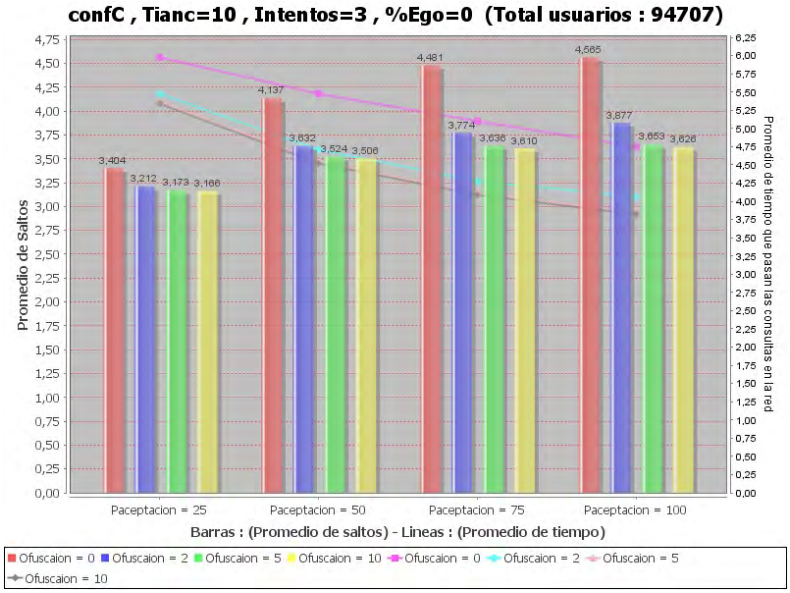

FIGURA 5.3: Promedio de saltos y tiempo de respuesta (Configuración C).

		$P_{aceptacion}$				
		25 %	50 %	75 %	100 %	*intentos*
ofuscacion	0	4,55801369	4,58032646	4,56564733	4,56983192	100
		4,34004057	4,53470451	4,57075594	4,55084564	10
		3,40351118	4,13677036	4,48084496	4,56535496	3
	2	3,79578384	3,79040562	3,79432461	3,7928856	100
		3,72010818	3,78712506	3,7919073	3,79313613	10
		3,21194578	3,63241056	3,77371299	3,87740479	3
	5	3,65177556	3,6491891	3,65326617	3,6522042	100
		3,59666302	3,65189034	3,65158087	3,649144	10
		3,17261018	3,52394719	3,63620747	3,65274869	3
	10	3,62674133	3,62427617	3,6249095	3,62444441	100
		3,57112378	3,62387259	3,62669199	3,62572217	10
		3,16618198	3,5056852	3,61031432	3,62554558	3

TABLA 5.1: Variación del promedio de saltos (Disponibilidad mínima de usuarios y 0 % de egoístas)

La Tabla 5.1 resume las 3 figuras anteriores mostrando resultados de diferentes combinaciones de todos los parámetros controlables; las casillas en color gris presentan el promedio de saltos similar al caso ideal. Se concluye que cuando los usuarios están disponibles durante poco tiempo con ofuscación alta, las consultas hacen menos saltos porqué los usuarios necesitan consultas para completar el pefil ofuscado. En este caso, los usuarios necesitan menos de 10 intentos para encontrar usuarios que colaboren en reenviar las consultas.

5.1.1.2. Proporción del perfil real que tiene el *WSE*

En las tres figuras, (5.4, 5.5, y 5.6), se ve claramente que la proporción del perfil conocida por el ***WSE*** no se afecta por la variación del parámetro de ofuscación sino por la probabilidad de aceptación; cuando se aceptan las consultas con poca probabilidad, se acaban enviando al ***WSE*** por los usuarios que las han generado. En los mejores casos, la proporción está entre 18 y 19 por ciento.

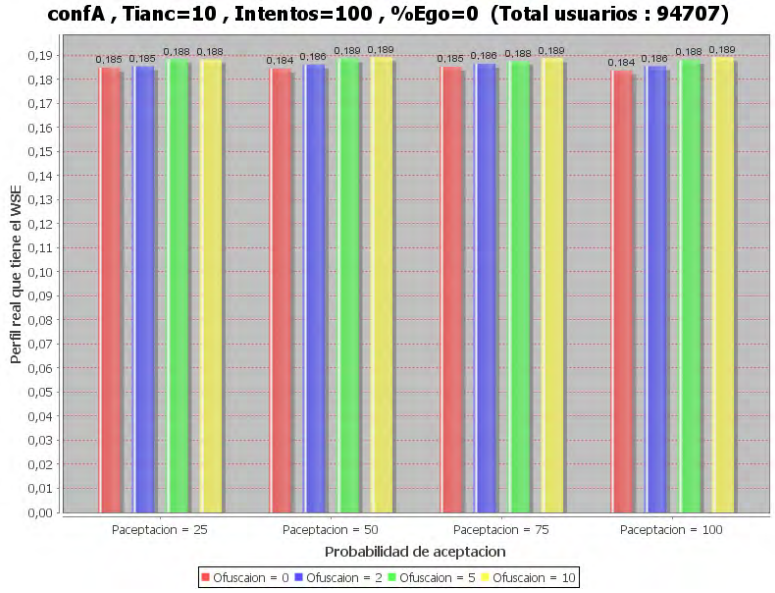

FIGURA 5.4: Proporción del perfil real conocida por el WSE (Configuración A).

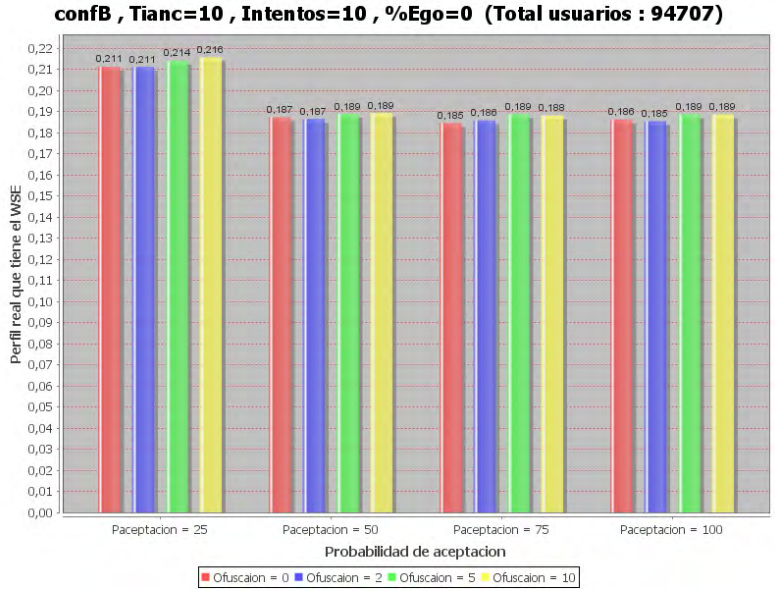

FIGURA 5.5: Proporción del perfil real conocida por el WSE (Configuración B).

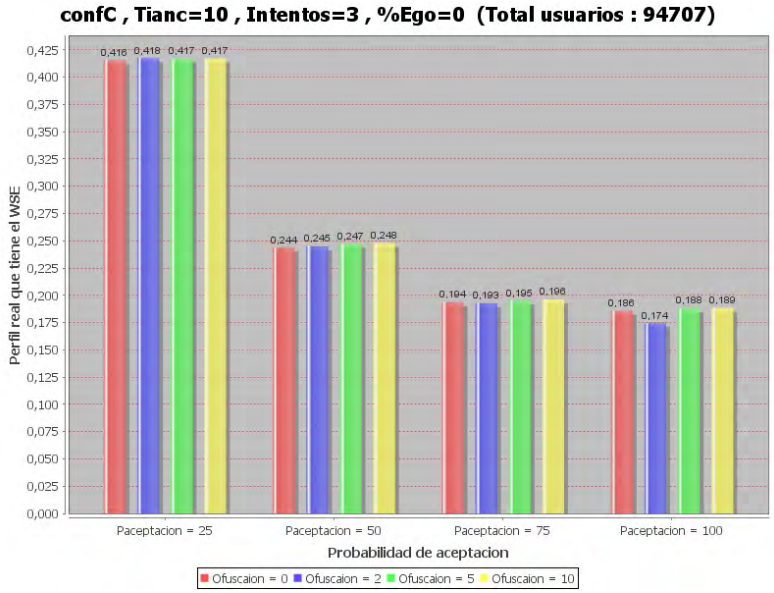

FIGURA 5.6: Proporción del perfil real conocida por el WSE (Configuración C).

Por otro lado, el número de intentos afecta también a la variación de la proporción: cuando los usuarios tienen pocas oportunidades para encontrar usuarios que acepten sus consultas, se encuentran obligados a enviarlas ellos mismos al ***WSE***, sobre todo cuando la probabilidad de aceptación es baja.

		$P_{aceptacion}$				
		25 %	50 %	75 %	100 %	*intentos*
ofuscacion	0	0,18485206	0,18445489	0,1852173	0,18363404	100
		0,21118834	0,18723685	0,18450002	0,18619329	10
		0,41555578	0,24388513	0,19361444	0,1858093	3
	2	0,18534964	0,18615079	0,186465	0,18552811	100
		0,21112962	0,18650878	0,18581084	0,18544151	10
		0,41750379	0,24494999	0,19277796	0,17444295	3
	5	0,18839023	0,18871077	0,1875551	0,18812945	100
		0,21406407	0,18879326	0,18872413	0,18870404	10
		0,41716135	0,24727967	0,19529883	0,18813147	3
	10	0,18829904	0,18917141	0,18887436	0,18921695	100
		0,21575835	0,18925467	0,18809077	0,18858262	10
		0,41709915	0,24797138	0,19621371	0,18852736	3

TABLA 5.2: Variación del porcentaje del perfil real conocido por el WSE (Disponibilidad mínima de usuarios y 0 % de egoístas)

Siguiendo con la misma idea de encontrar los casos similares al caso ideal, en la Tabla 5.2, se resumen las figuras anteriores. Con el orden de prioridad previamente mencionado (Promedio de salto, Proporción y Distancia), interesa que el valor de la proporción del perfil conocida por el ***WSE*** sea lo más pequeño posible en los casos similares al caso ideal a nivel de promedio de

saltos. Los valores marcados en casillas grises, corresponden a los mejores resultados que se han obtenido.

Sin ofuscación, se sigue teniendo un nivel aceptable de la privacidad (el ***WSE*** solamente conoce alrededor del 20 % del perfil real del usuario), aunque los valores suben cuando la probabilidad de aceptación es demasiado pequeña.

5.1.1.3. Distancia entre el perfil real y el perfil ofuscado

Evaluar la distancia entre el perfil real y el perfil ofuscado del usuario es equivalente a evaluar la usabilidad y la calidad del servicio que recibirá el usuario utilizando este sistema. La ofuscación es el parámetro que afecta más a la variación de la distancia entre los dos perfiles según las figuras (5.7, 5.8 y 5.9). Cuando el usuario ofusca más su perfil, es como si estuviera enviando consultas fuera del intervalo de su interés, y por lo tanto incrementa la distancia entre su perfil real y su perfil ofuscado.

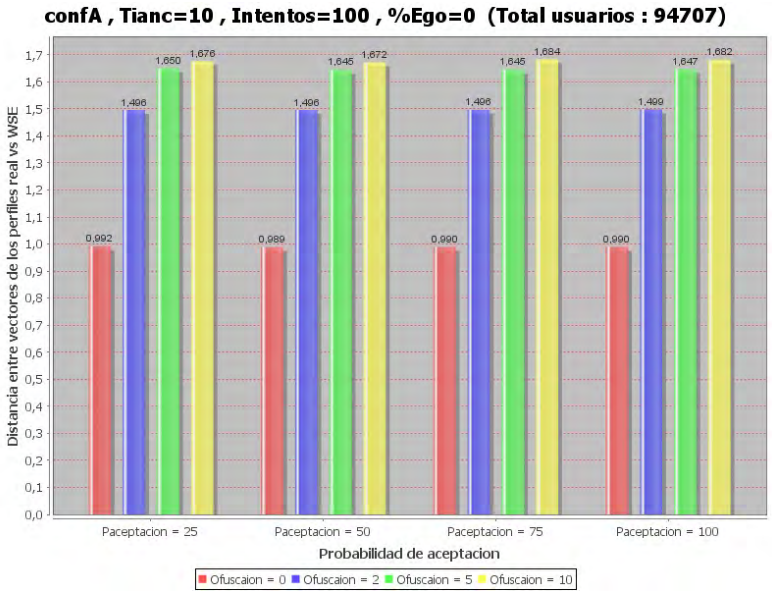

FIGURA 5.7: Distancia entre el perfil real y el perfil ofuscado (Configuración A).

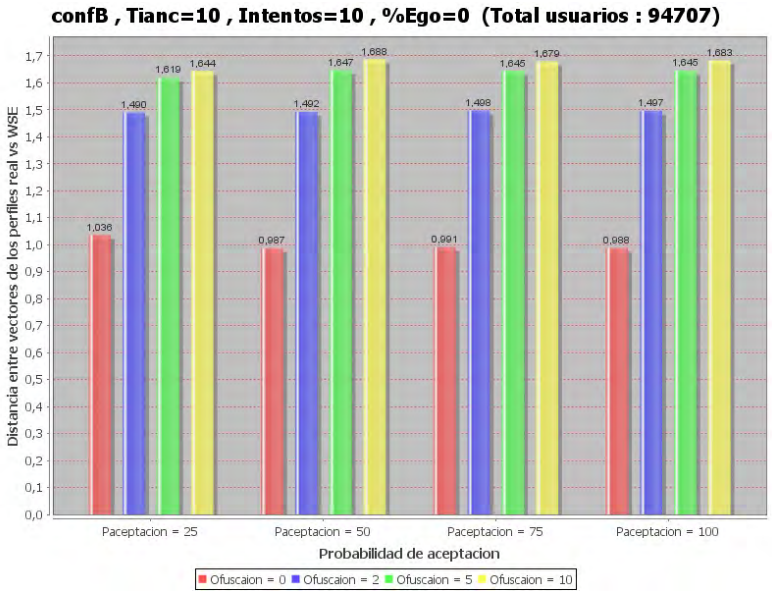

FIGURA 5.8: Distancia entre el perfil real y el perfil ofuscado (Configuración B).

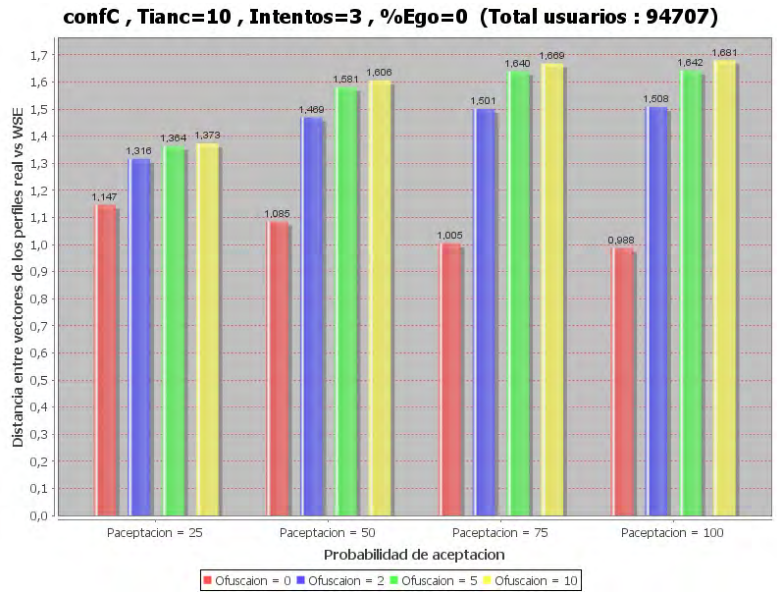

FIGURA 5.9: Distancia entre el perfil real y el perfil ofuscado (Configuración C).

Hacer más o hacer menos intentos, o que la probabilidad de aceptación sea grande o pequeña, no cambia prácticamente nada en frente del parámetro de ofuscación.

En la Tabla 5.3, se puede comprobar que los mejores resultados coinciden con los casos similares a la situación ideal anteriormente identificada. La distancia entre el perfil real y el perfil ofuscado esta alrededor de 1, y prácticamente se considera como un valor muy aceptable. La distancia 1 es equivalente a enviar una consulta de más o menos en el total de consultas que un usuario quiere realizar.

		$P_{aceptacion}$				
		25 %	50 %	75 %	100 %	*intentos*
ofuscacion	0	0,99193242	0,98872128	0,98951636	0,98966813	100
		1,03628028	0,9871964	0,99138361	0,98817932	10
		1,14701491	1,08503409	1,00450462	0,9876686	3
	2	1,49578681	1,49574989	1,4962536	1,49875997	100
		1,48981654	1,49242532	1,49753391	1,49657366	10
		1,31622585	1,46937872	1,50067878	1,50808693	3
	5	1,65007965	1,64482845	1,64513816	1,64687844	100
		1,6193262	1,64716501	1,64461264	1,64547378	10
		1,36432596	1,58136612	1,64025367	1,64241316	3
	10	1,67591402	1,67229897	1,68402022	1,68173247	100
		1,64420764	1,68777009	1,67883182	1,68339137	10
		1,37348091	1,60580061	1,66889227	1,6807002	3

TABLA 5.3: Variación de la distancia entre el perfil real y el perfil ofuscado (Disponibilidad mínima de usuarios y 0 % de egoístas)

5.1.1.4. Tiempo de respuesta

Ahora que se han encontrado 11 casos similares al caso ideal, bajo las dos condiciones de poca disponibilidad de usuario y ausencia de los usuarios egoístas en el sistema, sería muy interesante hacer una clasificación para todos estos casos.

A pesar que el sistema ofrece en estos casos un buen nivel de privacidad y calidad de servicio, el tiempo de respuesta es también un requisito que se tiene que cumplir. En la Tabla 5.4, se muestran los diferentes valores del tiempo de vida de la consulta en la red dependiendo de los parámetros controlables del sistema.

		$P_{aceptacion}$				
		25 %	50 %	75 %	100 %	*intentos*
ofuscacion	0	8,72856101	6,09991733	5,18519837	4,74758949	100
		8,16933264	6,02552828	5,19946011	4,73088533	10
		5,979289	5,48197247	5,1004497	4,74869506	3
	2	6,72862308	4,8916954	4,29232296	3,98489316	100
		6,5455419	4,88212228	4,28568951	3,98901958	10
		5,47680023	4,7102054	4,27177454	4,06045231	3
	5	6,35232311	4,67521911	4,13041193	3,84929098	100
		6,23202701	4,68000445	4,12567447	3,84502957	10
		5,36282528	4,54942948	4,12057973	3,84474001	3
	10	6,29892595	4,64620304	4,09326949	3,82186942	100
		6,18484042	4,64240223	4,09967843	3,82294221	10
		5,34616868	4,52228391	4,086426	3,82179892	3

Tabla 5.4: Variación del tiempo de respuesta (Disponibilidad mínima de usuarios y 0 % de egoístas)

A partir de la Tabla 5.4, se deduce la clasificación de estos 11 casos previamente encontrados según el tiempo de respuesta. Esta clasificación se muestra en la Tabla 5.5.

#	Tiempo	Saltos	Proporción	Distancia	P_{acep}	*Intentos*	*Ofusc.*
1	4,73088533	4,55084564	0,18619329	0,98817932	100	10	0
2	4,74758949	4,56983192	0,18363404	0,98966813	100	100	0
3	4,74869506	4,56535496	0,1858093	0,9876686	100	3	0
4	5,1004497	4,48084496	0,19361444	1,00450462	75	3	0
5	5,18519837	4,56564733	0,1852173	0,98951636	75	100	0
6	5,19946011	4,57075594	0,18450002	0,99138361	75	10	0
7	5,48197247	4,13677036	0,24388513	1,08503409	50	3	0
8	6,02552828	4,53470451	0,18723685	0,9871964	50	10	0
9	6,09991733	4,58032646	0,18445489	0,98872128	50	100	0
10	8,16933264	4,34004057	0,21118834	1,03628028	25	10	0
11	8,72856101	4,55801369	0,18485206	0,99193242	25	100	0

Tabla 5.5: Clasificación de los mejores casos (Disponibilidad mínima de usuarios y 0 % de egoístas)

Como se puede ver en la Tabla 5.5, el tiempo de vida de la consulta en la red depende en primer lugar de la probabilidad de aceptación; las consultas llegan más rápido al ***WSE*** cuando la probabilidad de aceptación es alta. Para el número de intentos, se puede ver que 10 intentos son suficientes ya que no se nota una gran diferencia en el tiempo cuando el máximo de intentos es 10 o 100.

En el caso que los usuarios estén disponibles para poco tiempo en el sistema, la privacidad está más protegida cuando la probabilidad de aceptación es máxima. Además, se consiguen también los mejores resultados a nivel de calidad de servicio.

5.1.2. Disponibilidad máxima de usuarios

5.1.2.1. Promedio de saltos

En la situación que los usuarios estén conectados en el sistema durante mucho tiempo, según las figuras (5.10, 5.11 y 5.12), hay muchos casos similares al caso ideal (promedio de saltos entre 4 y 6).

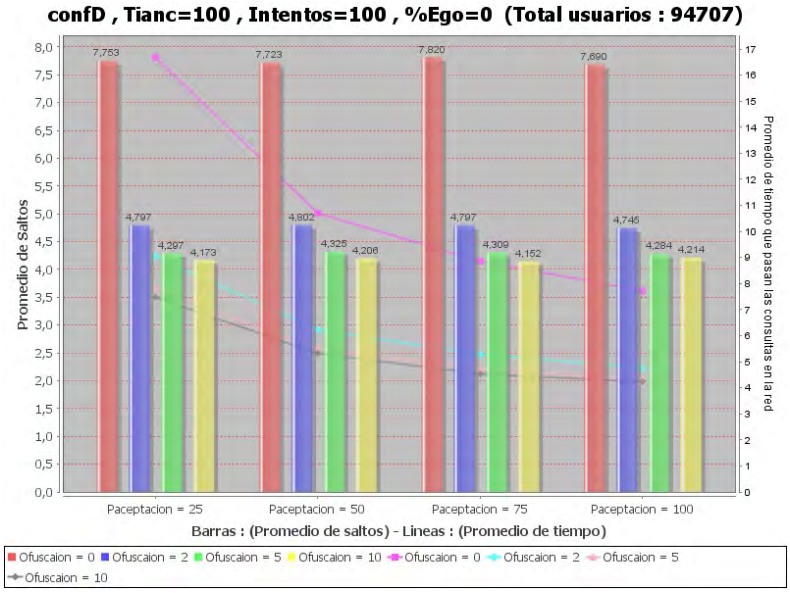

FIGURA 5.10: Promedio de saltos y tiempo de respuesta (Configuración D).

La variación del parámetro de ofuscación afecta de manera directa al promedio de saltos en esta situación también. Con más ofuscación, menos saltos hacen las consultas. En este caso se aplica el concepto de ***co-privacidad*** [29]; hay usuarios que necesitan enviar más consultas para ofuscar el perfil, y otros usuarios que les interesa que sus consultas se envíen al ***WSE*** mediante otros.

Cuando la ofuscación varia de 0 a 2, ocurre una reducción relevante (de 7,8 a 4,8) del promedio de saltos sobre todo en las figuras (5.10 y 5.11), donde se ganan hasta 3 saltos en algunos casos. Esta ganancia se consigue también a nivel de tiempo. De hecho, hay casos donde el tiempo de respuesta se reduce en 6 unidades (5.10). La explicación de este comportamiento es, cuando los usuarios están mucho tiempo conectados en el sistema con ofuscación 0, llegan momentos que la mayoría de los usuarios ya tienen el perfil completado. En estos casos, los usuarios siguen aceptando consultas pero no para ser enviadas al ***WSE***, sino para reenviarlas a otros usuarios.

En las figuras (5.10 y 5.11), la variación en la probabilidad de aceptación no afecta al promedio de saltos tanto como afecta al promedio de tiempo; con menos aceptación más tiempo se pierde haciendo intentos para encontrar usuarios que acepten la consulta. Por la gran semejanza entre

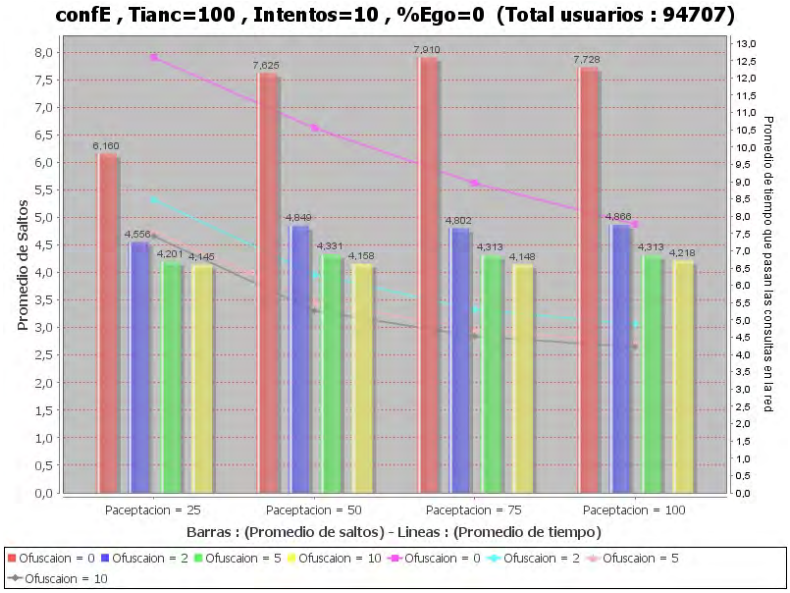

FIGURA 5.11: Promedio de saltos y tiempo de respuesta (Configuración E).

las figuras (5.10 y 5.11), se puede decir que 10 intentos son suficientes para encontrar usuarios que acepten las consultas. La diferencia se nota cuando el número de intentos baja a 3 (5.12), donde cambia el comportamiento del sistema, ya que a menos aceptación para ofuscación 0, los usuarios se encuentran obligados a enviar las consultas al ***WSE*** después de alcanzar los 3 intentos permitidos.

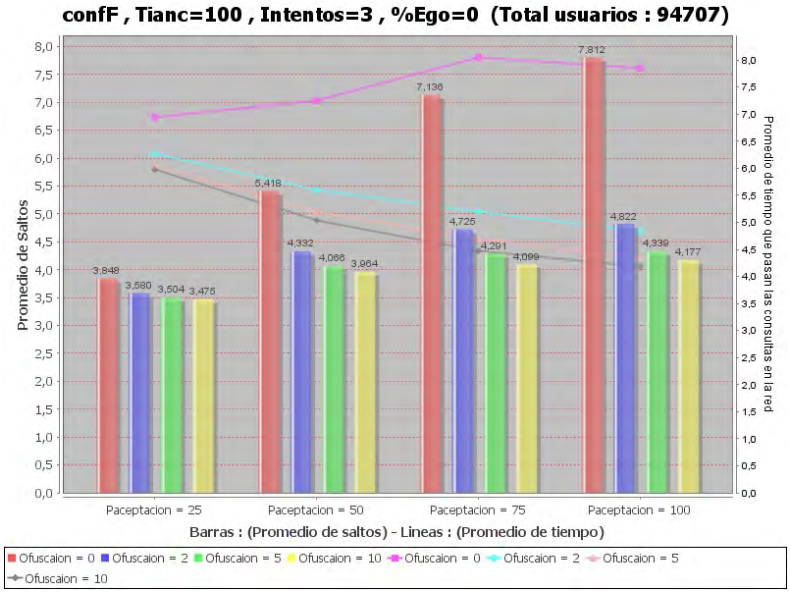

FIGURA 5.12: Promedio de saltos y tiempo de respuesta (Configuración F).

		$P_{aceptacion}$				
		25 %	50 %	75 %	100 %	*intentos*
$ofuscacion$	0	7,75345244	7,723184	7,819538	7,68951853	100
		6,15961008	7,62548215	7,90955915	7,72764025	10
		3,84821765	5,41768433	7,13635104	7,81179637	3
	2	4,79667591	4,80181385	4,79743188	4,74503817	100
		4,55607416	4,84944288	4,8019268	4,86581888	10
		3,57978373	4,33228705	4,72484732	4,8218896	3
	5	4,29678824	4,32492185	4,30918472	4,28412894	100
		4,20112652	4,33100041	4,31312064	4,31253744	10
		3,50367545	4,06601531	4,29145418	4,33923484	3
	10	4,17253099	4,20630702	4,1521081	4,21432736	100
		4,14463071	4,15838786	4,14797737	4,21843787	10
		3,47491083	3,9643717	4,09944794	4,17711225	3

TABLA 5.6: Variación del promedio de saltos (Disponibilidad máxima de usuarios y 0 % de egoístas)

La Tabla 5.6 resume las tres figuras anteriores. Las casillas en gris presentan los casos similares al caso ideal a nivel del promedio de saltos.

5.1.2.2. Proporción del perfil real que tiene el *WSE*

Cuando los usuarios están conectados durante mucho tiempo en el sistema, encuentran más rápido consultas para completar el perfil ofuscado, y también sus consultas encuentran usuarios que las hacen llegar al ***WSE***. La mayoría de los valores obtenidos en las figuras (5.13, 5.14 y 5.15) a nivel de la proporción del perfil real conocida por el ***WSE*** son muy cercanos al caso ideal (proporción igual a 0). La ofuscación es el parámetro que afecta más a la variación de estos valores en cada caso; con más ofuscación, más consultas de sobra se envían por los usuarios para una categoría determinada, y por lo tanto menos información tiene el ***WSE*** sobre los intereses reales reales de los usuarios. Nuevamente se vuelve a demostrar que con 10 intentos se puede alcanzar resultados muy parecidos a cuando el máximo de intentos permitidos sea 100.

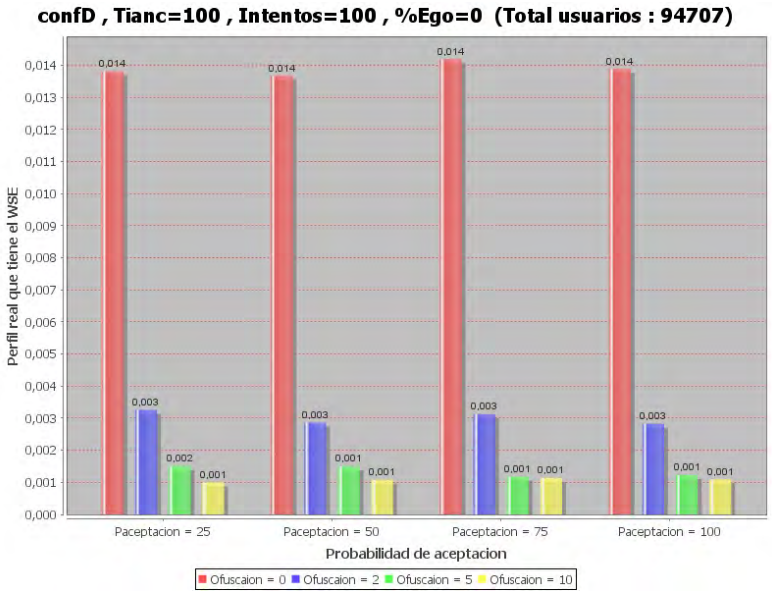

FIGURA 5.13: Proporción del perfil real conocida por el WSE (Configuración D).

La variación de la probabilidad de aceptación es muy importante sobre todo cuando se permiten solamente 3 intentos; con menos aceptación se alcanza el máximo de intentos sin que el usuario encuentre a otro que le acepte la consulra, y por lo tanto se encuentra obligado a enviar sus consultas por sí mismo. Con más aceptación y además a la gran cantidad de usuarios conectados, las consultas tienen más oportunidades para que se acepten aunque el máximo de intentos sea 3.

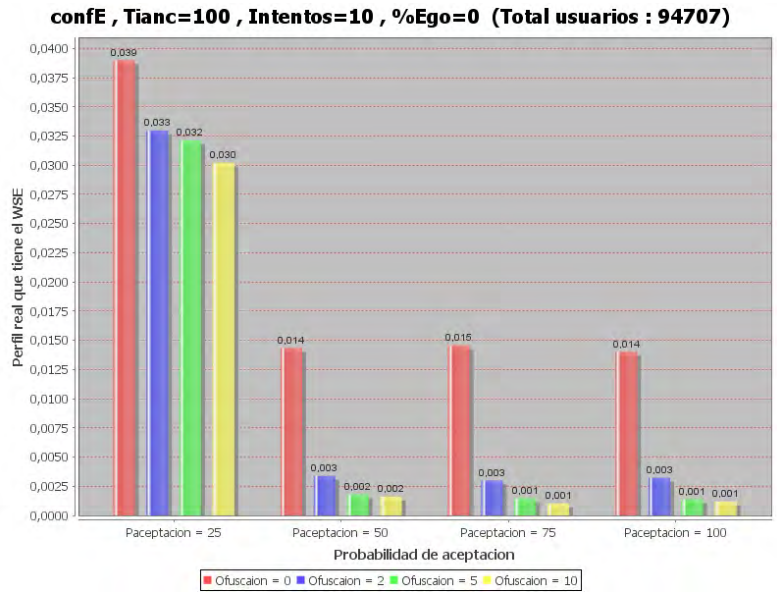

FIGURA 5.14: Proporción del perfil real conocida por el WSE (Configuración E).

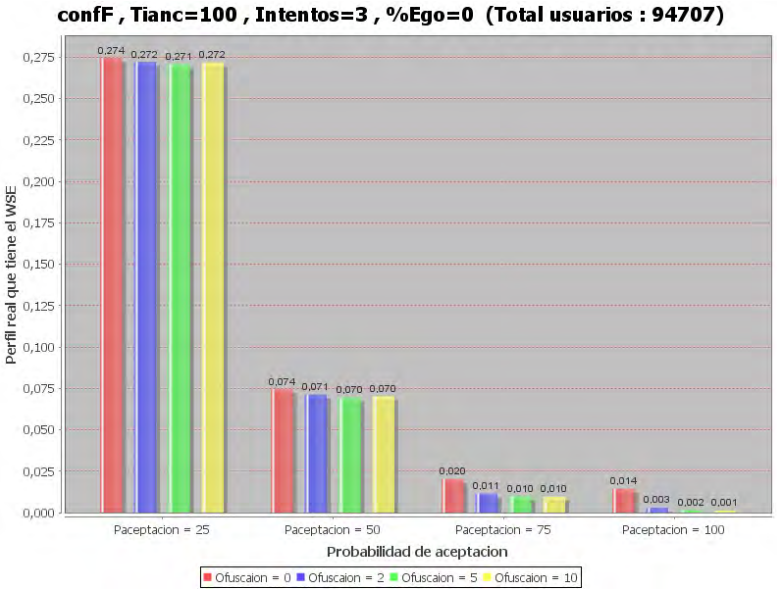

FIGURA 5.15: Proporción del perfil real conocida por el WSE (Configuración F).

		$P_{aceptacion}$				
		25 %	50 %	75 %	100 %	*intentos*
$ofuscacion$	0	0,01380875	0,01367035	0,01418885	0,01388162	100
		0,03899925	0,0143456	0,01456676	0,01402155	10
		0,27447085	0,07435257	0,02016093	0,01434579	3
	2	0,00325635	0,00286178	0,00312306	0,00282097	100
		0,03297675	0,00338332	0,0029947	0,00322699	10
		0,27215576	0,07122511	0,01143215	0,0029539	3
	5	0,00150715	0,00149834	0,00117103	0,00122482	100
		0,0321318	0,00180324	0,00146284	0,00139954	10
		0,27069153	0,0695625	0,00996759	0,00155159	3
	10	0,00100217	0,00107983	0,00113404	0,00109476	100
		0,03020438	0,00161148	0,00103237	0,00118897	10
		0,27173676	0,07032508	0,0096474	0,00128741	3

TABLA 5.7: Variación del porcentaje del perfil real conocido por el WSE (Disponibilidad máxima de usuarios y 0 % de egoístas)

La Tabla 5.7 resume las figuras (5.13, 5.14 y 5.15). Se puede observar que en todos los casos similares al caso ideal, los valores son muy cercanos a 0. Esto quiere decir que el ***WSE*** tiene poca información sobre el perfil real del usuario y por consiguiente el nivel de protección de privacidad en frente al ***WSE*** está muy alto.

5.1.2.3. Distancia entre el perfil real y el perfil ofuscado

La distancia entre el perfil real y el perfil ofuscado, se afecta principalmente por el incremento del parámetro de la ofuscación. De hecho, cuando más grande sea el parámetro de ofuscación, los usuarios envían más consultas en comparación con la cantidad de consultas que generan y por lo tanto la distancia se hace más grande.

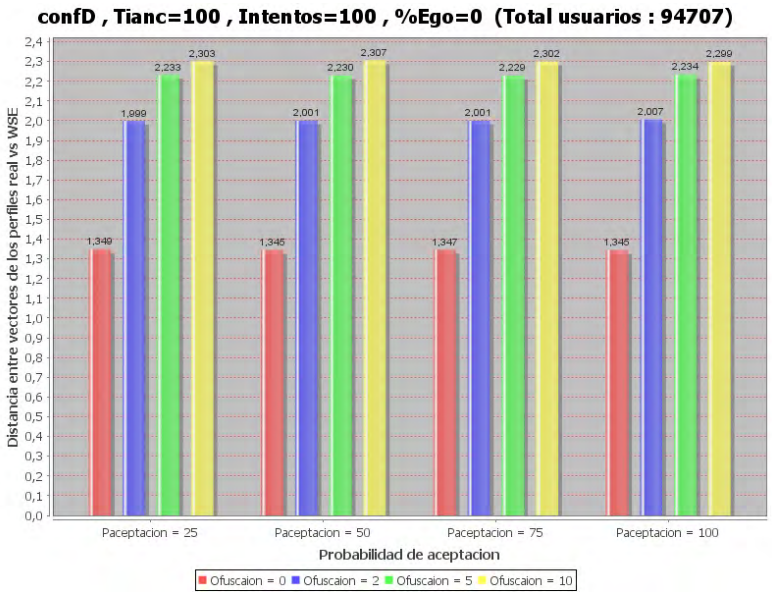

FIGURA 5.16: Distancia entre el perfil real y el perfil ofuscado (Configuración D).

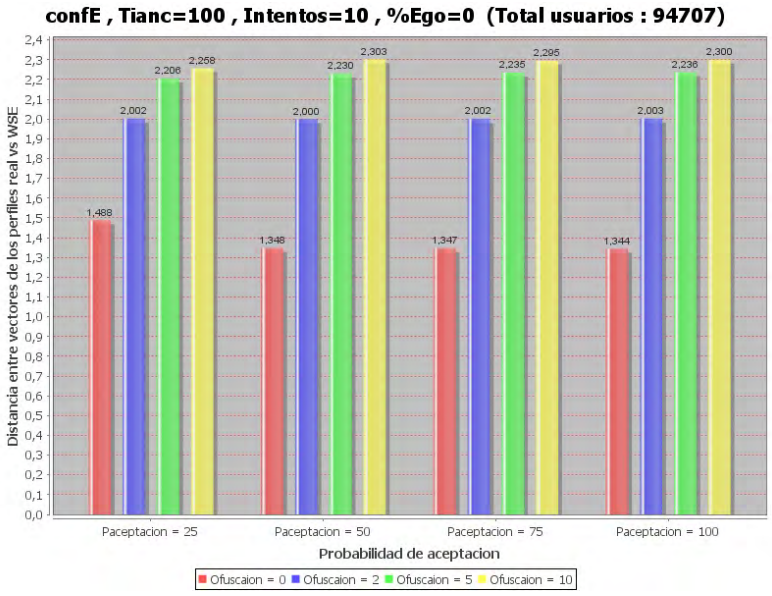

FIGURA 5.17: Distancia entre el perfil real y el perfil ofuscado (Configuración E).

Entre las figuras (5.16 y 5.17) no hay una gran diferencia debido a las consulas se aceptan con menos de 10 intentos gracias al gran número de usuarios que están en el sistema.

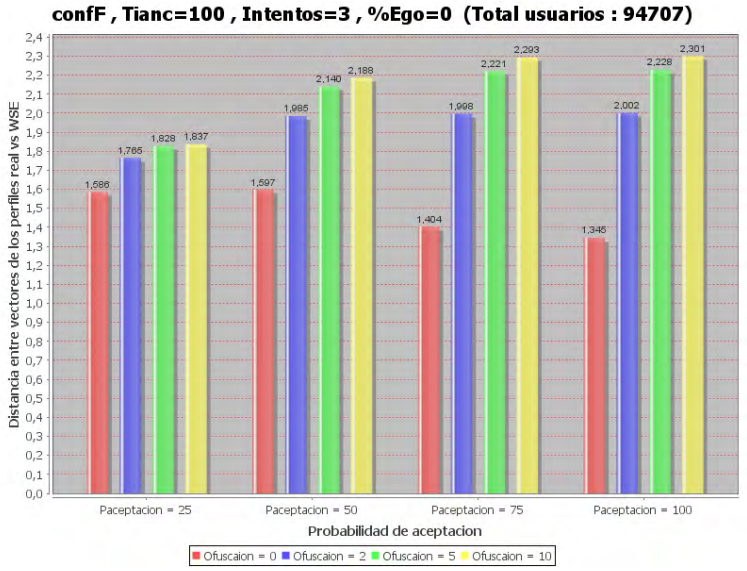

FIGURA 5.18: Distancia entre el perfil real y el perfil ofuscado (Configuración F).

La probabilidad de aceptación es el parámetro que afecta menos al comportamiento del sistema cuando se trata de la distancia entre el perfil real y el perfil ofuscado del usuario por el hecho que haya un gran número de usuarios en el sistema.

		$P_{aceptacion}$				
		25 %	50 %	75 %	100 %	*intentos*
ofuscacion	0	1,34947367	1,34537967	1,34728669	1,34519083	100
		1,48837378	1,34750759	1,34711741	1,34392792	10
		1,58648438	**1,59706869**	1,40355293	1,34487604	3
	2	**1,99914868**	**2,00083536**	**2,00063558**	**2,00663283**	100
		2,00228943	**2,00017594**	**2,00154786**	**2,00276741**	10
		1,76516325	**1,98496865**	**1,9978151**	**2,00187833**	3
	5	2,23312862	2,23014166	2,22862735	2,23435655	100
		2,20553271	2,22966156	2,23468102	2,23631553	10
		1,82836068	2,14032726	2,2209648	2,22809129	3
	10	2,30291239	2,30733727	2,30172264	2,29914996	100
		2,25825779	2,30306892	2,29513339	2,30020399	10
		1,83691948	2,18761861	2,29341777	2,30148582	3

TABLA 5.8: Variación de la distancia entre el perfil real y el perfil ofuscado (Disponibilidad máxima de usuarios y 0 % de egoístas)

Debido a que los valores de la proporción del perfil del usuario conocida por el ***WSE*** son prácticamente iguales, y a que la privacidad del usuario está muy bien protegida, sería interesante identificar los casos donde se recibe mejor calidad de servicio. En la Tabla 5.8, que resume las figuras (5.16, 5.17 y 5.18), se ha fijado un umbral a 2,007 (para limitar los casos y por lo tanto facilitar el análisis en las secciones posteriores); los valores que están por debajo del umbral corresponden a los casos donde los usuarios reciben mejor calidad del servicio. Estos valores están marcados en negrita dentro de las casillas grises.

5.1.2.4. Tiempo de respuesta

Después de analizar el promedio de saltos, la proporción y la distancia para la situación cuando los usuarios pasan mucho tiempo conectados en el sistema, se han determinado 11 casos considerados como más similares al caso ideal. En todos estos casos se mantiene un nivel alto de privacidad y además una calidad aceptable de los servicios del ***WSE***.

Evaluando la tercera necesidad, que es el tiempo, se podrá clasificar estos casos, y por lo tanto deducir la utilidad del sistema bajo estas condiciones.

		$P_{aceptacion}$				
		25 %	50 %	75 %	100 %	*intentos*
ofuscacion	0	16,6919482	10,7089984	8,85574185	7,7186321	100
		12,6010805	10,5595327	8,95452875	7,75756709	10
		6,9495811	**7,24709159**	8,05356878	7,84192228	3
	2	**9,05496756**	**6,23749217**	**5,28607458**	**4,75940536**	100
		8,47601398	**6,31101089**	**5,29050094**	**4,88038913**	10
		6,26678827	**5,5987012**	**5,20530144**	**4,83632504**	3
	5	7,7923621	5,5163724	4,70908136	4,29670291	100
		7,57540183	5,52386663	4,71431561	4,3252909	10
		6,05342173	5,19245298	4,69750553	4,35208725	3
	10	7,47679066	5,33706476	4,52281972	4,22680313	100
		7,42423546	5,26557075	4,52420357	4,23018359	10
		5,97116986	5,03876107	4,47048375	4,18933374	3

Tabla 5.9: Variación del tiempo de respuesta (Disponibilidad máxima de usuarios y 0 % de egoístas)

Según la Tabla 5.9, para estos 11 casos determinados, el tiempo de vida de las consulta está afectado de manera directa por la probabilidad de aceptación. Con menos aceptación se necesitan más intentos para poder enviar la consulta a otro usuario que la acepte.

#	Tiempo	Saltos	Proporción	Distancia	P_{acep}	*Intentos*	*Ofusc.*
1	4,75940536	4,74503817	0,00282097	2,00663283	100	100	2
2	4,83632504	4,8218896	0,0029539	2,00187833	100	3	2
3	4,88038913	4,86581888	0,00322699	2,00276741	100	10	2
4	5,20530144	4,72484732	0,01143215	1,9978151	75	3	2
5	5,28607458	4,79743188	0,00312306	2,00063558	75	100	2
6	5,29050094	4,8019268	0,0029947	2,00154786	75	10	2
7	5,5987012	4,33228705	0,07122511	1,98496865	50	3	2
8	6,23749217	4,80181385	0,00286178	2,00083536	50	100	2
9	6,31101089	4,84944288	0,00338332	2,00017594	50	10	2
10	7,24709159	5,41768433	0,07435257	1,59706869	50	3	0
11	8,47601398	4,79667591	0,00325635	1,99914868	25	100	2

Tabla 5.10: Clasificación de los mejores casos (Disponibilidad máxima de usuarios y 0 % de egoístas)

En Tabla 5.10 se demuestra que para mejorar el nivel de protección de privacidad, la calidad de servició y el tiempo de respuesta, se necesita un factor de ofuscación (> 0), en la situación que los usuarios estén mucho tiempo conectados en el sistema.

5.2. Un sistema con 25 % de usuarios egoístas

En esta sección se analiza el comportamiento del sistema cuando los usuarios egoístas forman el 25 % de la totalidad de los usuarios del sistema. Un usuario egoísta es aquel que rechaza todas las consultas enviadas por otros usuarios hacia él para que las envié al ***WSE*** o para que las reenvié a otros usuarios.

En la presencia de usuarios egoístas, es interesante comparar el comportamiento del sistema para los dos tipos de usuarios (egoístas y normales). De esta manera, se puede comparar los beneficios que se obtienen en cada caso.

En 5.1, se han identificado los mejores casos donde se obtiene un compromiso aceptable entre el nivel de privacidad conseguido y la calidad del servicio obtenida. Los puntos de partida para esta parte son: la Tabla 5.5 que presenta los mejores 11 casos para cuando la disponibilidad de usuarios es mínima y la Tabla 5.10 cuando es máxima.

5.2.1. Disponibilidad mínima de usuarios

Los resultados de la Tabla 5.5 coinciden con los mejores 11 casos donde se consigue un buen compromiso entre el nivel de la privacidad y la calidad de servicio. La Tabla 5.11, presenta los resultados obtenidos en los mismos casos pero con presencia de 23.677 usuarios egoístas que forman el 25 % del total de usuarios que se han conectado al sistema. Los resultados son globales, es decir, se incluyen los dos tipos de usuarios.

#	Tiempo	Saltos	Proporción	Distancia	P_{acep}	*Intentos*	$Ofusc.$
1	6,34929029	4,30621567	0,29020081	1,42935379	50	3	0
2	6,94544853	5,61207708	0,20570215	1,43482051	75	3	0
3	7,05130201	6,46896007	0,19158068	1,44409106	100	3	0
4	7,30723763	6,68059367	0,18943796	1,45473998	100	10	0
5	7,53735041	6,62217565	0,18965679	1,44788186	100	100	0
6	8,34095996	6,69778393	0,19316536	1,45173622	75	10	0
7	8,59474123	6,69072403	0,18979818	1,4538296	75	100	0
8	10,1271558	6,55148973	0,18931848	1,45033823	50	10	0
9	10,5023029	6,56791019	0,19251208	1,44645563	50	100	0
10	11,7115755	5,09686879	0,22978839	1,43893803	25	10	0
11	16,7529213	6,66931046	0,18796165	1,45388881	25	100	0

TABLA 5.11: Clasificación de los mejores casos (Disponibilidad mínima de usuarios y 25 % de egoístas)

Según la Tabla 5.11, se obtienen mejores resultados cuando se hacen menos intentos, y esto es una de las influencias de la presencia de los usuarios egoístas en el sistema. Cuando el parámetro de ofuscación está fijado en 0, los usuarios llegan más rápido a completar el perfil en este caso porque sólo el 75 % de los usuarios se consideran activos y envían consultas al ***WSE***, el resto

(25 % que son egoístas) sólo envían las consultas propias que por alguna razón no han podido ser atendidas por otros usuarios (hay menos de k usuarios o se ha alcanzado el máximo de intentos sin qua la acepte nadie). Estos 11 casos comparten valores similares de distancia, dentro del rango $\{1{,}42, 1{,}45\}$, que en comparación con la Tabla 5.5 se puede decir que se gana calidad de servicio ya que los valores en este caso son inferiores. Por otro lado, han incrementado los valores de la proporción del perfil conocida por el ***WSE*** y esto es una otra influencia de la presencia de los usuarios egoístas; muchas consultas se acaban enviando al ***WSE*** por los usuarios que las han generado, lo que permite al ***WSE*** obtener más informacioón sobre el perfil real del usuario. Se puede observar también un incremento a nivel del promedio de saltos que le corresponde un incremento en el tiempo de respuesta; después de completar el perfil ofuscado, los usuarios normales siguen aceptando las consultas pero las reenvían a otros en vez de enviarlas al ***WSE***. El tiempo de respuesta se afecta en primer lugar por la ofuscación, y luego por el número de intentos y la probabilidad de aceptación.

la Tabla 5.12 presenta una mejora de los resultados de la Tabla 5.11 incrementando el valor de la ofuscación a 2.

#	Tiempo	Saltos	Proporción	Distancia	P_{acep}	$Intentos$	$Ofusc.$
1	4,50409114	4,03527455	0,1936391	1,78729577	100	3	2
2	4,58749702	4,09067448	0,18436435	1,80365385	100	10	2
3	4,79850193	3,91515047	0,21910023	1,76994236	75	3	2
4	4,85970203	4,14129416	0,18235033	1,80227151	100	100	2
5	5,03812626	4,09938687	0,18477624	1,80039761	75	10	2
6	5,1842255	3,62985267	0,29403578	1,6916963	50	3	2
7	5,21590948	4,07542011	0,18562138	1,79786832	75	100	2
8	5,86595056	4,06168024	0,18901915	1,79300422	50	10	2
9	6,32437189	4,21127922	0,17418239	1,81686739	50	100	2
10	7,86725013	3,84469233	0,23675745	1,75852761	25	10	2
11	8,75679907	4,07346688	0,18511141	1,79406937	25	100	2

Tabla 5.12: Mejora de los casos (Disponibilidad mínima de usuarios y 25 % de egoístas)

Incrementar la ofuscación permite que los usuarios completen sus perfiles durante un tiempo relativamente largo en comparación con el caso anterior. Los mejores resultados se obtienen para una probabilidad de aceptación relativamente alta ($\geq 75\,\%$), y para un número de intento relativamente bajo (≤ 10).

Sería interesante también ver el comportamiento del sistema en particular para usuarios normales y usuarios egoístas por separado. A nivel de privacidad, los usuarios egoístas aprovechan que sus consultan estén enviadas al ***WSE*** por otros usuarios, lo que hace que la privacidad esté más protegida en comparación con los usuarios normales. El ***WSE*** conoce hasta el 21 % del perfil real de los usuarios normales en el peor caso mientras conoce solamente el 18 % del perfil real de los usuarios egoístas. A nivel de calidad de servicio, el hecho que los usuarios egoístas sólo envían un porcentaje bajo de sus consultas al ***WSE*** hace que reciban menos calidad de servicio

en comparación con lo que reciben los usuarios normales. Las consultas de los usuarios normales hacen menos saltos y por lo tanto están menos tiempo en el sistema.

La explicación anterior se ilustra gráficamente en la Figura 5.19 (A la izquierda los usuarios normales, y a la derecha los usuarios egoístas).

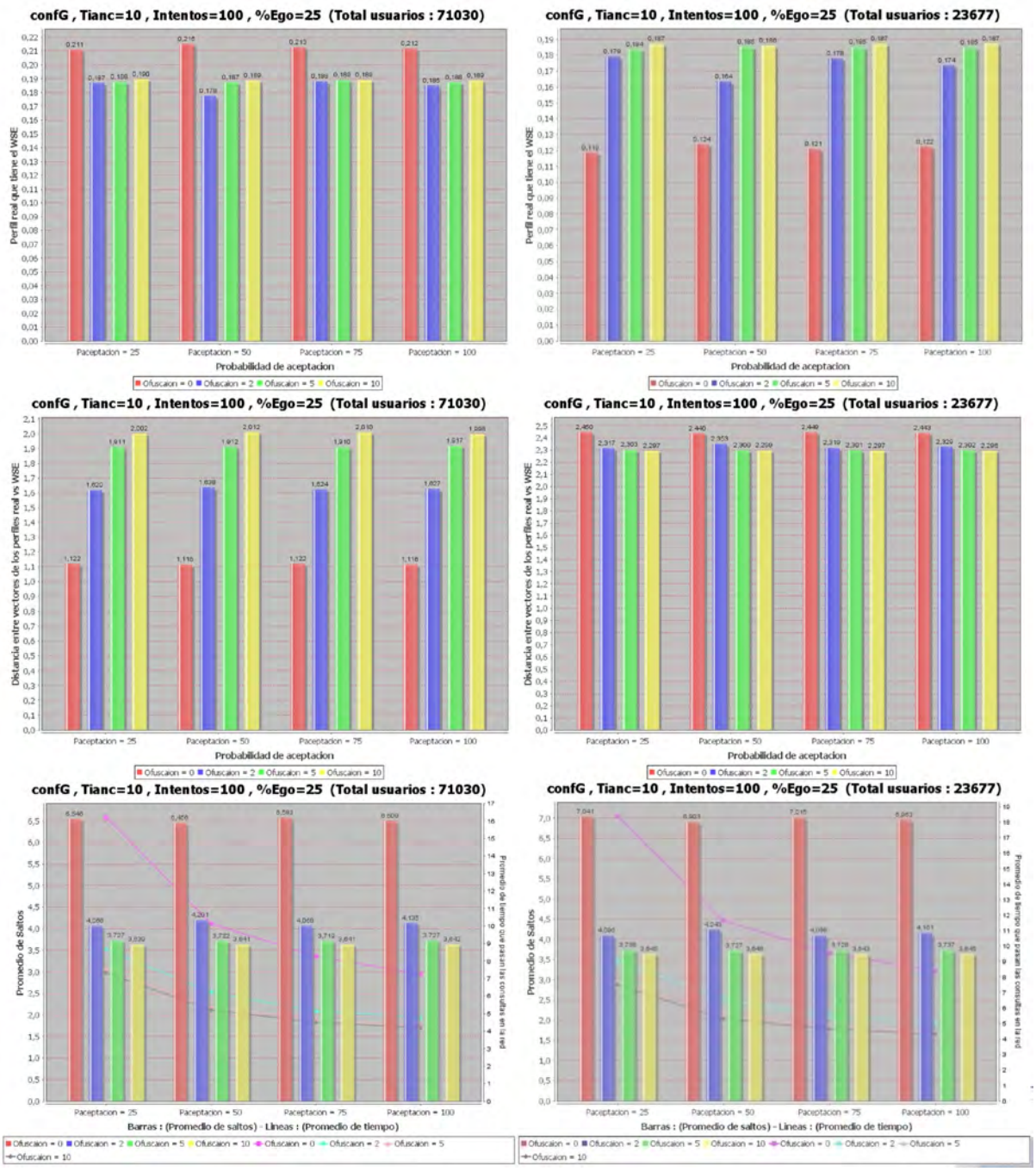

FIGURA 5.19: Comparación del comportamiento del sistema para usuarios normales y egoístas (mínima disponibilidad de usuarios y 25 % de egoístas).

5.2.2. Disponibilidad máxima de usuarios

De manera similar al punto anterior, el estudio que se ha realizado para esta parte se basa sobre los resultados obtenidos en la misma situación para un sistema sin egoístas. Se ha analizado la influencia de la presencia de egoístas basándose sobre la Tabla 5.10. Los resultados obtenidos se muestran en la Tabla 5.13.

#	Tiempo	Saltos	Proporción	Distancia	P_{acep}	*Intentos*	$Ofusc.$
1	6,42985971	4,61037078	0,10310749	2,22539642	50	3	2
2	6,54843024	6,29845228	0,00775143	2,25133098	100	100	2
3	6,58562185	6,33333632	0,00759801	2,25320298	100	10	2
4	6,64068195	6,38686338	0,00851582	2,25405951	100	3	2
5	6,68122684	5,71908612	0,0249263	2,26536335	75	3	2
6	7,30141154	6,24214113	0,00771325	2,25682527	75	10	2
7	7,41370746	6,33854343	0,00716006	2,25360248	75	100	2
8	8,6119068	6,00726397	0,10025487	1,94185187	50	3	0
9	9,03304355	6,33321119	0,00749356	2,25311851	50	100	2
10	9,16327897	6,42396321	0,00904238	2,254316	50	10	2
11	11,2295271	5,29743159	0,04808834	2,26111615	25	10	2

TABLA 5.13: Clasificación de los mejores casos (Disponibilidad máxima de usuarios y 25 % de egoístas)

Los nuevo resultados muestran que todavía se mantiene un compromiso muy aceptabe entre el nivel de privacidad y la calidad del servicio. La distancia entre el perfil real y perfil ofuscado del usuario tiene como promedio 2,25 mientras que estaba alrededor de 2 con respecto al sistema sin egoístas en las mismas condiciones. La proporción del perfil conocida por el ***WSE*** no se ve muy afectada por la presencia de egoístas (10 % mientras que estaba a 7 % en el peor caso). Se observa también un incremento tanto en el promedio de saltos que en el promedio de tiempo de respuesta. Generalmente, los mejores casos que han seguido ofreciendo un buen compromiso entre la privacidad y la calidad de servicio son los que tienen alta probabilidad de aceptación (≥ 75) y pocos intentos (≤ 10).

Aunque no hay una gran diferencia en general entre la Tabla 5.10 (sistemas sin egoístas) y la Tabla 5.13 (sistema con 25 % de egoístas), el incremento relativo en todos los valores de distancia, proporción, saltos y tiempo en la Tabla 5.10 con respecto a la Tabla 5.13 puede afectar tanto el nivel de privacidad que la calidad de servicio. Para la situación donde hay una presencia de egoístas hasta 25 %, con disponibilidad máxima de usuarios, los resultados se pueden mejorar incrementando el parámetro de ofuscación a 5.

La Tabla 5.14 muestra los resultados obtenidos al incrementar el parámetro de ofuscación a 5. Generalmente, los resultados son parecidos a los de la Tabla 5.10 (mismas condiciones pero sin egoístas) excepto una variación relativa en el promedio de distancia que está alrededor de 2,50 y esto es debido al incremento del parámetro de ofuscación. Para tener una idea, la distancia

igual a 2,50 es equivalente a enviar ±5 consultas de diferentes categorías o enviar ±2 consultas de la misma categoría. En este caso, el ***WSE*** sigue teniendo un perfil muy parecido al perfil real del usuario aunque la proporción que el ***WSE*** conoce del perfil real es de menos de 1 % en los mejores casos y de menos de 10 % en los peores casos.

Se consigue también un buen promedio de saltos que le corresponde un tiempo de respuesta muy razonable en la mayoría de los casos.

#	Tiempo	Saltos	Proporción	Distancia	P_{acep}	$Intentos$	$Ofusc.$
1	5,02580571	4,81348925	0,00243811	2,54653281	100	10	5
2	5,06750533	4,8529874	0,00442706	2,53863249	100	3	5
3	5,12079388	4,90720297	0,00301332	2,54337667	100	100	5
4	5,45245995	4,69942314	0,02511425	2,50771323	75	3	5
5	5,61065864	4,8500935	0,00304413	2,54583009	75	10	5
6	5,61478164	4,8488652	0,00284752	2,5407279	75	100	5
7	5,72848942	4,16550475	0,1054953	2,37976904	50	3	5
8	6,68737522	4,83627126	0,00301947	2,54663097	50	100	5
9	6,70353745	4,84464788	0,00462678	2,54511451	50	10	5
10	8,91825378	4,42812447	0,04937791	2,47434666	25	10	5
11	10,1590407	4,91286841	0,00317809	2,54029205	25	100	5

Tabla 5.14: Mejora de los casos (Disponibilidad máxima de usuarios y 25 % de egoístas)

A continuación, se analiza el caso 3 de la Tabla 5.14 para comparar el comportamiento del sistema en particular para usuarios normales y para usuarios egoístas por separado. A nivel de privacidad, los usuarios egoístas aprovechan que sus consultan estén enviadas al ***WSE*** por otros usuarios, y por lo tanto su privacidad está protegida mejor en comparación con los usuarios normales. El ***WSE*** conoce hasta el 3 % del perfil real de los usuarios normales en el peor caso mientras conoce solamente el 0,1 % del perfil real de los usuarios egoístas. A nivel de calidad de servicio, el hecho que los usuarios egoístas sólo envían un porcentaje bajo de sus consultas al ***WSE*** hace que reciban menos calidad de servicio en comparación con lo que reciben los usuarios normales. Las consultas de los usuarios normales hacen relativamente menos saltos y por lo tanto están poco tiempo en el sistema.

La explicación anterior se ilustra gráficamente en la Figura 5.20. (A la izquierda los usuarios normales, y a la derecha los usuarios egoístas).

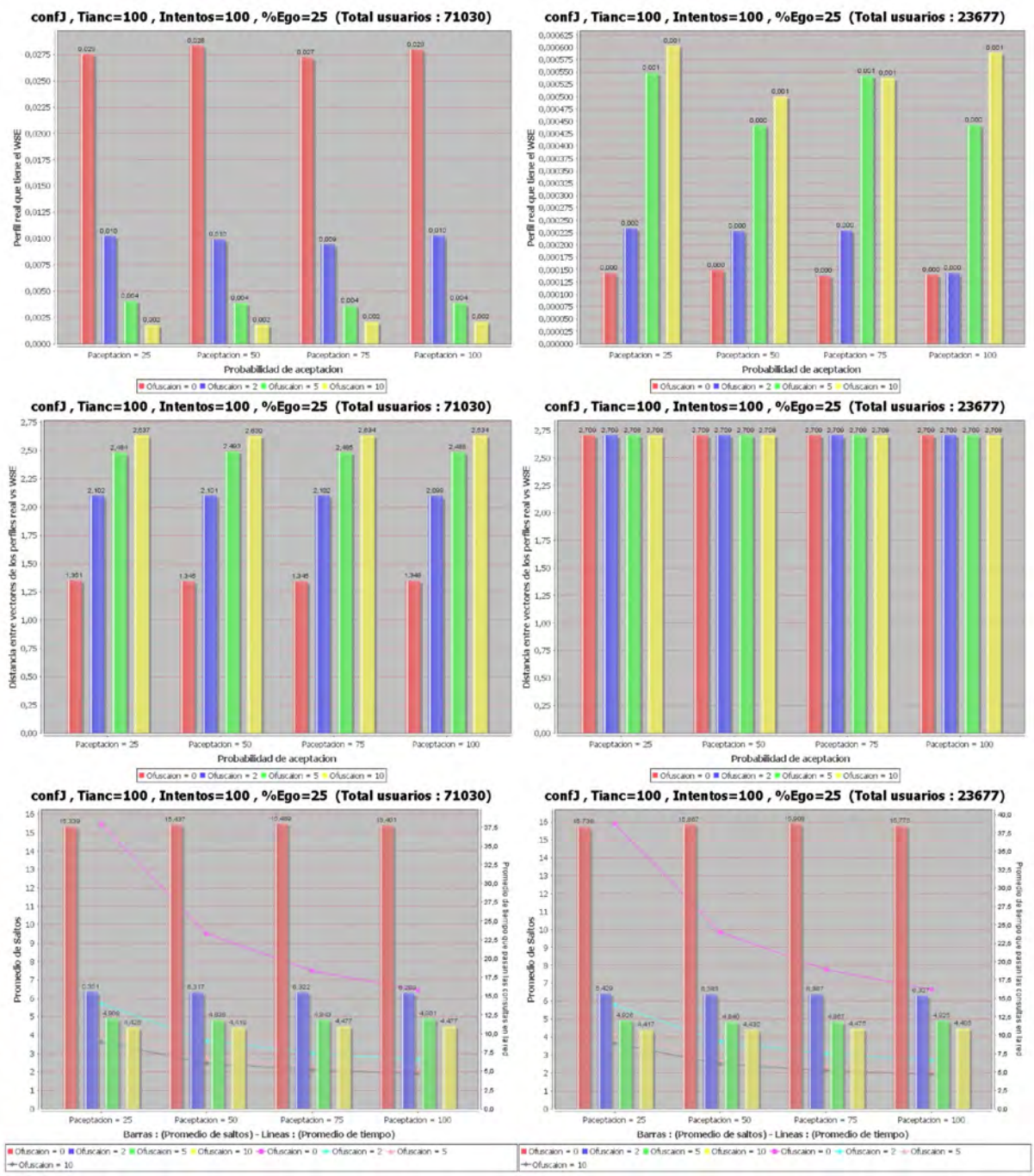

FIGURA 5.20: Comparación del comportamiento del sistema para usuarios normales y egoístas (máxima disponibilidad de usuarios y 25 % de egoístas).

5.3. Un sistema con 50 % de usuarios egoístas

En esta sección se analiza el comportamiento del sistema cuando la mitad de los usuarios conectados al sistema son egoístas.

En la sección anterior se ha visto que cuando el 25 % de los usuarios del sistema son egoístas, el sistema sigue dando solución para alcanzar un buen compromiso entre el nivel de protección de la privacidad y la calidad de servicio. Cambiando el parámetro de ofuscación de 0 a 2 en el caso de disponibilidad mínima y de 2 a 5 en el caso de disponibilidad máxima, los usuarios siguen teniendo un buen nivel de privacidad y buena calidad de servicio.

5.3.1. Disponibilidad mínima de usuarios

Para llevar a cabo el estudio de los resultados en este caso, se ha basado en los resultados de la Tabla 5.12. Cuando se conectan más usuarios egoístas en el sistema, las consultas empiezan a tener menos probabilidad para que estén aceptadas. En este caso se tienen que hacer más intentos para encontrar usuarios que acepten las consultas pero esto tiene un coste a nivel de tiempo. La Tabla 5.15 es una actualización de la Tabla 5.12 a nivel del porcentaje de usuarios egoístas que están conectados al sistema.

#	Tiempo	Saltos	Proporción	Distancia	P_{acep}	*Intentos*	$Ofusc.$
1	5,92260495	4,33392431	0,3715828	1,97991408	50	3	2
2	5,98633885	4,33392431	0,26352592	2,21565867	75	3	2
3	6,6182479	5,54276822	0,19893205	2,35966118	100	3	2
4	7,62245858	6,33128122	0,17483892	2,45858496	100	10	2
5	8,76930829	6,36707321	0,17534026	2,45449246	75	10	2
6	8,81169844	6,25193101	0,17821353	2,45734261	100	100	2
7	9,96043084	6,35967951	0,17536958	2,46157445	75	100	2
8	10,1627625	5,88381603	0,18967458	2,40590959	50	10	2
9	11,0357278	4,29462238	0,26671633	2,2044143	25	10	2
10	12,4580329	6,50765156	0,17098585	2,47234413	50	100	2
11	18,9408983	6,28599999	0,17647163	2,45664753	25	100	2

Tabla 5.15: Clasificación de los casos (Disponibilidad mínima de usuarios y 50 % de egoístas)

El tiempo de respuesta depende en primer lugar del número de intentos que se hacen para encontrar usuarios que acepten las consultas. Es posible ver que a medida que el número de intentos va subiendo y la probabilidad de aceptación va bajando, se necesita más tiempo para encontrar usuarios que acepten las consultas. En general, se mantiene un nivel aceptable de privacidad (37 % del perfil real conocido por ***WSE*** en el peor caso) y una buena calidad de servicio (2,47 como peor distancia). Aunque hay un incremento relativo a nivel de saltos, se necesita más tiempo para tener la respuesta. Para mejorar los resultados, el incremento del parámetro de ofuscación de 2 a 5 ofrece una compensación a nivel de tiempo, saltos y proporción

pero con una subida relativa a nivel de distancia. La Tabla 5.16 muestra los nuevos resultados, donde se puede ver que los mejores casos son aquellos donde se hacen menos saltos y tienen una probabilidad de aceptación alta.

#	Tiempo	Saltos	Proporción	Distancia	P_{acep}	$Intentos$	$Ofusc.$
1	4,96597886	4,06395684	0,20877138	2,47997201	100	3	5
2	5,4181497	4,35113063	0,17799213	2,58928222	100	10	5
3	5,90974733	4,23109741	0,18526591	2,56537276	75	10	5
4	6,77729317	4,33393723	0,17567412	2,59605768	100	100	5
5	7,0176809	4,17360336	0,19300211	2,53701235	50	10	5
6	7,44563894	4,41672348	0,17411674	2,60149363	75	100	5
7	8,51095868	4,11470802	0,18951819	2,55788901	50	100	5
8	12,259637	4,16866477	0,18679024	2,5669283	25	100	5

TABLA 5.16: Mejora de los casos (Disponibilidad mínima de usuarios y 50 % de egoístas)

Cuando la mitad de usuarios son egoístas, el sistema comporta diferente dependiendo del tipo del usuario. A nivel de privacidad, con más ofuscación los usuarios normales tienen más privacidad, en cambio los usuarios egoístas pierden privacidad cuando tienen que ofuscar. El comportamiento de esta manera es debido a que cuando el parámetro de ofuscación es relativamente pequeño los usuarios normales llegan rápido a completar el perfil enviando consultas propias al ***WSE***, debido al gran porcentaje del rechazo sobre todo cuando la probabilidad de aceptación es pequeña. A nivel de calidad de servicio, pasa el contrario; con más ofuscación los usuarios egoístas tienen más calidad de servicio ya que se encuentran obligados a enviar sus propias consultas al ***WSE***. A nivel de saltos, las consultas que pertenecen a los usuarios egoístas siempre hacen un salto de más en la mayoría de los casos. Este comportamiento va desapareciendo a medida que el número de los intentos va disminuyendo. La Figura 5.21, es el mejor ejemplo donde se puede ver este comportamiento con más claridad (A la izquierda los usuarios normales, y a la derecha los usuarios egoístas).

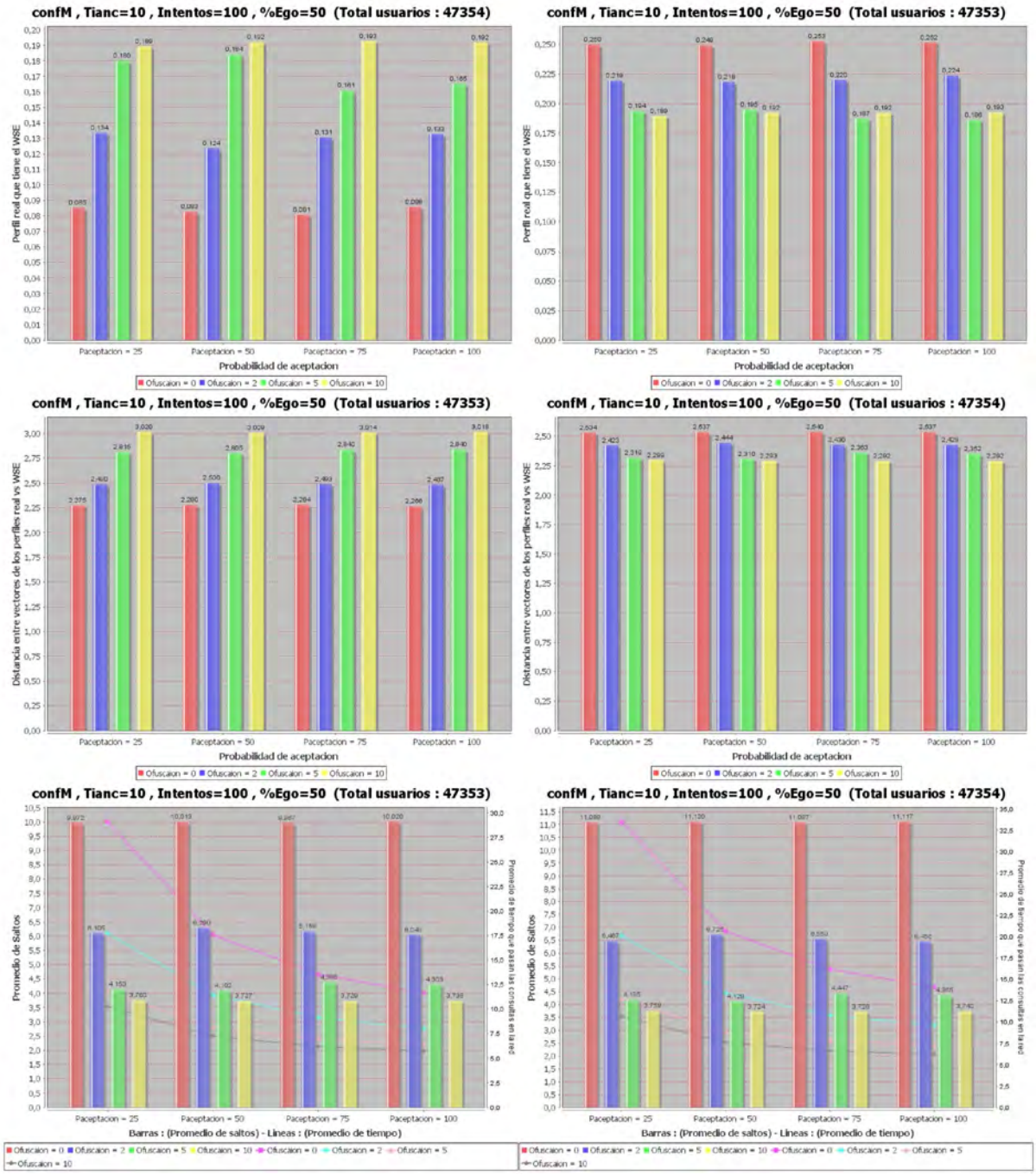

FIGURA 5.21: Comparación del comportamiento del sistema para usuarios normales y egoístas (mínima disponibilidad de usuarios y 50 % de egoístas).

5.3.2. Disponibilidad máxima de usuarios

En esta parte, el estudio realizado se basa sobre la Tabla 5.14. El sistema sigue ofreciendo buenos resultados para cuando los usuarios ofuscan con 5 consultas en la presencia del 50 % de usuarios egoístas. Tal y como se puede ver en la Tabla 5.17 los resultados son aceptables tanto a nivel de privacidad que a nivel de calidad de servicio. Los mejores resultados se obtienen cuando se hacen menos intentos, de esta manera se mejora mucho el tiempo de respuesta ya que en la presencia de los usuarios egoístas la probabilidad del rechazo sube.

#	Tiempo	Saltos	Proporción	Distancia	P_{acep}	$Intentos$	$Ofusc.$
1	6,72676001	4,4504656	0,14810172	2,6938314	50	3	5
2	7,23436329	5,84003631	0,04234856	2,94948346	75	3	5
3	7,53018135	6,94505595	0,01152059	3,01965029	100	3	5
4	7,741129	7,15653466	0,00732858	3,03204282	100	10	5
5	7,81532798	7,21798441	0,00765251	3,03284393	100	100	5
6	8,76365159	7,10632347	0,00759741	3,03071133	75	100	5
7	8,92633739	7,25668819	0,00782728	3,02851278	75	10	5
8	10,5521164	6,89452922	0,01033992	3,02096772	50	10	5
9	11,1961409	7,31237228	0,0076692	3,0289666	50	100	5
10	12,4230113	5,31832823	0,0672816	2,89135679	25	10	5
11	17,4779196	7,18705454	0,0073724	3,02956303	25	100	5

Tabla 5.17: Clasificación de los mejores casos (Disponibilidad máxima de usuarios y 50 % de egoístas)

Para mejorar los resultados de la Tabla 5.17 a nivel del tiempo de respuesta y el promedio de salto, sólo hay que incrementar el parámetro de ofuscación. La Tabla 5.18 muestra las mejoras cuando se utiliza un parámetro de ofuscación igual a 10. Es posible observar que en la presencia de egoístas siempre es mejor hacer menos intentos.

#	Tiempo	Saltos	Proporción	Distancia	P_{acep}	$Intentos$	$Ofusc.$
1	5,71715926	5,21200116	0,01141292	3,14362389	100	3	10
2	5,75067475	5,24089492	0,0033017	3,16442072	100	10	10
3	5,97227887	5,45341073	0,00396593	3,16338903	100	100	10
4	6,04539753	4,88041515	0,04694603	3,03492265	75	3	10
5	6,147621	4,09716582	0,15455321	2,729803	50	3	10
6	6,79356472	5,51566525	0,00402088	3,15904064	75	10	10
7	6,82258975	5,53121835	0,00434658	3,15883793	75	100	10
8	8,05389138	5,34696497	0,00785368	3,14948434	50	10	10
9	8,17016946	5,41161662	0,00380178	3,15942567	50	100	10
10	10,389717	4,57428542	0,07231074	2,96182419	25	10	10
11	12,5791044	5,39678234	0,00363342	3,16525383	25	100	10

Tabla 5.18: Mejora de los casos (Disponibilidad máxima de usuarios y 50 % de egoístas)

En la comparación entre el comportamiento del sistema para los usuarios egoístas y los usuarios normales por separado, los egoístas obtienen más privacidad y mejor calidad de servicio por el hecho que sus consultas se envían mediante los usuarios normales (privacidad) y sólo les toca

enviar sus propias consultas al ***WSE*** (calidad de servicio). Pero esto no quiere decir que los usuarios normales pierden la privacidad o la calidad de servicio, ya que ellos también obtienen un compromiso aceptable cuando se hacen muchos intentos para reenviar consultas. Cuando se hacen pocos intentos, el comportamiento del sistema es casi idéntico para los dos tipos de usuarios. La Figura 5.22 muestra el comportamiento del sistema para ambos usuarios en el caso de 3 intentos.

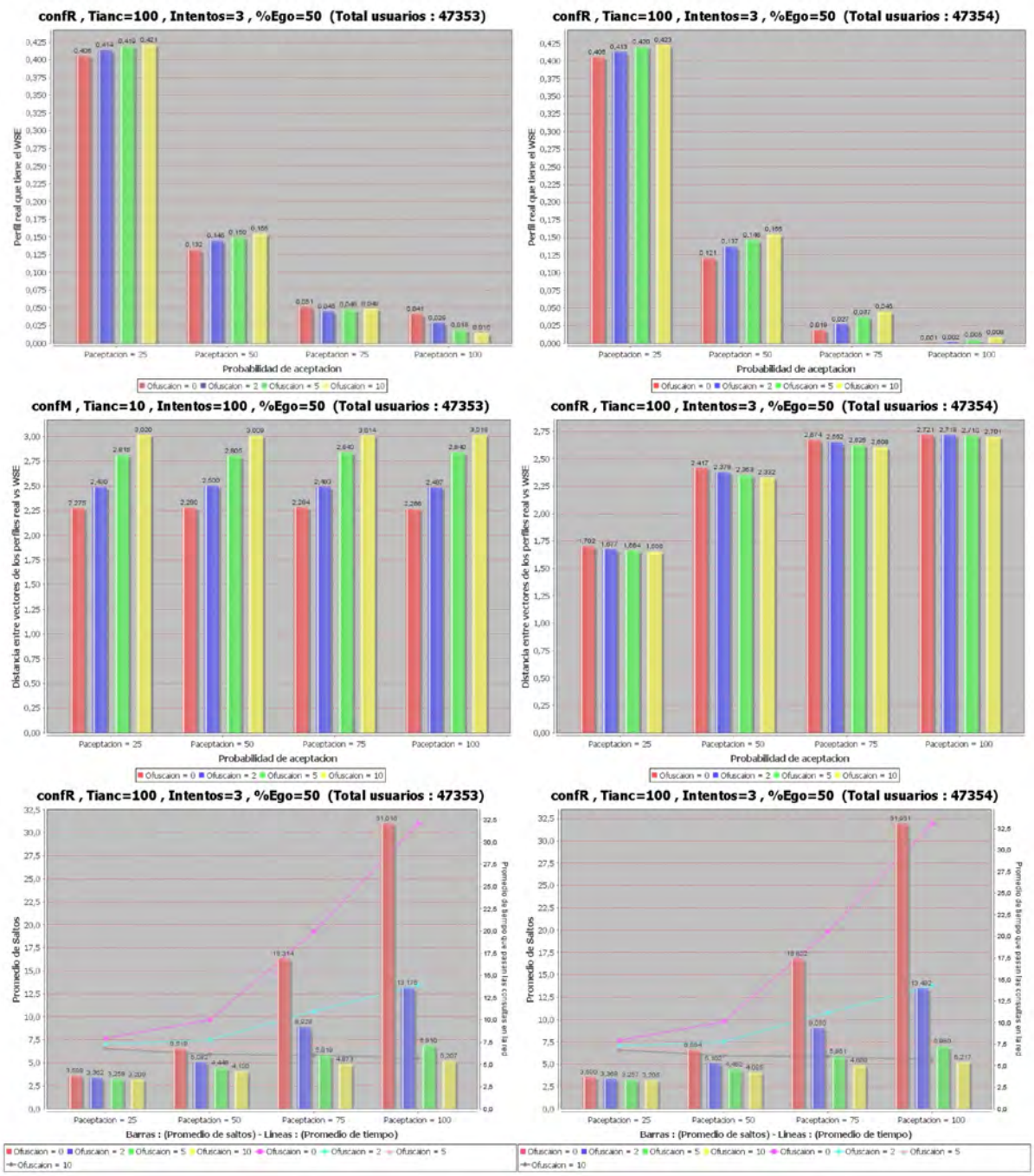

FIGURA 5.22: Comparación del comportamiento del sistema para usuarios normales y egoístas (máxima disponibilidad de usuarios y 50 % de egoístas).

5.4. Un sistema con 75 % de usuarios egoístas

En esta sección se analiza el comportamiento cuando la mayoría del los usuarios son egoístas, se ha valorado la utilidad del sistema en los dos casos de disponibilidad de usuarios.

5.4.1. Disponibilidad mínima de usuarios

Anteriormente, se ha visto que a medida que el número de egoístas conectados en el sistema incrementaba, el sistema seguía ofreciendo buen compromiso entre el nivel de protección de la privacidad y la calidad del servicio incrementando el valor del parámetro de ofuscación. Los mejores resultados en esta situación se han obtenido con el parámetro de ofuscación igual a 10. La Tabla 5.19 muestra los resultados obtenidos.

#	Tiempo	Saltos	Proporción	Distancia	P_{acep}	*Intentos*	$Ofusc.$
1	6,32784784	4,60219133	0,24757681	3,05422865	100	3	10
2	8,23485417	5,82731334	0,17677539	3,4301402	100	10	10
3	9,20097142	5,63542023	0,18987436	3,37100968	75	10	10
4	10,4903461	5,10387311	0,22107796	3,22246835	50	10	10
5	12,4379923	6,00168928	0,18232839	3,42480436	100	100	10
6	13,7510117	5,75046477	0,1772299	3,44870988	75	100	10
7	16,3355258	5,91142138	0,17670207	3,43963547	50	100	10
8	24,3402697	6,02789822	0,17535987	3,44083204	25	100	10

Tabla 5.19: Clasificación de los casos (Disponibilidad mínima de usuarios y 75 % de egoístas)

Según la Tabla 5.19, el tiempo de respuesta se ve afectado por el número de intentos en primer lugar, y después por la probabilidad de aceptación. Se sigue teniendo un promedio aceptable de saltos y una distancia razonable, pero el nivel de privacidad se ve afectado y el ***WSE*** llega a conocer hasta el 25 % del perfil real del usuario en el peor caso. A nivel de tiempo, los resultados obtenidos no se pueden considerar aceptables cuando se hacen muchos intentos. El comportamiento del sistema se puede justificar por el gran número de usuarios egoístas que hace que las consultas se rechacen con una probabilidad muy alta, y por lo tanto los usuarios se encuentran obligados a enviar sus propias consultas al ***WSE***. Cuando se requieren muchos intentos para encontrar usuarios para poder reenviar la consulta, se necesita más tiempo para tener la respuesta.

El comportamiento del sistema en esta situación va a favor de los usuarios egoístas que obtienen mejor calidad de servicio y mejor privacidad. Esto es debido a que sólo envían al ***WSE*** las consultas propias que se rechazan por otros usuarios. Los usuarios normales obtienen mejor privacidad ofuscando más pero perdiendo calidad de servicio. La Figura 5.23 es una comparación entre el comportamiento del sistema para usuarios normales y usuarios egoístas en el caso que se permiten 10 intentos (A la izquierda los usuarios normales, y a la derecha los usuarios egoístas).

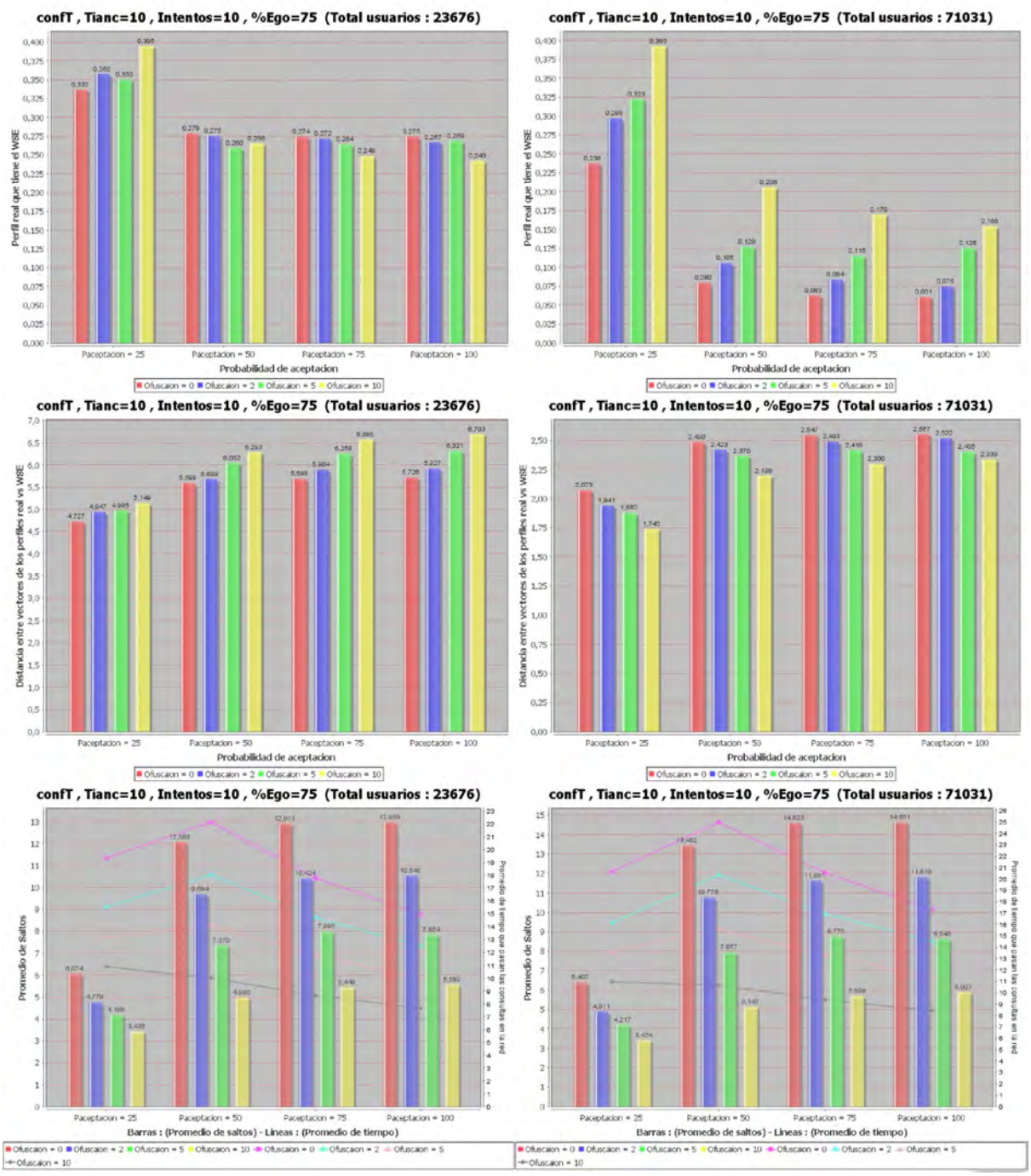

FIGURA 5.23: Comparación del comportamiento del sistema para usuarios normales y egoístas (mínima disponibilidad de usuarios y 75 % de egoístas).

5.4.2. Disponibilidad máxima de usuarios

Con más tiempo que los usuarios estén conectados en el sistema, menos oportunidades tienen para que las consultas propias estén enviadas al ***WSE*** por otros. Las simulaciones que se han hecho para esta parte muestran que los mejores resultados se obtienen cuando se hacen pocos intentos (≤ 3) con probabilidad de aceptación igual a 50 % y para parámetro de ofuscación (≥ 2). La Tabla 5.20 muestra los resultados obtenidos.

#	Tiempo	Saltos	Proporción	Distancia	P_{acep}	$Intentos$	$Ofusc.$
1	7,48803684	4,42129328	0,21405493	3,04534343	50	3	10
2	8,37967513	4,96436567	0,19638191	3,03931198	50	3	5
3	9,61230798	5,70980596	0,17910327	3,02823844	50	3	2

TABLA 5.20: Clasificación de los mejores casos (Disponibilidad máxima de usuarios y 75 % de egoístas)

Que la probabilidad de aceptación esté al 50 %, esto quiere decir que los usuarios normales rechazan también una parte de consultas que reciben, cosa que da un equilibrio al comportamiento del sistema en este caso. Con menos intentos se mejora el tiempo de respuesta y el promedio de saltos.

En general, las distancias obtenidas son aceptables pero el ***WSE*** conoce una proporción interesante del perfil real del usuario.

Los usuarios egoístas han obtenido mejor calidad de servicio en esta situación, por el hecho que sólo envían las consultas propias al ***WSE***. En la Figura 5.24 se hace una comparación del comportamiento del sistema para usuarios normales y egoístas (A la izquierda los usuarios normales, y a la derecha los usuarios egoístas).

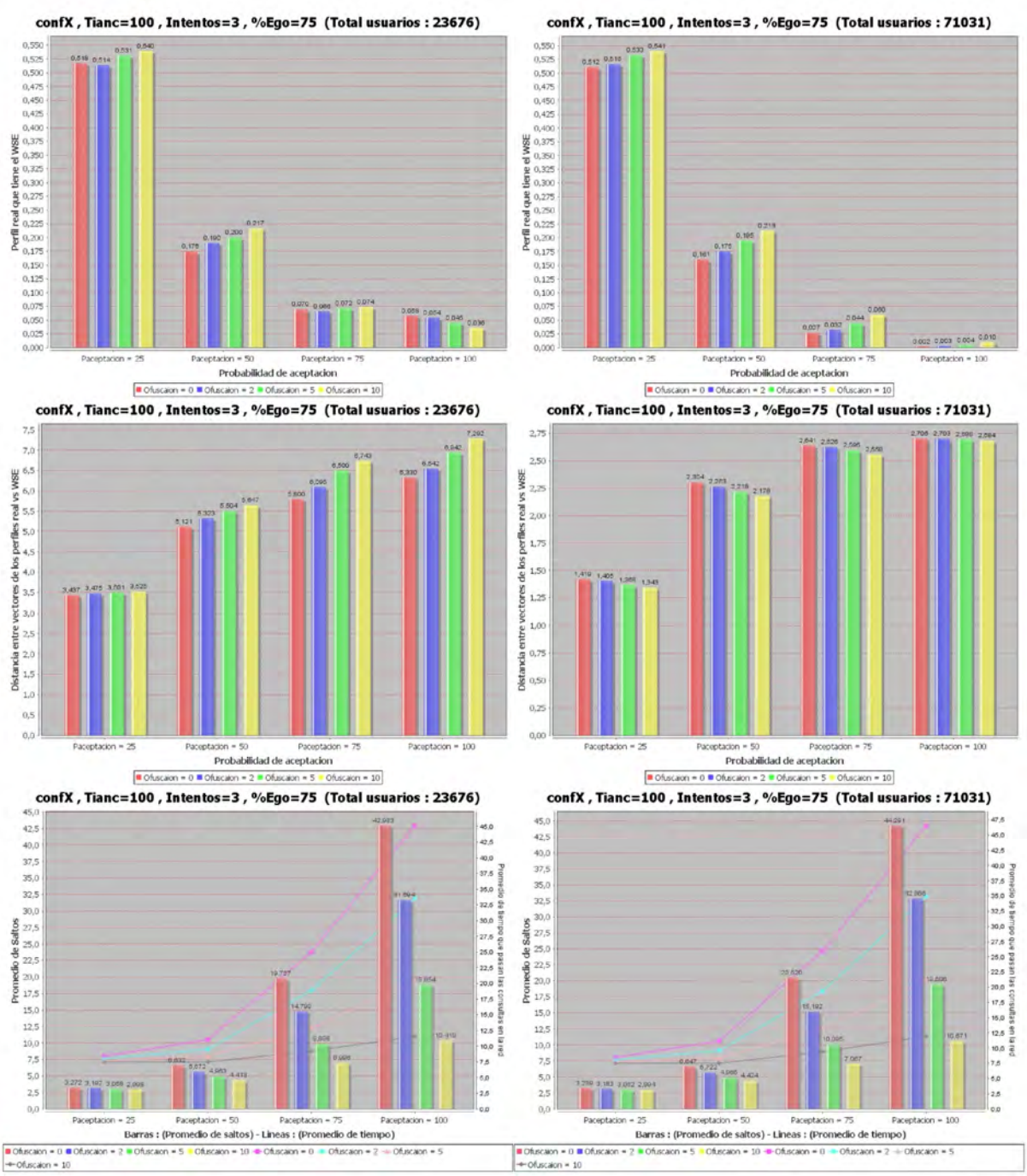

FIGURA 5.24: Comparación del comportamiento del sistema para usuarios normales y egoístas (máxima disponibilidad de usuarios y 75 % de egoístas).

5.5. Un sistema con 100 % de usuarios egoístas

Cuando todos los usuarios son egoístas, se rechazan todas las consultas que se quieren reenviar entre los usuarios, lo que hace que la consulta se envía finalmente al ***WSE*** por el mismo usuario que la ha generado. Para este caso, se obtiene mejor calidad de servicio (Figura 5.25) y el mínimo de saltos, pero sin privacidad ya que el ***WSE*** conoce la totalidad del perfil del usuario (Figura 5.26).

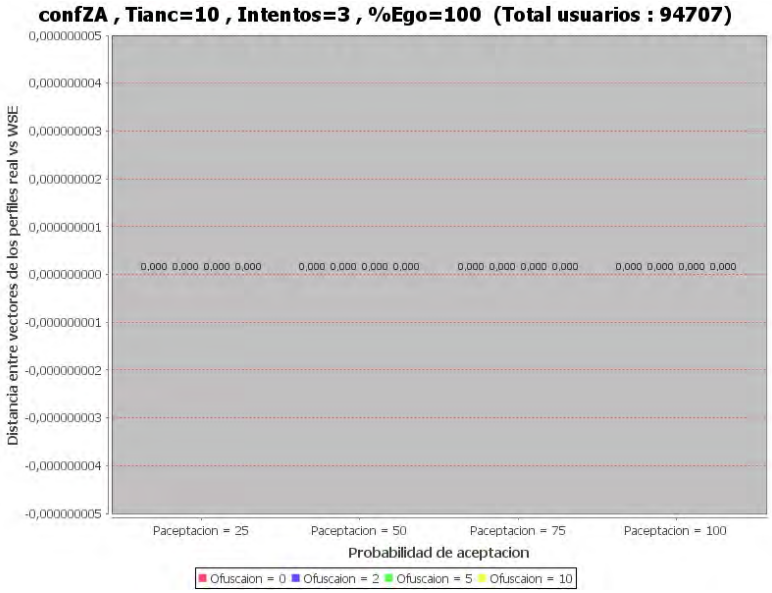

FIGURA 5.25: Distancia entre el perfil real y el perfil ofuscado (Configuración ZA).

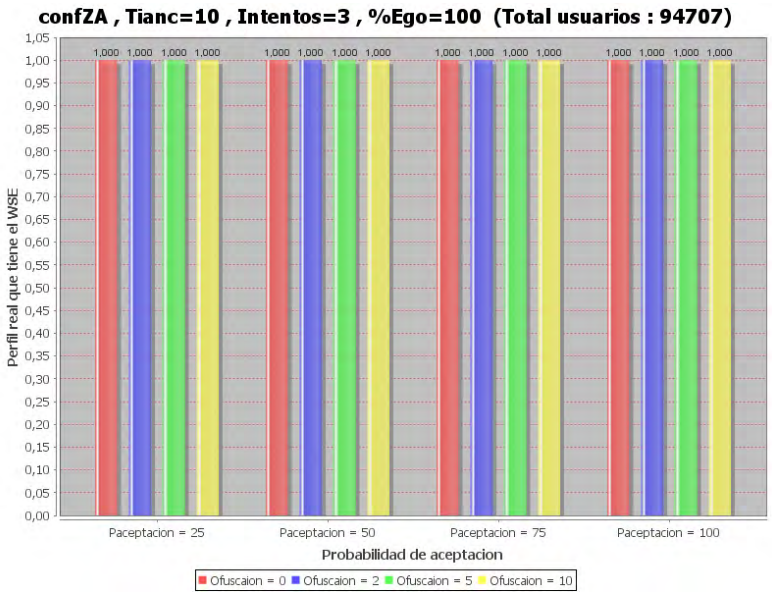

FIGURA 5.26: Proporción del perfil real conocida por el WSE (Configuración ZA).

Capítulo 6

Conclusiones y trabajo futuro

6.1. Conclusiones

La gran cantidad de datos que existe en Internet requiere herramientas de búsqueda que sean eficientes a nivel del contenido devuelto como resultado y a nivel de tiempo de respuesta. Los ***WSE***s cumplen este requerimiento mediante la creación de perfiles de usuarios ofreciendo de esta manera servicios avanzados y resultados personalizados a favor de los usuarios. La información que contiene el perfil del usuario almacenado en el ***WSE*** puede ser una verdadera amenaza para su privacidad.

Con la finalidad de proteger la privacidad del usuario, se han realizado varios trabajos que implementan diferentes protocolos. En algunos trabajos se han propuesto protocolos que proporcionan un buen nivel de privacidad pero la calidad de servicio obtenida es muy ineficiente. Se han realizado otros trabajos basados en redes ***P2P*** o redes sociales, pero todavía no se ha conseguido un buen compromiso entre el nivel de protección de privacidad alcanzado y la calidad de servicio obtenida.

En este trabajo de investigación se ha propuesto una arquitectura ***P2P*** que permite la agrupación de los usuarios en función de sus intereses, permitiendo al mismo tiempo conservar un perfil sintético muy similar al perfil real. Los usuarios del mismo grupo ejecutan un protocolo de privacidad para enviar las consultas al ***WSE***. Con este protocolo se consigue que el ***WSE*** tenga un perfil que no se corresponde al del usuario en lo que se refiere a las consultas pero si a las categorías. El perfil que se obtiene sigue siendo útil para el usuario, y por lo tanto, le permite conseguir una buena calidad de servicio manteniendo un buen nivel de protección de la privacidad.

Para analizar el sistema, varias simulaciones se han llevado a cabo con diferentes configuraciones de los parámetros. Se ha evaluado el promedio de saltos y tiempo que hacen las consultas durante

el envío hacia el ***WSE***, el porcentaje del perfil real conocido por el ***WSE*** y la distancia entre el perfil real del usuario y su perfil ofuscado.

Los resultados muestran que cuando los usuarios están conectados en el sistema durante poco tiempo, el promedio de saltos se afecta principalmente por el parámetro de ofuscación, en segundo lugar por el número de intentos y finalmente por la probabilidad de aceptación. Las consultas hacen menos saltos a medida que el parámetro de ofuscación vaya incrementando, y se reducen más con menos intentos y poca probabilidad de aceptación. El porcentaje del perfil se ve afectado por el número de intentos y la probabilidad de aceptación; el ***WSE*** tiene menos información del perfil real cuando hay una gran posibilidad de aceptación con muchos intentos para encontrar usuarios que acepten las consultas. La distancia entre el perfil real y el perfil ofuscado del usuario crece a medida que se ofusca más con probabilidad alta de aceptación. El tiempo de respuesta está relacionado con el número de intentos y la probabilidad de aceptación: cuando se aceptan las consultas con menos intentos, la consulta se sirve en menos tiempo.

El comportamiento del sistema cambia relativamente cuando los usuarios permanecen conectados durante mucho tiempo. El promedio de saltos y la distancia se hacen más altos, y la proporción del perfil real conocida por el ***WSE*** más baja.

Para ambas situaciones, los mejores resultados se obtienen para ofuscación (≤ 2) con alta probabilidad de aceptación. En este caso, los usuarios aceptan las consultas para enviarlas al WSE si necesitan consultas para completar el perfil o la reenvian a otro usuario en caso contrario.

Para evaluar la eficiencia del sistema, se han determinado los mejores casos cuando se obtiene el mejor compromiso entre el nivel de privacidad y la calidad de servicio. Estos casos se iban analizando mientras aumentando el número de egoístas en el sistema. Las simulaciones han demostrado que incrementando el parámetro de ofuscación mientras el número de egoístas esté creciendo, el sistema sigue siendo eficiente pero con la condición de hacer menos intentos y de aceptar más de la mitad de las consultas que se intercambian entre los usuarios. De este modo, los usuarios honestos siguen teniendo un buen compromiso entre el nivel de privacidad y la calidad de servicio mientras que los usuarios egoístas acaban perdiendo la privacidad ya que toda la información que tiene el WSE sobre su perfil es exacta aunque no envian todas las consulas que generan por sí mismos.

Generalmente,

- Los resultados obtenidos demuestran que el sistema propuesto es una buena alternativa para la prtección de privacidad manteniendo al mismo tiempo una calidad de servicio muy aceptable.
- En la presencia de usuarios egoístas, el sistema sigue ofreciendo un compromiso aceptable con la condición de ofuscar el perfil.

- Se soluciona el problema estructural de la detección de consultas falsas en la mayoria de los esquemas single-party ya que el sistema utiliza consultas generadas por usuarios reales.
- Se supera la debilidad de los esquemas multi-party propuestos hasta ahora en ofrecer buena calidad de servicio.

6.2. Trabajo futuro

El sistema propuesto se considera robusto contra las consultas ilegales que los usuarios deshonestos intenten enviar al ***WSE*** mediante otros usuarios, por el hecho que los usuarios solamente envían al ***WSE*** consultas que tienen categorías que coinciden con sus intereses. Pero aun así, sería interesante como trabajo futuro implementar un mecanismo para controlar el origen de las consulas ilegales.

Los buenos resultados, que se han obtenido, motivan para hacer más trabajos en el futuro, basándose en el protocolo propuesto. Entre las ideas que surgen, la implementación de un ***plugin*** para el ***google chrome*** o el ***mozilla firefox***. De esta manera el usuario no tendrá que cambiar su navegador favorito y el sistema funcionará sin interferencias.

Nuestra contribución

Cristina Romero-Tris, Alexandre Viejo, Jordi Castellá-Roca e Youssef Benkaryouh. Sistema P2P de protección de la privacidad en motores de búsqueda basado en perfiles de usuario. *XIII Reunión Española sobre Criptología y Seguridad de la Información, Alicante 2014*. En prensa.

Bibliografía

[1] NetCraft. Febrero 2014. URL http://news.netcraft.com/archives/category/web-server-survey/.

[2] Blog Corperativo de Google. Febrero 2011. URL http://googleamericalatinablog.blogspot.com.es/2011/12/como-es-que-google-gana-dinero.html.

[3] Google Privacy Center. Febrero 2014. URL http://www.google.com/policies/privacy/.

[4] F. Crivellari M. Agosti and G. M. Di Nunzio. Web log analysis: a review of a decade of studies about information acquisition, inspection and interpretation of user interaction. *Data Min. Knowl. Discov.*, 24(3):663–696, December 2012. URL http://dx.doi.org/10.1007/s10618-011-0228-8.

[5] SeoReacherchers. Distribution of clicks on google's serps. Octubre 2006. URL http://www.seoresearcher.com/distribution-of-clicks-on-googles-serps-and-eye-tracking-analysis.htm.

[6] A. Cooper. A survey of query log privacy-enhancing techniques from a policy perspective. *ACM Trans. Web*, 2(4):19:1–19:27, Octubre 2008. URL http://doi.acm.org/10.1145/1409220.1409222.

[7] Discusión Metafilter. Agusto 2010. URL http://www.metafilter.com/95152/Userdriven-discontent#3256046.

[8] B. N. Levine and C. Shields. A multicast based protocol for anonymity. *Journal of Computer Security*, (10):213–240, 2002.

[9] DuckDuckGo. 2014. URL https://duckduckgo.com/.

[10] Ixquick. 2014. URL https://www.ixquick.com/.

[11] Page Wash. 2014. URL http://www.pagewash.com/.

[12] Start Page. 2014. URL https://startpage.com/.

[13] Yippy. 2014. URL http://yippy.com/.

[14] N. Mathewson R. Dingledine and P. Syverson. Tor: the second-generation onion router. *in Proceedings of the 13th conference on USENIX*, pages 21–31, 2004.

[15] TrackMeNot. Trackmenot: Resisting surveillance in web search. *Lessons from the Identity Trail: Anonymity, Privacy, and Identity in a Networked Society*, 23:417–436, 2009. URL http://mrl.nyu.edu/dhowe/trackmenot.

[16] A. Solanas J. Domingo-Ferrer and J. Castella-Roca. h(k)-private information retrieval from privacy uncooperative queryable databases. *Journal of Online Information Review*, 33(4): 1468–1527, 2009.

[17] R. Chow and P. Golle. Faking contextual data for fun, profit, and privacy. *in Proceedings of the 8th ACM Workshop on Privacy in the electronic society, WPES'09*, pages 105–108, 2009.

[18] S. T. Peddinti and N. Saxena. On the privacy of web search based on query obfuscation: a case study of trackmenot. *Proceedings of the 10th international conference on Privacy enhancing technologies, PETS'10*, pages 19–37, 2010.

[19] A. Viejo J. Castella-Roca and J. Herrera-Joancomarti. Preserving user's privacy in web search engines. *Computer Communications*, 32(13–14):1541–1551, 2009.

[20] A. Viejo C. Romero-Tris and J. Castella-Roca. Improving query delay in private web search. *P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), International Conference*, pages 200–206, 2011.

[21] Y. Lindell and E. Waisbard. Private web search with malicious adversaries. *Proceedings of the 10th international conference on Privacy enhancing technologies*, pages 220–235, 2010.

[22] J. Castella-Roca C. Romero-Tris and A. Viejo. Multi-party private web search with untrusted partners. *in 7th International ICST Conference on Security and Privacy in Communication Networks, SecureComm'11*, 2011.

[23] M. Reiter and A. Rubin. Crowds: anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

[24] B. N. Levine M. K. Wright, M. Adler and C. Shields. The predecessor attack: An analysis of a threat to anonymous communications systems. *ACM Transactions on Information and System Security*, 7(4):489–522, 2004.

[25] A. Viejo and J. Castella-Roca. Using social networks to distort users' profiles generated by web search engines. *Computer Networks*, 54(9):1343–1357, 2010.

[26] A. Viejo A. Erola, J. Castella-Roca and J. M. Mateo-Sanz. Exploiting social networks to provide privacy in personalized web search. *Journal of Systems and Software*, 84(9): 1734–1745, 2011.

[27] Open Directory Project. 2014. URL http://www.dmoz.org/.

[28] J. Castella-Roca D. Sanchez and A. Viejo. Knowledge-based scheme to create privacy-preserving but semantically-related queries for web search engines. *Inf. Sci.*, 218:17–30, 2013. URL http://dx.doi.org/10.1016/j.ins.2012.06.025.

[29] J. Domingo-Ferrer. Coprivacy: an introduction to the theory and applications of co-operative privacy. *SORT-Statistics and Operations Research Transactions*, 35:25–40, Sep 2011. URL http://crises2-deim.urv.cat/docs/publications/journals/638.pdf.