ANEXO III

(DOCUMENTACIÓN TÉCNICA PLATAFORMA "VICUS")

Capítulo 1

Introducción

1.1. Presentación

Desde finales del siglo XX, Internet ha tenido un impacto profundo en el trabajo, el ocio y el conocimiento a nivel mundial. Gracias a la web, millones de personas tienen acceso fácil e inmediato a una cantidad extensa y diversa de información en línea. La web ha permitido una descentralización repentina y extrema de la información y de los datos.

Son muchos usuarios los que utilizan Internet para descargar música, películas y otros trabajos, para ello hay fuentes que cobran por su uso y otras gratuitas en las que son los propios usuarios quienes ponen a disposición del resto los recursos. Éstas son las tecnologías P2P.

El P2P (Peer-to-Peer o redes de Igual a Igual) se basa principalmente en la filosofía de que todos los usuarios que desean descargar archivos, deben aportar recursos a la red. Conocida como filosofía P2P, es aplicada en algunas redes en forma de un sistema enteramente meritocrático en donde, el que más comparta, más privilegios tiene y más acceso dispone de manera más rápida a más contenido. Con este sistema se pretende asegurar la disponibilidad del contenido compartido, ya que de lo contrario no sería posible la subsistencia de la red.

El presente proyecto parte del programa de intercambio de archivos (P2P) aMule. El aMule se trata de la versión libre y multiplataforma del conocido eMule, publicado como software libre para sistemas Microsoft Windows, y que tiene como una de sus principales ventajas su gran base de usuarios (actualmente de cinco a diez millones) lo cual hace que sea un excelente medio para encontrar todo tipo de archivos.

Para el desarrollo de este trabajo han sido clave características de esta red, destacando especialmente el no-anonimato de sus usuarios.

El presente proyecto ha sido realizado para la colaboración de la Universidad de Vigo y la Unidad de Delitos Telemáticos de la Guardia Civil de Pontevedra, con el fin de poder ser incorporado a futuras investigaciones de casos de pedofilia en esta red.

A partir de ahora, y para facilitar la comprensión del texto, se hará referencia a nuestra aplicación aMule como -aMule Espía-

1.2. Motivación

La pornografía infantil constituye un problema de dimensión internacional, que se ha amplificado con la irrupción de nuevas tecnologías que han transformado las pautas de producción y difusión de este tipo de material.

El motivo por el que se ha realizado este proyecto ha sido la necesidad de desarrollar una herramienta útil para este tipo de investigaciones, las cuales carecían de sistemas tecnológicos que pudiesen discernir entre un usuario que por error se hubiera descargado un archivo de este tipo, y uno que realmente era consciente de su tenencia y contenido; todo ello centralizado en un sistema desde el que se puede unificar y consultar toda la información registrada de dichos usuarios de manera sencilla.

1.3. Proceso de desarrollo

El punto de partida de este trabajo ha sido el código fuente del aMule 2.2.3 [11], así como los proyectos para su compilación en Microsoft Visual Studio (Express Edition).

Una vez estudiado y compilado el código fuente original, se ha gestionado un servidor en el que fue creada una base de datos a la cual se conectará el aMule Espía.

El cliente aMule establecerá una comunicación bidireccional con la base de datos, de tal forma que ésta le indicará los ficheros por los que ha de preguntar en la red, mientras que el cliente aMule espía registrará en ella toda la información relevante. Para ello, se ha modificado el código fuente de tal forma que se han automatizado las búsquedas y conexíones a la base de datos, así como los registros de datos en la misma.

Posteriormente, para establecer una relación entre usuarios-servidor, se ha creado una interfaz web en la que, como se verá en el capítulo 3.4 , se facilita toda la comunicación entre la base de datos y el usuario de la aplicación, y a su vez añade seguridad y privacidad al sistema.

Una vez desarrollado lo expuesto anteriormente, se obtiene por un lado, un sistema centralizado y privado (servidor/ aMule espía) y por otro lado un sistema distribuido y público. ya que la aplicación aMule espía será para el resto de usuarios uno más en la red.

En resumen, el sistema se divide en tres capas: el servidor (que incluye la base de datos), la aplicación aMule espía y la interfaz web que comunica los dos anteriores.

1.4. Objetivos

El objetivo general para el desarrollo de este sistema es el detectar e identificar usuarios de la redes P2P con las que trabaja el aMule que tengan en su posesión archivos (fotos, videos, etc) de pornografía infantil, y que sean conscientes de su contenido, intentando descartar a aquellos usuarios que por un mero error se hayan descargado ficheros de este tipo.

Con el fin de alcanzar el objetivo general expuesto anteriormente, se fueron marcando objetivos a lo largo del desarrollo del proyecto. Éstos son los que se exponen a continuación:

- Conseguir que los usuarios del sistema creado no compartan los recursos descargados en la red aMule. Es decir, los usuarios del aMule espía se podrán descargar todos los recursos solicitados por la aplicación, pero éstos nunca podrán ser descargados por otros usuarios de la red.
- Gestionar un servidor de base de datos y web, conformando un sistema centralizado al cual se conectará la aplicación tanto para registrar información como para leerla de la misma.
- Modificar el código fuente del software aMule para que se conecte al servidor de base de datos (tanto para leer información como para registrar datos) de forma automatizada.
- Clasificar la información de la base de datos mediante una aplicación web desde la cual se ordena la información registrada de una forma clara y sencilla de tratar. De esta forma se consigue diferenciar entre usuarios con pocos archivos registrados, o aquellos que tienen un mayor volumen de ficheros de pornografía infantil, y por lo tanto es poco probable estar ante un falso positivo. Además desde dicha interfaz web se realizan tareas de gestión tales como añadir nuevos archivos a la base de datos o crear la misma.
- Realizar todo el sistema en un entorno seguro y privado, en el que no puedan acceder usuarios no autorizados a la base de datos, aplicación o información de la interfaz Web.
- Realizar un entorno de fácil manejo.

1.5. Descripción del documento

En este documento se describe el sistema desarrollado, tanto la aplicación aMule espía, como el servidor, la base de datos y la interfaz web implementada.

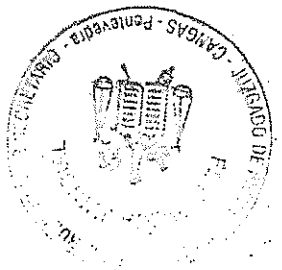
En el capítulo 2 se expiica la descripción del sistema. En él se describe el sistema cliente-servidor, las caracterísitcas del aMule original, una breve introducción a las redes P2P, así como todas las tecnologías y lenguajes utilizadas a lo largo del desarrollo del sistema.

A continuación, en el capítulo 3, se expone todo el trabajo realizado (servidor, base de datos, aMule e interfaz).

El documento continúa con un manual de usuario, con el fin de que un usuario con pocos conocimientos técnicos, pueda comprender de una manera clara cómo se debe poner en marcha, así como utilizar todo el sistema.

En el capítulo 5 se marcan las conclusiones a las que se ha llegado , así como las líneas futuras de desarrollo.

Finalmente se podrá consultar el apéndice y bibliografía consultada.



Capítulo 2

Descripción del sistema

Como ya hemos adelantado nuestro sistema se basa en un servidor el cual alberga una base de datos, y una aplicación P2P (Peer to Peer) modificada para que establezca una comunicación con el servidor. Además para facilitar el manejo de datos, el sistema se completa con una interfaz de comunicación web. En este capítulo se presentarán las características del sistema con el fin de que conformen una base antes de exponer el trabajo realizado. En primer lugar se explicará el tipo de arquitectura escogida para todo el sistema, Cliente/Servidor, actuando como cliente los usuarios del aMule Espía, y como servidor el punto donde se encuentra la base de datos. Posteriormente se presenta una introducción a las redes P2P, y para comprender las modificaciones y funcionamiento del aMule Espía se han de conocer características fundamentales del aMule original las cuales serán descritas en este cápítulo. Para finalizar se hará una breve introducción a las tecnologías y lenguajes necesarios para el desarrollo del trabajo.

2.1. Arquitectura cliente-servidor

Esta arquitectura consiste básicamente en un cliente que realiza peticiones a otra aplicación, el servidor, que le da respuesta.

Un servidor es una aplicación que ofrece un servicio a usuarios de Internet, siendo el cliente el que pide ese servicio. Los usuarios invocan la parte cliente de la aplicación, que construye una solicitud y se la envía al servidor usando TCP/IP como transporte (Internet).

El cliente es quien inicia las solicitudes o peticiones tomando el papel activo en la comunicación. Éste espera y recibe las respuestas del servidor (o servidores). El servidor al iniciarse espera la llegada de solicitudes de clientes desempeñando así un papel pasivo en la comunicación. Por lo general un servidor acepta conexiones de un gran número de clientes, aunque el número máximo de peticiones suele estar limitado.

La principal ventaja de esta arquitectura es que el servidor como centro de la red, es el único administrador de los recursos comunes a todos los usuarios,

como por ejemplo una base de datos, de esta forma los clientes no juegan un papel importante en la administración del sistema, creándose así una red centralizada y escalable. Gracias a esta arquitectura, es posible quitar o agregar clientes sin afectar al funcionamiento de la red y sin la necesidad de realizar mayores modificaciones. Por otro lado el servidor es el único eslabón débil en la red de cliente/servidor, debido a que toda la red está construida en torno a él. Afortunadamente, el servidor es altamente tolerante a los fallos.

En el presente proyecto cada cliente tiene a su alcance dos tipos de aplicaciones: una aplicación web y el propio aMule Espía. La aplicación web solicita al servidor información de la base de datos, sirviendo como interfaz entre usuario e información registrada. Por otro lado, el aMule Espía se conecta con el servidor para hacer peticiones de archivos y registrar información como veremos en el capítulo 3.

2.2. Redes P2P

Las redes P2P son redes de computadores descentralizadas y distribuidas en las cuales las aplicaciones pueden comunicarse entre sí, intercambiando información sin la intervención de un servidor central durante el momento del intercambio. La clave fundamental de este tipo de redes es que los nodos son tratados de igual a igual.

Las características deseables de toda red P2P son:

- Escalabilidad: Las redes P2P tienen un alcance mundial con cientos de millones de usuarios potenciales. En general, lo deseable es que cuantos más nodos estén conectados a una red P2P mejor será su funcionamiento. Así, cuando los nodos llegan y comparten sus propios recursos, los recursos totales del sistema aumentan. Esto es diferente en una arquitectura del modo cliente-servidor con un sistema fijo de servidores, en los cuales la adición de más clientes podría significar una transferencia de datos más lenta para todos los usuarios.

- Descentralización: Estas redes por definición son descentralizadas y todos los nodos son iguales. No existen nodos con funciones especiales, y por tanto ningún nodo es imprescindible para el funcionamiento de la red. En realidad, algunas redes comúnmente llamadas P2P no cumplen esta característica, como Napster, eDonkey o BitTorrent.

- Reparto de costes: Los costes o recursos están repartidos entre los usuarios.Se comparten o donan recursos a cambio de recursos. Según la aplicación de la red, los recursos pueden ser archivos, ancho de banda, ciclos de proceso o almacenamiento de disco.

- Anonimato: Es deseable que en estas redes quede anónimo el autor de un contenido, el editor, el lector, el servidor que lo alberga y la petición para

encontrarlo siempre que así lo necesiten los usuarios. Esta característica no se aplica a todas las redes P2P, de hecho es una de las vulnerabilidades que son útiles para la identificación de usuarios en el sistema desarrollado.

- Seguridad: Es una de las características deseables de las redes P2P menos implementada. Los objetivos de un P2P seguro serían identificar y evitar los nodos maliciosos, evitar el contenido infectado, evitar el espionaje de las comunicaciones entre nodos, creación de grupos seguros de nodos dentro de la red, protección de los recursos de la red... En su mayoría aún están bajo investigación, pero los mecanismos más prometedores son: cifrado multiclave, gestión de derechos de autor (la industria define qué puede hacer el usuario, por ejemplo la segunda vez que se oye la canción se apaga), reputación (sólo permitir acceso a los conocidos), comunicaciones seguras, comentarios sobre los ficheros...

La arquitectura de la red puede llevar a distintas clasificaciones, una de ellas puede establecerse mediante el grado de centralización (figura 2.1). De esta forma se tiene:

Redes centralizadas: Este tipo de red P2P se basa en una arquitectura monolítica en la que todas las transacciones se hacen a través de un único servidor que sirve de punto de enlace entre dos nodos y que, a la vez, almacena y distribuye los nodos donde se almacenan los contenidos. Poseen una administración muy dinámica y una disposición más permanente de contenido. Sin embargo, está muy limitada en la privacidad de los usuarios y en la falta de escalabilidad de un sólo servidor, además de ofrecer problemas en puntos únicos de fallo (caida del servidor), situaciones legales y enormes costos en el mantenimiento así como el consumo de ancho de banda. Por tanto, su principal característica es que todas las comunicaciones (como las peticiones y encaminamientos entre nodos) dependen exclusivamente de la existencia del servidor.

Redes P2P puras o totalmente descentralizadas: Las redes P2P de este tipo son las más comunes, siendo las más versátiles al no requerir de un gestionamiento central de ningún tipo, lo que permite una reducción de la necesidad de usar un servidor central, por lo que se opta por los mismos usuarios como nodos de esas conexiones y también para almacenar esa información. En otras palabras, todas las comunicaciones son directamente de usuario a usuario con ayuda de un nodo (que es otro usuario) quien permite enlazar esas comunicaciones. Todo nodo puede desempeñar el papel de servidor, cliente y enrutador.

Redes P2P híbridas, semi-centralizadas o mixtas: En este tipo de red, se puede observar la interacción entre un servidor central que sirve como unión y administra los recursos de banda ancha, enrutamientos y comunicación entre nodos pero sin saber la identidad de cada uno y sin almacenar información alguna, por lo que el

servidor no comparte archivos de ningún tipo a ningún nodo. Tiene la peculiaridad de funcionar de ambas maneras, es decir, puede incorporar más de un servidor que gestione los recursos compartidos, pero también en caso de que el o los servidores que gestionan se caigan, el grupo de nodos sigue en contacto a través de una conexión directa entre ellos mismos, con lo que es posible seguir compartiendo y descargando más información en ausencia de los servidores.

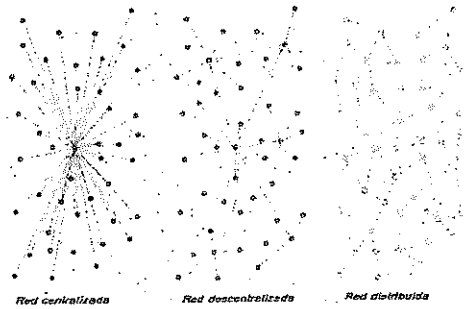


Figura 2.1: Topologías de redes P2P

2.3. El aMule

AMule [11] es un cliente multiplataforma para la red de compartición de archivos basado en el cliente de Windows eMule.

La filosofía fundamental de este tipo de aplicaciones es el intercambio de ficheros. De esta forma los usuarios conectados compartirán sus archivos con el fin de que ellos puedan beneficiarse de los publicados por el resto de usuarios (clientes de la red).

2.3.1. Arquitectura de las redes del aMule

Cuando un usuario desea gestionar una búsqueda o una descarga, ha de conectarse a la red mediante los protocolos Edonkey (también llamado Ed2k), Kademlia, o ambos. De esta forma, un usuario que esté conectado a la red Edonkey buscará los archivos en ésta, mientras que si la conexión se establece por la red Kad (protocolo Kademlia) los usuarios que compartirán los archivos con nosotros estarán también conectados a esta red.

Red Edonkey

La característica principal es que todo cliente ha de conectarse a un servidor para poder acceder a los servicios ofrecidos (figura 2.2).

Al iniciar la aplicación se obtiene una lista de servidores preconfigurados, y una lista de archivos compartidos albergados en su ordenador. El usuario establece

una conexión TCP con el servidor, obteniendo de él la información de los archivos que desea, así como de los usuarios que los tienen. Al mismo tiempo, se establecen cientos de conexiones TCP con otros clientes para la bajada y subida de archivos.

Durante el proceso de conexión, el servidor le da al cliente un ID el cual le servirá como identificador para todas las sesiones futuras (ver 2.3.2). Un cliente nunca puede estar conectado a más de un servidor.

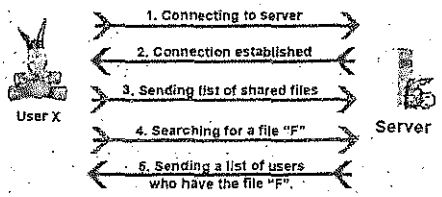


Figura 2.2: Funcionamiento del protocolo Edonkey

Red Kad

La red Kad [26] [24] comprende más del 80 % de la base de usuarios y probablemente cerca del 95 % de las instalaciones ed2k.

Esta red se basa en el protocolo de comunicación Kademlia. Éste no requiere de servidores para su funcionamiento sino que se sustenta en la comunicación con otros nodos (de los cuales conoce la IP y puerto) y la información que de éstos puede extraer, la cual generalmente consiste en paquetes con información sobre otros nodos o sobre archivos relacionados con una palabra clave de búsqueda. Es por tanto una red descentralizada (figura 2.3).

El objetivo de Kad es conseguir una conexión directa, en la cual cada usuario es a la vez cliente y servidor. Para esto, se transforma en un nodo de la red y se comunica con sus nodos vecinos para tener acceso a todas las informaciones de la red. Para su funcionamiento, Kad necesita :

- Aprender la topología de la red sobre la cual se conecta el cliente.
- Buscar la información sobre todos los nodos y sus ficheros.
- Recibir una respuesta de un nodo que responde a los criterios buscados.

La red Kad funciona como un enlace de comunicación entre usuarios, en cambio, cuando se inicia la transferencia de un archivo, los clientes se conectan directamente uno con el otro usando la red IP estándar.

Cuando se busca, cada cliente actúa como un pequeño servidor y se le da la responsabilidad de ciertas palabras clave o fuentes. Esto añade complejidad al encontrar datos, ya que no existe un servidor central al que preguntar, pero a cambio se propagará la consulta a través de la red y así se va conociendo la topología de la misma.

La red Kad es fundamental para este proyecto, ya que, como se ha mencionado antes, son más de un 80 % de usuarios los que la utilizan, y además teniendo en cuenta que sólo se van a realizar búsquedas sobre archivos de pornografía infantil, será muy probable que los nodos que se vayan conociendo sean del mismo tipo.

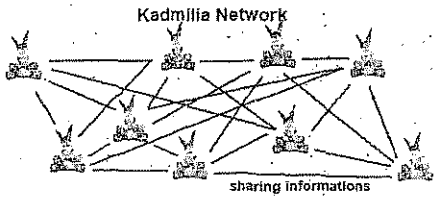


Figura 2.3: Esquema de la red Kad

Si establecemos una comparativa entre la red Kad y la Ed2k, ambas redes tienen un concepto totalmente distinto de conseguir lo mismo: buscar ficheros y buscar fuentes para los ficheros. El principal objetivo de la red Kademlia es ser totalmente independiente de los servidores y facilitar la escalabilidad. Los servidores sólo pueden aceptar un cierto número de clientes y si un servidor grande deja de funcionar la red se resiente gravemente. Kademlia es una red autoorganizativa y se ajusta a sí misma para obtener el mejor rendimiento posible en base al número de usuarios y la calidad de sus conexiones. Por tanto es más resistente a daños a gran escala en la red, pero por otro lado el hecho de que en la red Edonkey se encuentre toda la información centralizada en servidores permite búsquedas más rápidas tanto de archivos como de usuarios que lo poseen.

2.3.2. Identificación de usuarios

Las redes P2P [26] se basan en conexiones directas entre dos usuarios para transferir datos. Para que esto se cumpla se debe conocer la dirección IP del usuario de destino. Por medio de esta dirección se puede identificar y localizar al usuario. Sólo hay dos formas posibles que garanticen el anonimato: desviar el tráfico a través de un *proxy* anonimizador o canalizar dicho tráfico a través de otros clientes. Los servidores *proxy* tienen una considerable serie de desventajas: pueden estar caídos, el grado de anonimato depende de la buena voluntad del administrador del *proxy* y el tráfico que consumen las conexiones cuesta mucho dinero. Pongamos un ejemplo: un servidor para 100.000 usuarios aMule. Cada uno de los clientes conectados genera un tráfico de 10 kB/s en ambas direcciones, por lo que el servidor debe de soportar un volumen de tráfico de 2000 MB/s (1,98 GB/s). Pocos pueden mantener económicamente un servidor de esas características.

Desviar o canalizar el tráfico a través de terceros clientes causa un impacto similar, debido a que la mitad del tráfico sería usado para el anonimato, lo cual

significa que los usuarios deberían de tener el doble de ancho de banda de subidas para mantener el rendimiento de la red en los niveles actuales.

Aparte de la dirección IP que identifica la máquina, el aMule identifica a sus clientes mediante dos tipos de identificadores: el ID de usuario y el Hash de usuario (*UserHash* o *HashID*).

El ID de usuario es un identificador utilizado sólo en la red Edonkey (y no en Kad). Cuando conectas tu cliente a un servidor, el servidor comprueba si otros clientes van a poder conectarse directamente contigo. Si es así, el servidor asigna lo que se llama una "ID alta". Si la comunicación se encuentra bloqueada, entonces el servidor asigna una "Id baja". Como vemos este ID nos indica si es posible establecer una comunicación correcta entre dos usuarios. [1]

Tener ID alta quiere decir que el puerto predeterminado se halla abierto y puede accederse a él libremente. Por el contrario, tener ID baja supone tener el puerto bloqueado o inaccesible. La causa de esto puede hallarse en la presencia de cortafuegos (*firewalls*), enrutadores (*routers*) o servidores *proxy* (*proxy servers*).[2] Tener una ID baja no quiere decir que no se puedan descargar o subir archivos, si bien es cierto que tiene algunas desventajas ya que la mayoría de los mensajes del protocolo de comunicación (peticiones de conexión, de cola para descargar un fichero, de búsqueda de fuentes, etc...) tienen que hacerse desde el servidor.

En Kad no se habla de ID baja o alta, sino que se utiliza el Estado. Si nuestro puerto UDP es accesible desde el exterior tendremos estado abierto(*connected*), equivalente a una ID alta en la red Ed2k. Si el puerto está cerrado nos hallaremos tras cortafuegos en Kad (*firewalled*), situación ésta análoga a una ID baja en la red tradicional con servidores. La red Kademlia soporta un Mediador (*Buddy*) para los usuarios tras un cortafuegos. Los Mediadores son otros clientes de Kademlia que tienen estado abierto y actúan como punto de contacto para aquellos usuarios que están tras cortafuegos y por tanto no son accesibles directamente.

Al conectar nuestra red Kad nos encontraremos en estado "desconectado", una vez que se establezca una comunicación con un número bajo de nodos (no superior a 200 y albergados en el fichero nodes.dat) pasaremos a estado *"firewalled"*, y cuando conozcamos aproximadamente 300 nodos de la red pasaremos a estar "conectado".

Como se ha mencionado antes, el ID de usuario es calculado mediante operaciones sobre la IP de la máquina, de tal forma que si ésta cambia (una IP dinámica), este ID también lo hace (aunque seguirá siendo Alto o Bajo como antes). No obstante, existe un identificador cuyo valor es independiente de la dirección IP, se trata del *Hash* de usuario.

El *userhash* es un conjunto de 16 bytes que el programa asigna la primera vez

[1] No tiene ninguna importancia el valor numérico, para los efectos de las transferencias solo interesa saber si es *HIGH* o *LOW* . La relación entre un IP y una ID alta es como sigue: a una IP = A.B.C.D le corresponde la ID alta = A + B*256 + C*256*256 + D*256*256*256.

[2] por un error en la programación del protocolo Ed2k, no se ha tenido en cuenta las Ips finalizadas en 0, de tal forma que es totalmente imposible conseguir un *High ID* para Ips cuyo último octeto finalicen en cero.

que es lanzado. Cuando entramos en contacto con otros clientes lo envíamos para que puedan identificarnos del mismo modo que ellos nos mandan su userhash. La tarea para la cual fue creado el *HashID* es la identificación de usuarios en la gestión de créditos. Los créditos recompensan a los usuarios que suben datos a la red, estableciéndose una relación proporcional entre créditos y datos transferidos. Éstos son el modificador más alto cuando se calcula la posición y el progreso en la cola de descargas de otro cliente de tal forma que cuantos más créditos se tengan más rápido se avanzará.[3]

2.3.3. Identificación de archivos

Todos los ficheros tienen un identificador *hash*. Este *hash* es una combinación de números y letras que son capaces de identificar de forma única a un fichero. Un fichero puede tener múltiples nombres, pero esto no cambia su valor de *hash*. Esto permite que cada usuario encuentre todas las fuentes para un fichero en particular sin importar que alguien le haya cambiado el nombre al mismo.

El aMule utiliza el *hash MD4* tanto para identificar archivos, como para localizar errores en la transmisión de éstos. Para ello divide el fichero en partes a las cuales se les calcula su *hash MD4*, una vez se tengan todas las partes se concatenan sus *hashes* obteniendo un gran bloque, al cual se le vuelve a calcular la operación dando lugar al *hash* del archivo.

aMule cuenta con un sistema de gestión inteligente de la corrupción [26] de forma que cada archivo se divide en varias partes, que son conocidas como *chunks* y se obtiene el *hash* de cada uno. Cada *chunk* tiene 9.28MB, así que, por ejemplo, un archivo de 15MB se divide en dos *chunks* (9.28MB + 5.72MB), un archivo de 315KB será una única parte y un archivo de 100MB se dividirá en 11 *chunks* (10x9.28MB + 7.2MB). Cada vez que aMule finaliza una de esas partes comprueba si los datos que se han descargado coinciden con el *hash* de esa parte finalizada. Si es así, este *chunk* se ofrece a otros clientes para ayudar en su distribución. Si no coincide, ha habido un problema de corrupción y toda la parte necesita ser descargada de nuevo.

2.3.4. Búsqueda de ficheros y fuentes

Una vez conectado a la red, el cliente puede buscar ficheros basándose en palabras clave o bien en enlaces Ed2K.

Si hablamos de palabras clave, y en el caso de conexión a la red Edonkey, una búsqueda puede ser local o bien global. Si es local (busca sólo en el servidor en el que estás conectado) las búsquedas son más rápidas, pero pueden ofrecer menos resultados. Si la búsqueda es global (busca en todos los servidores de la red), le costará más pero se obtendrán un mayor número de resultados. Cada servidor busca

[3] aunque esta sea su cometido, para nosotros lo fundamental es que se trate de un identificador permanente y que no varíe con la IP.

las palabras en su base de datos local y devuelve una lista con los nombres de los archivos (y sus valores *hash*) que se ajustan a esa búsqueda.

En la red Kad, se asocia a cada cliente una cierta "responsabilidad" en base a su hash. Cada cliente de la red Kademlia trabaja como un servidor para ciertas palabras clave o fuentes. El objetivo de cualquier búsqueda es encontrar a aquellos clientes que tienen la responsabilidad para el término de búsqueda actual. Esto se consigue mediante un cálculo complejo acerca de la posible distancia al cliente destino y preguntando a otros clientes cual será la ruta más corta hasta él. Obviamente, todo esto se encuentra automatizado en las opciones de búsqueda dadas por el software.

Por otro lado, los enlaces Ed2K son un formato especial [4] de enlaces que permiten añadir una descarga directamente a aMule. Las partes fundamentales son: el *hash* del archivo y el tamaño del mismo. Además puede incluir otros datos como el nombre, Ips y puertos de clientes que lo poseen.

Una vez que se encuentran los ficheros en nuestra cola de descargas, se han de buscar las fuentes[5] que los posean. En el caso de que nos encontremos conectados a un servidor, aMule primero contacta con él y posteriormente al resto de los servidores de la red, buscando fuentes para esa descarga en particular. El servidor busca ese *hash* en su base de datos y devuelve los clientes que él sabe que poseen dicho fichero. En el caso de la red Kad, las fuentes se buscan del mismo modo que las palabras clave.

Sólo pueden ser fuentes válidas aquellos clientes que hayan descargado al menos una parte entera (9,28 MB) del fichero (la última parte puede tener un tamaño menor). Así, todo usuario que se haya descargado un *chunk* (parte de un fichero) pasa directamente a compartirlo con el resto de usuarios.

Esta búsqueda es un proceso dinámico, es decir cada vez que se encuentra una fuente para nuestro archivo se establece un intercambio, de manera que si conociera alguna fuente útil para nosotros, nos enviaría una lista de ellas y pasarían a ser fuentes de este fichero.

En la ventana tráfico de la aplicación, se puede ver en la columna fuentes el formato de hasta cuatro valores a/b+c (d) [27]:

- El primer número "a" son las fuentes encontradas que tienen partes que se necesitan y a las que no se les ha pedido ningún otro archivo. Si una fuente está en A4AF [6] no se contabiliza en este número. Se podría decir que son las fuentes útiles, de las que se está en la cola de espera, esté llena o no, incluyendo de las que se esté descargando.

- El segundo "b" son todas las fuentes encontradas, estén o no útiles en ese

[4] ejemplo de enlace Ed2k:
ed2k://|file|xxfilename.avixx|14997504|955e013e991ee246d63d45ea71954c4d|/ donde la palabra "file" precede al nombre del archivo,si existe, al cual le sigue el tamaño y el hash

[5] las fuentes son usuarios que aMule va encontrando y que tienen al menos 9.28 Mb de un archivo que se ha puesto en la lista de descargas.

[6] Ask For Another File: fuentes a las cuales se les pide más de un archivo

instante. Tampoco se contabilizan aquí las fuentes A4AF. Siempre "b" es mayor que "a". La diferencia entre ambos números es que en "b" se contabilizan también las fuentes "inútiles" en ese instante:

- Fuentes a las que se está conectando aún o preguntando.
- Fuentes con low ID en caso de que tambien se tenga ya que imposibilita la conexión.
- Fuentes con demasiadas conexiones.
- Fuentes en estado Desconocido.
- Fuentes problemáticas o bloquedas.
- Fuentes de las que no se necesitan partes.

Algunas de estas, si es posible, irán pasando a útiles según avanza el tiempo. Las que son totalmente inútiles irán saliendo de la lista en más o menos tiempo, dependiendo de su estado. Por este motivo en un periodo de aproximadamente 5 minutos tendremos un estado estable de las fuentes.

- El tercero de los números arriba citados, "c", cuenta el número de fuentes en estado A4AF. Una fuente concreta sólo puede ser usada para una única descarga. Si esa fuente tiene partes disponibles para otra, aMule decide qué descarga tendrá preferencia. En ese momento las otras mostrarán esa fuente como "(A4AF) Se le ha pedido otro archivo"

- El cuarto "d", el que va entre paréntesis, es el número de fuentes de las que se está descargando en ese momento.

2.4. Tecnologías y lenguajes empleados

Para el desarrollo del sistema se han empleado numerosas herramientas existentes. A continuación se hace una breve introducción las características principales de las más destacadas.

2.4.1. XAMPP

XAMPP [12] es un servidor independiente de plataforma, que consiste principalmente en el gestor de base de datos MySQL, el servidor Web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo de X (para cualquiera de los diferentes sistemas operativos), Apache, MySQL, PHP, Perl. El programa actúa como un servidor Web libre, fácil de usar y capaz de interpretar páginas dinámicas. Actualmente XAMPP esta disponible para Microsoft Windows, GNU/Linux, Solaris, y MacOS X, bajo una compilación de software libre, es gratuito y libre para ser copiado conforme los términos de la licencia GNU (Licencia Publica General).

XAMPP también incluye otros módulos como OpenSSL y phpMyAdmin, para incorporar seguridad y gestionar las base de datos respectivamente.

La filosofía detrás de XAMPP es la construcción de una versión fácil de instalar para los desarrolladores que entran al mundo de Apache. Para hacerlo más conveniente, XAMPP está configurado con todas las funciones activadas, pero esta configuración no es buena desde el punto de vista de seguridad y no es suficientemente buena para un ambiente de producción, por este motivo, como se verá en la sección 3.1, se han modificado numerosos parámetros de su configuración.

Los principales módulos empleados de XAMPP se muestran a continuación:

Apache

El servidor HTTP Apache es un servidor web HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Windows, Macintosh y otras. El hecho de que sea un sistema multiplataforma lo hace practicamente universal.

Apache es usado primariamente para enviar páginas web estáticas y dinámicas en la World Wide Web. Muchas aplicaciones web están diseñadas asumiendo como ambiente de implantación a Apache, o que utilizarán características propias de este servidor web, pero éste es usado para muchas otras tareas donde el contenido necesita ser puesto a disposición en una forma segura y confiable.

La arquitectura del servidor Apache es muy modular. El servidor consta de una sección core y diversos módulos que aportan muchas de las funcionalidades que podría considerarse básica para un servidor web. Los módulos más interesantes para el desarrollo del proyecto son: el módulo para comunicaciones seguras mediante seguridad en la capa de transporte (SSL y TLS) y un módulo externo para gestión de páginas dinámicas en PHP.

La licencia de software bajo la cual el software de la fundación Apache es distribuido es una parte distintiva de la historia de Apache HTTP Server y de la comunidad de código abierto. La Licencia Apache permite la distribución de derivados de código abierto y cerrado a partir de su código fuente original.

PHP

PHP es un lenguaje interpretado de propósito general ampliamente usado y que está diseñado especialmente para desarrollo web y puede ser incrustado dentro de código HTML. Éste es usado principalmente en interpretación del lado del servidor (server-side scripting), tomando el código en PHP como su entrada y creando páginas web como salida. Puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. Es publicado bajo la PHP License, considerando esta licencia como software libre.

El gran parecido que posee PHP con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores crear aplicaciones complejas con una curva de aprendizaje muy corta. También les

permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones. Cuando el cliente hace una petición al servidor para que le envíe una página web, el servidor ejecuta el intérprete de PHP. Éste procesa el script solicitado que generará el contenido de manera dinámica (por ejemplo obteniendo información de una base de datos). El resultado es enviado por el intérprete al servidor, quien a su vez se lo envía al cliente. Mediante extensiones es también posible la generación de archivos PDF, Flash, así como imágenes en diferentes formatos; además permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, Postgres, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite.

MySQL

MySQL [5] es un sistema de gestión de base de datos (SGBD [7]) veloz, multihilo, multiusuario y robusto. Éste está proyectado tanto para sistemas críticos en producción soportando intensas cargas de trabajo, como para empotrarse en sistemas de desarrollo masivo de software. El software MySQL tiene licencia dual, pudiéndose usar de forma gratuita bajo licencia GNU o bien adquiriendo licencias comerciales de MySQL AB en el caso de no desear estar sujeto a los términos de la licencia GPL. MySQL es una marca registrada de MySQL AB.

Existen varias APIs que permiten a aplicaciones escritas en diversos lenguajes de programación, acceder a las bases de datos MySQL, entre estos lenguajes podemos encontrar C, C++, Pascal, Delph, Perl, PHP, Java o TCL entre otros. Cada uno de estos utiliza una API específica.

Cualquier consulta a la base de datos se hace por medio del lenguaje SQL (Structured Query Language). Éste es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre ellas. Una de sus características es el manejo del álgebra y el cálculo relacional permitiendo efectuar consultas con el fin de recuperar información de interés de una base de datos, así como también hacer cambios sobre ella. Es un lenguaje declarativo de "alto nivel" o "de no procedimiento", que gracias a su fuerte base teórica y su orientación al manejo de conjuntos de registros, y no a registros individuales, permite una alta productividad en codificación y la orientación a objetos. De esta forma una sola sentencia puede equivaler a uno o más programas que utilizados en un lenguaje de bajo nivel orientado a registro.

PHPMyAdmin

PHPMyAdmin es una herramienta escrita en PHP con el fin de administrar bases de datos de MySQL a través de páginas web (figura 2.4). Actualmente puede crear y eliminar bases de datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos,

[7] son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.

administrar privilegios, exportar datos en varios formatos y está disponible en 50 idiomas.

Al igual que PHP, se encuentra disponible bajo la licencia GPL.[15]

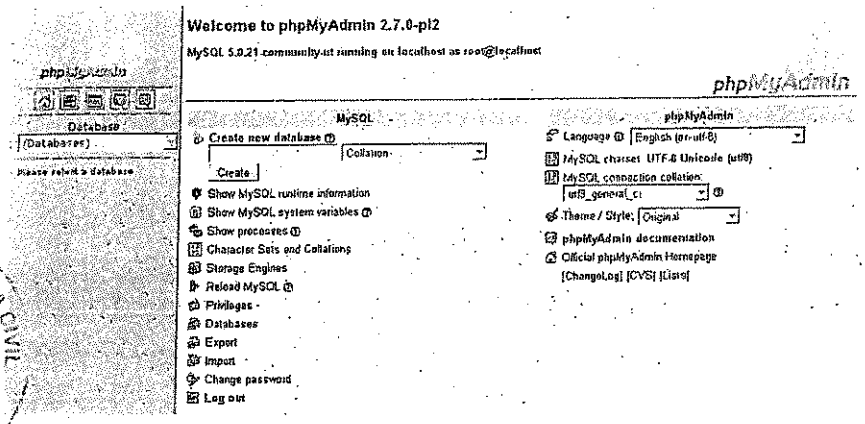


Figura 2.4: Menú de inicio del administrador de bases de datos PHPMyadmin

2.4.2. HTML

HTML, siglas de HyperText Markup Language (Lenguaje de Marcas de Hipertexto), es el lenguaje de marcado predominante para la construcción de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementarlo con objetos tales como imágenes.

HTML utiliza etiquetas o marcas, que consisten en breves instrucciones de comienzo y final, mediante las cuales se determina la forma en la que debe aparecer en su navegador el texto, así como también las imágenes y los demás elementos, en la pantalla del ordenador. Toda etiqueta se identifica porque está encerrada entre los signos menor que y mayor que, y algunas tienen atributos que pueden tomar algún valor. La mayoría de los atributos de un elemento son pares nombre-valor, separados por un signo de igual "=" y escritos en la etiqueta de comienzo de un elemento, después del nombre de éste.

Dentro de HTML puede ser incorporado código PHP además de poder incluir un script (por ejemplo Javascript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

2.4.3. Microsoft Visual Studio

Microsoft Visual Studio (MVS) es un entorno de desarrollo integrado (IDE) para sistemas operativos Windows. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic .NET, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros.

Visual Studio permite a los desarrolladores crear aplicaciones, sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión net 2002). Así se pueden crear aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas web y dispositivos móviles.

A diferencia de las tecnologías explicadas anteriormente, MVS no es código libre ni gratuito; pero existen versiones limitadas (Ediciones Express Edition) [13] las cuales hemos utilizado para la compilación y desarrollo de nuestra aplicación Amule Espía [8].

2.4.4. C++

El aMule, al igual que el eMule, ha sido programado en C++. Éste es un lenguaje de programación que extiende el conocido C con mecanísmos que permiten la manipulación de objetos haciéndolo un lenguaje multiparadigma, ya que permite la programación genérica, estructurada y orientada a objetos.

La programación genérica es un tipo de programación que está mucho más centrada en los algoritmos que en los datos. La idea de esta forma de programar pretende generalizar las funciones utilizadas para que puedan usarse en más de una ocasión. Esto se consigue parametrizando lo máximo posible el desarrollo del programa y expresados o devueltos de la forma más simple posible, evitando detalles concretos. La biblioteca de funciones conseguida con esta manera de programa permite que esas funciones puedan servir para más programas de los que, otras más concretas, podrían ser útiles; y también aplicando pocos cambios, conseguir que realice diferentes acciones.

La programación estructurada se basa en una programación clara. Para ello utiliza únicamente tres estructuras: secuencia, selección e iteración; siendo innecesario y no permitiéndose el uso de la instrucción o instrucciones de transferencia incondicional (GOTO, EXIT FUNCTION, EXIT SUB o múltiples RETURN. Los programas son más fáciles de entender, ya que pueden ser leídos de forma secuencial. El principal inconveniente de este método de programación es que se obtiene un único bloque de programa, que cuando se hace demasiado grande puede resultar problemático su manejo; esto se resuelve empleando la programación modular, definiendo módulos interdependientes programados y compilados por separado.

La Programación Orientada a Objetos (POO) es un paradigma de programación que usa objetos y sus interacciones para diseñar aplicaciones. Está basado en varias técnicas, incluyendo herencia, modularidad, polimorfismo y encapsulamiento. Los objetos son entidades que combinan estado, comportamiento e identidad: El estado está compuesto de datos, será uno o varios atributos a los que se habrán asignado unos valores concretos (datos), el comportamiento está definido por los procedimientos o métodos con que puede operar dicho objeto, es decir, qué operaciones se pueden realizar con él, mientras que la identidad es una propiedad de un objeto que lo diferencia del resto, dicho con otras palabras, es su identificador

[8] concretamente Microsoft Visual C++

(concepto análogo al de identificador de una variable o una constante). La programación orientada a objetos expresa un programa como un conjunto de estos objetos, que colaboran entre ellos para realizar tareas. Esto permite hacer los programas y módulos más fáciles de escribir, mantener y reutilizar. De esta forma, un objeto contiene toda la información que permite definirlo e identificarlo frente a otros objetos pertenecientes a otras clases e incluso frente a objetos de una misma clase, al poder tener valores bien diferenciados en sus atributos. A su vez, los objetos disponen de mecanismos de interacción llamados métodos que favorecen la comunicación entre ellos. Esta comunicación favorece a su vez el cambio de estado en los propios objetos. Esta característica lleva a tratarlos como unidades indivisibles, en las que no se separan ni deben separarse el estado y el comportamiento.

Una particularidad del C++ es la posibilidad de redefinir los operadores (sobrecarga de operadores), y de poder crear nuevos tipos que se comporten como tipos fundamentales. Además como hemos visto permite trabajar tanto a alto como a bajo nivel.

Capítulo 5

Conclusiones y líneas futuras

5.1. Conclusiones

En esta memoria se ha expuesto el sistema desarrollado como Proyecto Fin de Carrera para la localización de recursos en redes P2P. Este sistema ha sido realizado en colaboración con la Guardía Civil con el fin de identificar usuarios cuyos recursos hayan sido previamente etiquetados como pornografía infantil. El objetivo principal era la identificación veraz de estos usuarios, reduciendo así el número de falsos positivos a la hora de realizar una investigación.

El sistema formado por servidor, base de datos, interfaz web y la propia aplicación modificada aMule (aMule espía) interaccionan entre si de forma autónoma. La existencia de una base de datos donde se registran los datos, junto con la aplicación web, permite cotejar la información registrada para conocer cuántos archivos se le ha detectado a cada usuario, reduciendo así el mayor número de falsos positivos.

El sistema ha sido diseñado para que su manejo sea lo más sencillo posible. Para ello, se facilita dentro de la presente memoria un manual de usuario dotado de los pasos a seguir, glosario, posibles problemas y distintas explicaciones que facilitan su utilización.

Durante el desarrollo del sistema, se han realizado numerosas pruebas en las que se ha demostrado el correcto funcionamiento del proyecto así como su indiscutible utilidad. Dichas pruebas han sido realizadas con una duración menor de una semana, y en ellas se han podido detectar más de dos mil usuarios sospechosos, de los cuales a más de cincuenta se le han detectado más de cuatro archivos diferentes de un total de ochenta y cuatro de la base de datos.

Al poderse ubicar a los usuarios geográficamente, destaca el hecho de la masificación de los casos detectados en Alemania, Estados Unidos o el Este Asiático, habiéndose detectado también casos en España y Portugal entre otras muchas situaciones.

Actualmente el sistema ha sido entregado a la Guardia Civil y se encuentra en un periodo de pruebas por sus usuarios finales con el fin de solventar cualquier

duda o posible problema que se detecte.

5.2. Líneas futuras

Son muchas las ampliaciones que se pueden realizar sobre este proyecto con el fin de hacerlo más completo, robusto y seguro.

Una de las funcionalidades más deseadas sería poder crear un sistema capaz de detectar nuevos archivos además de identificar a los usuarios.

Además se podrían hacer registros de otra información relevante, como por ejemplo el primer usuario al que se le fue detectado un archivo (con el fin de establecer un acercamiento a usuarios generadores de pornografía infantil) o bien si un usuario ha borrado los archivos una vez descargados (para así identificar aquel que es consumidor de pornografía infantil aunque no comparta dicha información en la red porque lo guarda en cualquier otro dispositivo externo o simplemente los mueve a una carpeta no compartida).

Se plantea además la posibilidad de crear un archivo espía, de tal forma que si un usuario solicita un fichero de pornografía infantil, se le suplante por dicho espía instalándose como *spyware* en su ordenador, con el fin de obtener nuevos archivos y un análisis exausitivo de dicho usuario (siempre de forma legal).

Para obtener un sistema más robusto y seguro sería aconsejable el escalado horizontal del servidor. Es decir, disponer de más de un servidor de base de datos y aplicación web tanto para el reparto de la carga como por seguridad ante un posible fallo en el servidor central.

De esta forma, si se interconectan dos servidores, formando un *cluster* de servidores, si el principal (activo) se cayese pasaría a funcionar el otro (pasivo). Además habría que realizar una copia de los datos almacenados en la base de datos, para preveer posibles desastres sobre los discos del servidor principal.

Este servidor (o servidores) secundarios se podrían ubicar en el mismo centro que el principal, o formar un centro de respaldo específicamente diseñado para tomar el control del otro principal en caso de contingencia. Es decir, proveer al sistema de más de un servidor, que se comuniquen entre sí y que a su vez repartan la carga de los posibles usuarios que quieran acceder a los mismos.

Apéndice E

Glosario

Api: Una interfaz de programación de aplicaciones o API (del inglés Application Programming Interface) es el conjunto de funciones y procedimientos (o métodos, si se refiere a programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software.

A4AF: Asked For Another File (Preguntado por otro archivo o se le ha pedido otro archivo).

Cookies: Es un fragmento de información que se almacena en el disco duro del visitante de una página web a través de su navegador, a petición del servidor de la página. Esta información puede ser luego recuperada por el servidor en posteriores visitas.

Dirección IP: Es un número que identifica de manera lógica y jerárquica a una interfaz de un dispositivo (habitualmente una computadora) dentro de una red. Es habitual que un usuario que se conecta desde su hogar a Internet utilice una dirección IP. Esta dirección puede cambiar cada vez que se conecta; y a esta forma de asignación de dirección IP se denomina una dirección IP dinámica (normalmente se abrevia como IP dinámica).

Ed2k: Edonkey.

Hash: Clave hexadecimal que identifica de forma única un archivo, aunque éste tenga diversos nombres, de manera que un mismo archivo que tengan diferentes usuarios, aunque alguno de ellos haya modificado el nombre, continúa siendo el mismo archivo.

HTTPS: Protocolo de red basado en el protocolo HTTP, destinado a la transferencia segura de datos de hipertexto, es decir, es la versión segura de HTTP.

Interfaz web: Interfaces gráficas de usuario con unos elementos comunes de presentación y navegación. Este tipo de interfaces deben servir de intermediarias entre unos usuarios genéricos, no acostumbrados generalmente al uso de aplicaciones informáticas, y unos sistemas de información y procesos transaccionales que corren por debajo, debiendo posibilitar la localización de la información deseada, el entendimiento claro de las funcionalidades ofrecidas, la realización práctica de tareas específicas por parte de los usuarios y la navegación intuitiva por las diferentes páginas que forman el sitio web.

Link: Elemento de la interface de un programa que permite cambiar rápidamente lo que se está viendo sin cambiar de ventana que se usa en un programa o menú.

Localhost: Es un nombre reservado que tienen todas las computadoras, router o dispositivo que disponga de una tarjeta de red ethernet para referirse a sí mismo. El nombre localhost es traducido como la dirección IP 127.0.0.1

Login: Autentificación verificando que el usuario que intenta realizar funciones en un sistema es de hecho el usuario que tiene la autorización para realizarlas.

Maxfiles: Es el parámetro que indica el número máximo de archivos que se le pueden detectar a un usuario sin que sea catalogado como sospechoso.

Pestaña: Elemento de la interface de un programa que permite cambiar rápidamente lo que se está viendo sin cambiar de ventana que se usa en un programa o menú. Desempeñar una tarea a través de pestañas permite cargar varios elementos separados dentro de una misma ventana y así es posible alternar entre ellos con una mayor comodidad.

P2P: Peer-to-Peer, red de computadoras en la que todos o algunos aspectos de esta funcionan sin clientes ni servidores fijos, sino una serie de nodos que se comportan como iguales entre si. Es decir, actúan simultáneamente como clientes y servidores respecto a los demás nodos de la red.

Root: Nombre convencional de la cuenta de usuario que posee todos los derechos en todos los modos, es también llamado superusuario o administrador.

Servidor: Es una computadora que, formando parte de una red, provee servicios a otras computadoras denominadas clientes.

Sesión: Momento en el que se establece un registro de datos en la base de datos, tanto si se trata de un registro de un nuevo usuario, como de la actualización de datos de una sesión anterior.

GUARDIA CIVIL - PONTEVEDRA
UNIDAD ORGÁNICA DE POLICÍA JUDICIAL
Equipo de Investigación Delitos Tecnológicos

197

SSL: Secure Sockets Layer -Protocolo de Capa de Conexión Segura- (SSL) es un protocolo criptográfico que proporciona comunicaciones seguras por una red, comúnmente Internet, implicando autenticación y privacidad de la información entre extremos mediante el uso de criptografía

Userhash: Clave hexadecimal de 16 bytes que identifica de forma única a un usuario de la red Edonkey o Kad. Identifica a la máquina donde se encuentra ejecutando la aplicación independiente de la localización de la misma, es decir de su dirección IP.

APÉNDICE E. GLOSARIO

Bibliografía

[1] Beginning PHP5, Dave W. Mercer, Indianapolis: Wiley.

[2] Domine PHP 5, José López Quijado, RaMa.

[3] PHP y MySQL, Timothy Boronczyk con Martin E.Psinas, Anaya Multimedia.

[4] Desarrollo web con PHP y MySQL : [PHP 5 y MySQL 4.1 y 5], Luke Welling, Laura Thomson, Anaya Multimedia.

[5] MySQL 5,Guía práctica para usuarios. Ed Anaya, J.D. Gutierrez Gallardo.

[6] MySQL, Paul DuBoisPublicación, Anaya Multimedia.

[7] Microsoft Visual Studio 2008 unleashed, Lars Powers, Mike Snell, Publicación Indianapolis.

[8] C++: cómo programar, Harvey M. Deitel, Paul J. Deitel, Pearson Educación.

[9] Enciclopedia del lenguaje C++, Francisco Javier Ceballos Sierra, Rama.

[10] La biblia de HTML, Francisco Charte Ojeda, Anaya Multimedia.

[11] Amule 2.2.3, http://www.amule.org/

[12] XAMPP. http://www.apachefriends.org/es/xampp.html

[13] Microsoft Visual Studio, http://www.microsoft.com/spanish/

[14] MySQl, http://www.mysql.com

[15] PHPMyAdmin. http://www.phpmyadmin.net

[16] Librería wxWidgets, http://www.wxwidgets.org/

[17] Librería Crypto++, http://www.cryptopp.com/

[18] Librería MySQL para C++, http://www.mysql.com/

[19] Librería FPdf http://www.fpdf.org/

[20] API Google Maps, https://google.es/apis/maps

[21] MS-Excel Stream Handler http://www.phpclasses.org/Excel

[22] Curso MySQL con clase, Gestión de BBDD, http://mysql.conclase.net/curso

[23] Curso C++ con clase, http://c.conclase.net/

[24] Conferencias Kademlia http://www.cs.rice.edu/Conferences/IPTPS02/

[25] ¿How does Emule work? http://emuledeaaz.free.fr/en/howworks.html

[26] eMule-Project Monk, Ayuda y soporte http://www.emule-project.net

[27] Foro emule Espana http://www.emulespana.net/foros/