Desanonimización de usuarios en la red social Twitter

Sergio Lucas Navajas

Consultora: Cristina Pérez Solá

Grado de Ingeniería Informática

TFG Seguridad Informática

Enero 2017

A María y a Mía, por ser mis estrellas

A mis padres, porque os lo debía

A Carlos, por ayudarme con los primeros baches de este camino

Resumen

El anonimato en Internet se utiliza en ocasiones para cometer delitos o faltas, por lo que las Fuerzas de Seguridad necesitan métodos y herramientas para la identificación de usuarios maliciosos. En primer lugar, se exploran las técnicas y herramientas utilizadas que permiten ocultar la dirección IP real, así como métodos y opciones para anonimizar la navegación web; también se presenta el uso de buzones de correo electrónico desechables. A continuación, se analiza la información de Twitter que permite obtener algunos rasgos identificativos, para continuar con las técnicas de identificación basadas en la búsqueda cruzada de datos en diferentes redes sociales y foros públicos.

Como conclusión, se realiza una prueba de concepto mediante el desarrollo de un script en Python que, a través de la monitorización de Twitter, permite ver en funcionamiento las técnicas de identificación descritas tomando como fuente de datos las publicaciones en Twitter, tanto las que se obtienen en tiempo real como las que se obtienen del histórico.

Palabras clave: anonimización, desanonimización, Twitter, alias, monitorizar, seguridad.

Abstract

Anonymization on the Internet is used sometimes in order to commit crimes or misdemeanours, which is the reason why Security Forces need methods and tools to identify malicious users. First of all, techniques and tools which allow us to hide the IP address will be explored, as well as methods and options for web browsing anonymization and the use of disposable mailboxes. Furthermore, the Twitter information that allows to get some identifying traits will be analysed to continue with cross finding techniques of identification based on the search, crossed with data in different social networks and public forums.

In conclusion, a proof of concept will be realised through the development of a script in Python which, through the monitorization of Twitter, allows us to see functioning of the described identification techniques, taking publications on Twitter as a source of data, both those obtained in the real time stream and those obtained from history.

Keywords: anonymization, deanonymization, Twitter, alias, monitor, security.

Índice

1. Lista de figuras

2. Introducción

Las redes sociales suponen un medio de comunicación para millones de personas en todo el mundo desde su inicio en la década de los 2000, con una creciente tasa de usuarios a medida que las redes de comunicación mejoran o bien llegan a zonas a las que antes no lo hacían, como los países en vías de desarrollo. Además, este medio de comunicación ofrece un canal entre empresas, y en general entidades, con sus clientes y usuarios. Por otra parte, la gran adopción de *smartphones* y la ubicuidad que suponen, ha conducido a que el tiempo de uso de las redes sociales sea aún mayor.

Para concretar, según el estudio estadístico de IABSpain [1] (con fuentes del INE y otros) en 2015 el número de internautas en España (en el rango de edades entre 16 y 55 años) es de 25,4 millones de personas, de las cuales el 81% utilizan alguna red social, lo que supone 15,4 millones de usuarios de redes sociales. En cuanto al tiempo de conexión, la media se sitúa en casi 3 horas diarias, con un uso principal basado en el envío y recepción de mensajes. A la vista de estos datos, queda patente que la interacción de las personas a través de medios digitales es cada vez mayor.

Entre las principales redes sociales, se encuentra Twitter. Su origen data del 2006, creada por Jack Dorsey, Noah Glass, Biz Stone y Evan Williams. Twitter es una red de *microblogging* en la que los usuarios envían mensajes cortos –llamados *tweets*– de hasta 140 caracteres, que otros usuarios (seguidores) pueden leer e incluso reenviar produciendo así un efecto altavoz. Según fuentes oficiales de Twitter [2] el número de usuarios activos en todo el mundo el tercer trimestre del 2016 ha sido de 317 millones, con una media de 6000 tweets por segundo a nivel global. A pesar de que estas mismas fuentes confirman que existe una tendencia de descenso en el número de usuarios, el uso de esta red social sigue siendo enorme.

La seguridad en las redes sociales es un tema con una preocupación social muy alta, ya que es un reflejo de la realidad, pero a diferencia de las relaciones en persona, internet permite un grado de anonimidad muy alto, de modo que en muchos casos es difícil conocer la autoría de muchas comunicaciones, y también de la persona o personas que se encuentran detrás de un usuario concreto en alguna de las redes sociales. En ocasiones es necesario identificar a usuarios que cometen abusos o delitos en las redes sociales, pero es una tarea compleja ya que habitualmente utilizan técnicas para ocultar su rastro, además de perfiles falsos, que dificultan conocer el origen de una conexión.

A lo largo de este trabajo se investigarán técnicas de anonimización existentes con el fin de analizarlas y explorar métodos que permitan de-anonimizar usuarios en la red social Twitter a través de las herramientas que proporciona la red social en forma de API (del inglés *application programming interface*) pública, para obtener datos. Además, mediante el análisis de esos datos se comprobará hasta qué punto es posible obtener información de los mensajes o tweets que permitan identificar cuál es su fuente de origen, con el fin de poder presentarlos en algún formato que facilite su asimilación. Por lo tanto, la idea principal consiste en automatizar lo máximo posible el proceso de obtención, tratamiento y procesamiento de los datos en forma de herramienta que pueda ser utilizada por algún servicio de investigación.

2.1. Objetivos

Los objetivos específicos que se buscan con el desarrollo de este proyecto son los siguientes:

- Estudiar técnicas de anonimización utilizadas en las redes sociales.
- Explorar técnicas para la identificación y monitorización de usuarios asociados a comportamientos maliciosos.
- Elaboración de un prototipo orientado a la obtención de información identificativa preservando la seguridad del analista.
- Elección de una interfaz de visualización de la información obtenida para que sea consumida y analizada.
- Realización de una prueba de concepto que muestre la aplicación y utilidad del producto.

De manera adicional, se espera alcanzar los siguientes sub objetivos con carácter transversal a todas las áreas del Grado.

- Aplicar los métodos de desarrollo de software adquiridos durante el Grado de Ingeniería Informática, con especial énfasis en las metodologías ágiles; así como las técnicas de trabajo por proyectos.
- Explotar las herramientas y fuentes de información adquiridas a lo largo de todas las asignaturas, para así poder reflejar las relaciones entre las distintas áreas de conocimiento.

2.2. Metodología

Debido a que una buena parte del proyecto va a consistir en aprender una nueva tecnología, la metodología se va a basar, en primer lugar, en la investigación y la búsqueda de información sobre las técnicas de anonimización existentes. Además, será necesario comprender el funcionamiento de Twitter y sus APIs públicas. También será necesario investigar las técnicas o aplicaciones que permiten obtener datos de fuentes de información para su tratamiento y procesamiento.

En segundo lugar, para el diseño y desarrollo de la prueba de concepto la metodología a usar será, en la medida de lo posible, un acercamiento al estilo *bottom-up* en el que se irá diseñando, programando y ajustando de forma ágil para poder adaptar el código a los requisitos planteados en el diseño y las modificaciones que puedan surgir con el fin de mejorar el producto final.

Por otra parte, el desarrollo de la memoria será progresivo, a medida que se avance por las distintas etapas. De esta forma, en la última fase solo será necesario ensamblar las distintas partes, darlos el formato y estilo adecuado para su presentación final.

2.3. Planificación temporal

La planificación temporal se presenta mediante el gráfico con el detalle de las fechas de inicio y fin, así como la duración en horas de cada una de ellas. También se muestra el detalle y descripción de cada una de ellas.

2.3.1. Diagrama de Gantt

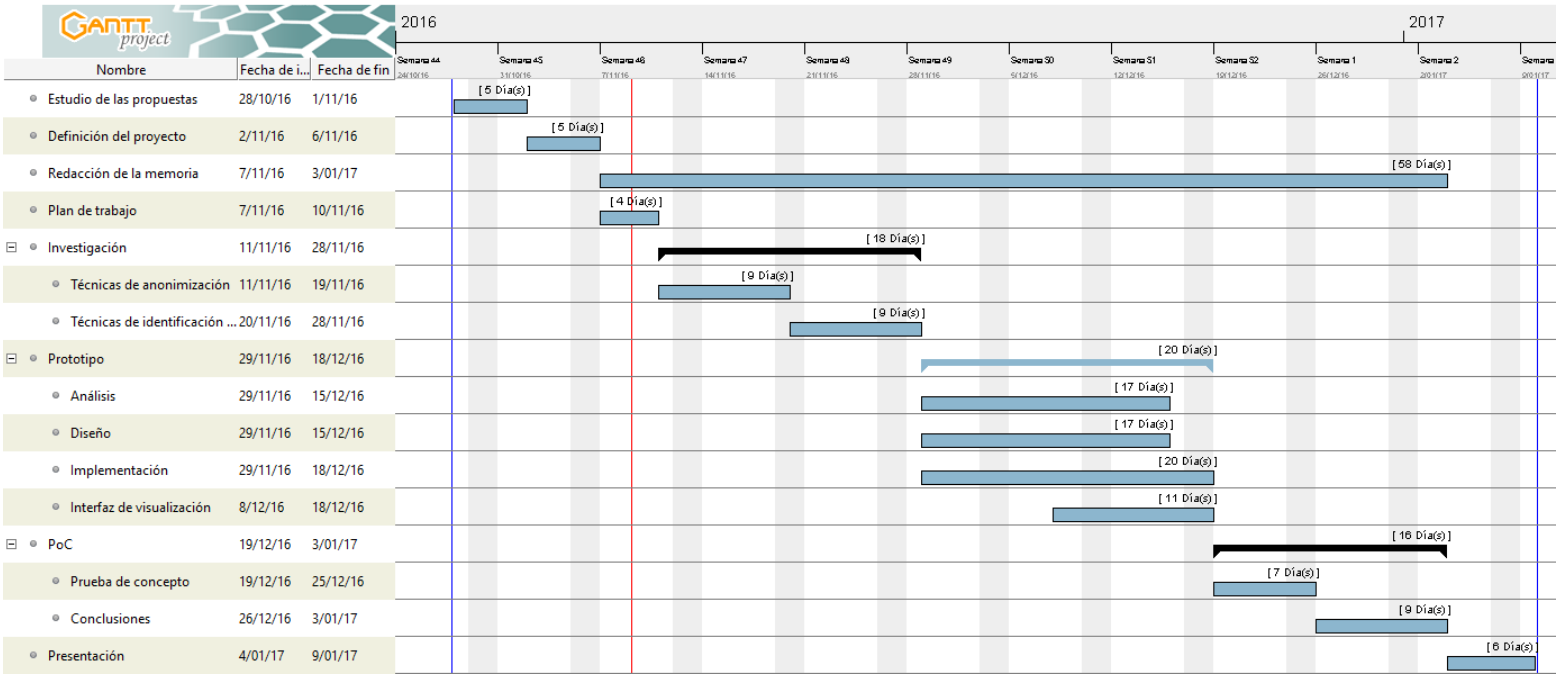


Figura 1. Diagrama de Gantt. Fuente: Gantt Project

2.3.2. Tareas

- **Estudio de las propuestas (5 días):** Revisión de las propuestas propias y de aquellas facilitadas por el área de seguridad informática. En esta fase se decide sobre el tema principal del proyecto a desarrollar.

- **Definición del proyecto (5 días):** Elección del objetivo específico del proyecto. El resultado de esta etapa marca, en principio, cuál será el problema a resolver por el proyecto y la prueba de concepto.

- **Redacción de la memoria (58 días):** La redacción la memoria comienza con el plan de trabajo, y será un constante trabajo durante todas las etapas que culminará con la maquetación final. El resultado de esta tarea será la memoria final del TFG (trabajo de fin de grado).

- **Plan de trabajo (4 días):** Explicación detallada del objetivo, en la que se detalla el conjunto de procesos a realizar, así como la metodología y un conjunto de hitos establecidos en el tiempo. El producto resultante servirá de guía durante todo el trabajo.

- **Investigación (18 días):** Una gran parte de este proyecto consiste en la investigación de las técnicas que permiten mantener el anonimato en internet.
  - Técnicas de anonimización (9 días): Investigación sobre las técnicas de anonimización usadas en internet en general, y en particular en las redes sociales.
  - Técnicas de identificación y monitorización (9 días): Investigación sobre técnicas para identificar usuarios en los distintos niveles de la pila de protocolo de Internet, así como de técnicas y herramientas que permitan la monitorización.

- **Prototipo (20 días):** Desarrollo de un prototipo Debido a que se trabajará con una metodología ágil, las etapas del desarrollo se irán revisitando de forma continua hasta la versión final.
  - Análisis (17 días): Recogida de requisitos según el planteamiento de alto nivel.
  - Diseño (17 días): Creación del modelo en lenguaje natural o algorítmico, junto a los diagramas necesarios.
  - Implementación (20 días): Programación en el lenguaje específico, con el resultado final del código fuente de la aplicación.
  - Interfaz de visualización (11 días): Selección de un interfaz de visualización que permita consumir los datos obtenidos de una forma adecuada.

- **PoC (16 días):** Creación y desarrollo de una prueba de concepto que muestre el funcionamiento del prototipo generado.
  - Prueba de concepto (7 días): Desarrollo de la prueba en un entorno definido y sobre datos reales.
  - Conclusiones (17 días): Extracción de conclusiones del modelo que permitan comprobar si cumple con el objetivo y cuál es el alcance real.

- **Presentación (6 días):** Desarrollo del material audiovisual que sirva como síntesis de todo el trabajo.

3. Estado del arte

La privacidad y el anonimato en Internet son temas de los que últimamente se habla mucho en los medios de comunicación debido a recientes filtraciones por parte de distintas fuentes (véase caso Snowden [4]), que han puesto de manifiesto como algunos gobiernos, agencias o entidades han estado utilizando métodos muy sofisticados para monitorizar comunicaciones y accesos a distintos servicios. Por estos y otros motivos, cada vez son más usuarios los que usan y demandan métodos que salvaguarden esta privacidad y permitan mantener el anonimato en la red cuando así lo quieran.

Por su parte, los datos ofrecidos por la Europol en el informe “Internet Organised Crime Threat Assessment 2016 (IOCTA)” [5] indican que la mayoría de cibercriminales utilizan métodos para anonimizar la dirección IP. En este sentido, la herramienta más utilizada es el proxy (servidor intermediario entre dos extremos de una comunicación) seguido de la red TOR [6] y los servicios de *Virtual Private Network* (VPN), ya sean públicos y ofrecidos por empresas legales, o bien aquellas ofrecidas por redes criminales como un servicio (CaaS o Crime as a Service). También aparecen usos en algunos casos de la red I2P [7] (Invisible Internet Project), que es otra red de anonimización.

Son muchos los datos que permiten identificar a un usuario en la red, más allá de los evidentes como pueden ser la dirección IP, que un ISP (Internet service provider) podría mapear hasta el punto de conexión. Un servidor web puede recolectar mucha información sobre el cliente que se conecta a través de los distintos métodos o funciones que el navegador ofrece, con el fin de proporcionar una mejor experiencia de navegación. En este sentido, existen iniciativas como “Am I unique?” [8] o el proyecto Panopticlick [9] de la Electronic Frontier Foundation (EFF) que estudian la huella o *fingerprint* que el navegador web de un cliente deja en un servidor web. De esta forma, pretenden demostrar que sería posible identificar a un usuario a pesar de que hubiera puesto medios a nivel de red o de transporte para mantener su conexión anónima.

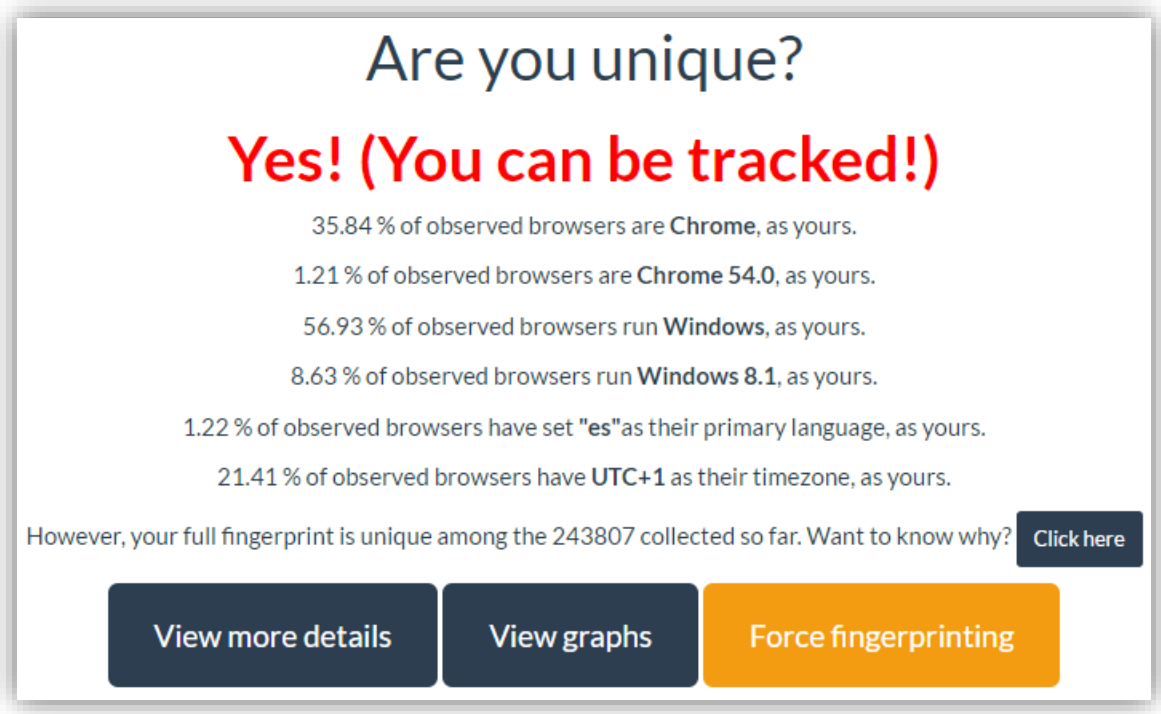


Figura 2 Captura de pantalla del test "Am I unique?"

Por otra parte, hay varios trabajos sobre las técnicas para la re identificación de usuarios en conjuntos de datos que han pasado por un proceso de anonimización, como el estudio presentado por Narayanan y Shmatikov [10] en el que a partir de un conjunto de datos anonimizados de Twitter fueron capaces de re identificar a los usuarios, con el apoyo de otro conjunto de datos de la red Flickr con un error del 12%. En este mismo sentido, Nilizadeh, Kapadia y Ahn [11] desarrollaron un algoritmo con una aproximación mediante la metodología de "*divide and conquer*" que consigue mejorar la re identificación, también en conjuntos de datos de redes sociales.

Un ejemplo más concreto y enfocado en la información que proporciona una única red social es el trabajo de Wondracek, Holz y otros [12] en el que en lugar de usar grandes conjuntos de datos, se utilizan los datos que proporciona la propia red social que se analiza. De esta forma, demuestran que es posible identificar a una persona a partir de los grupos a los que pertenece su usuario en la red social. Es decir, la pertenencia a los grupos diferencia a unos usuarios de otros hasta el punto de formar una huella única o al menos común a un conjunto pequeño.

4. Técnicas de anonimización

Las comunicaciones en internet se basan en el protocolo –más bien conjunto de protocolos o pila– TCP/IP. Los identificadores de paquete que intervienen en cada capa del protocolo tienen un ámbito distinto. Sin embargo, es el protocolo IP el que permite seguir una conexión hasta su origen. De esta forma, la identificación de cada host en internet se realiza a través de la dirección IP única que permite dirigir paquetes hasta ese destino. Normalmente, un ISP (Internet service provider) proporciona a cada suscriptor, sea este un usuario final, una organización pública o una empresa privada una dirección IP única.

Sin embargo, aunque la dirección IP podría permitir identificar a un usuario, o al menos la red desde la que se ha conectado, existen muchos otros datos que podrían identificar a un equipo final. Si atendemos a nivel de acceso en el protocolo TCP/IP, la dirección MAC (*Media acces control*) utilizada por muchos protocolos es única y puede permitir identificar al equipo que la utiliza. Por otra parte, a nivel de aplicación y en función del protocolo, también es posible obtener información que identifique o al menos reduzca considerablemente el anonimato.

Por lo tanto, las técnicas principales de anonimización en internet pasan, en primer lugar, por ocultar la dirección IP real del equipo que quiere mantener el anonimato. Es este sentido, existen diferentes soluciones en función del nivel del protocolo TCP/IP en el que se trabaje. Si atendemos al nivel de red, se suelen utilizar VPNs. Mientras que si se trabaja en el nivel de aplicación sobre contenidos web, la solución más habitual suelen ser los proxy HTTP. Además de la ocultación de la dirección IP, se suelen utilizar otros métodos que evitan el rastreo de información a través del navegador para sortear otras técnicas avanzadas de identificación.

En los siguientes puntos se explorarán las principales técnicas usadas por cibercriminales, según el informe de la Europol - Internet Organised Crime Threat Assessment 2016 [5], que son: VPN, proxies, la red TOR y la red I2P; así mismo, se mostrarán las características de los navegadores que permiten crear un perfil del sistema utilizado y que lo puede diferenciar de cualquier otro.

4.1. VPN

Como ya se ha mencionado con anterioridad, las VPN o *virtual private network* por sus siglas en inglés, son redes privadas virtuales, es decir que se establece una red privada de manera virtual sobre una red pública –como internet– que permite comunicaciones de tipo local en redes distantes. Debido al auge y mejoras de las infraestructuras de Internet, el uso de las VPN está muy extendido en la actualidad en las empresas para unir redes en sedes que están distantes, o bien para permitir el acceso a recursos de la intranet a usuarios remotos.

La mayoría de las implementaciones de las VPN se basan en tres protocolos, cada uno ofrece ciertas ventajas y desventajas:

- IPSec: Protocolo de internet seguro, que ofrece encriptación de la información. Ofrece dos modos de trabajo, aunque para el uso en VPN se utiliza el modo túnel.
- PPTP: Protocolo de punto a punto, ofrece también encriptación de la información, está muy extendido gracias a su inclusión en sistemas Windows. Sin embargo, existen ciertas vulnerabilidades [13] que desaconsejan su uso.
- SSL-TLS: Protocolo de seguridad a nivel de transporte. Gracias a las propiedades de encapsulación del protocolo TCP/IP es posible encapsular paquetes TCP/IP y enviarlos a través de un túnel SSL.

Una de las principales ventajas del uso de VPN frente a otras soluciones es que trabaja a nivel de red, es decir, que permite el uso de cualquier protocolo de transporte (TCP, UDP) y por extensión cualquier protocolo de aplicación.

No obstante, el esquema de uso de las VPN para mantener el anonimato es algo diferente al utilizado por organizaciones que necesitan unir redes, ya que no se utilizan para compartir recursos como podría ser en una red local. En su lugar, el cliente se conecta al servidor VPN y envía y recibe todo el tráfico IP a través del túnel creado. Por su parte, el servidor VPN procesa los paquetes IP, y los reenvía al punto de destino con su IP de *Gateway,* de modo que actúa de intermediario ya que cuando recibe la respuesta, reenvía el paquete IP de vuelta al emisor original. De esta forma, para el *host* de destino, la comunicación ha sido originada por el servidor VPN.

De esta forma, para identificar a un usuario o equipo final a través de la IP ya no se podría recurrir al proveedor de servicios de internet, ya que solo se verían conexiones al servidor VPN, sin conocer el destino real al que van dirigidas. Sin embargo, sería factible que el servidor VPN pudiera ofrecer información sobre la conexión, de modo que se relacione la IP de origen (cliente) con la IP de destino (servidor). En este sentido, son muchos los servicios VPN que declaran no guardar ningún tipo de registro sobre las conexiones de los clientes, de modo que tampoco sería posible obtener por esta vía la identidad correspondiente a una conexión.

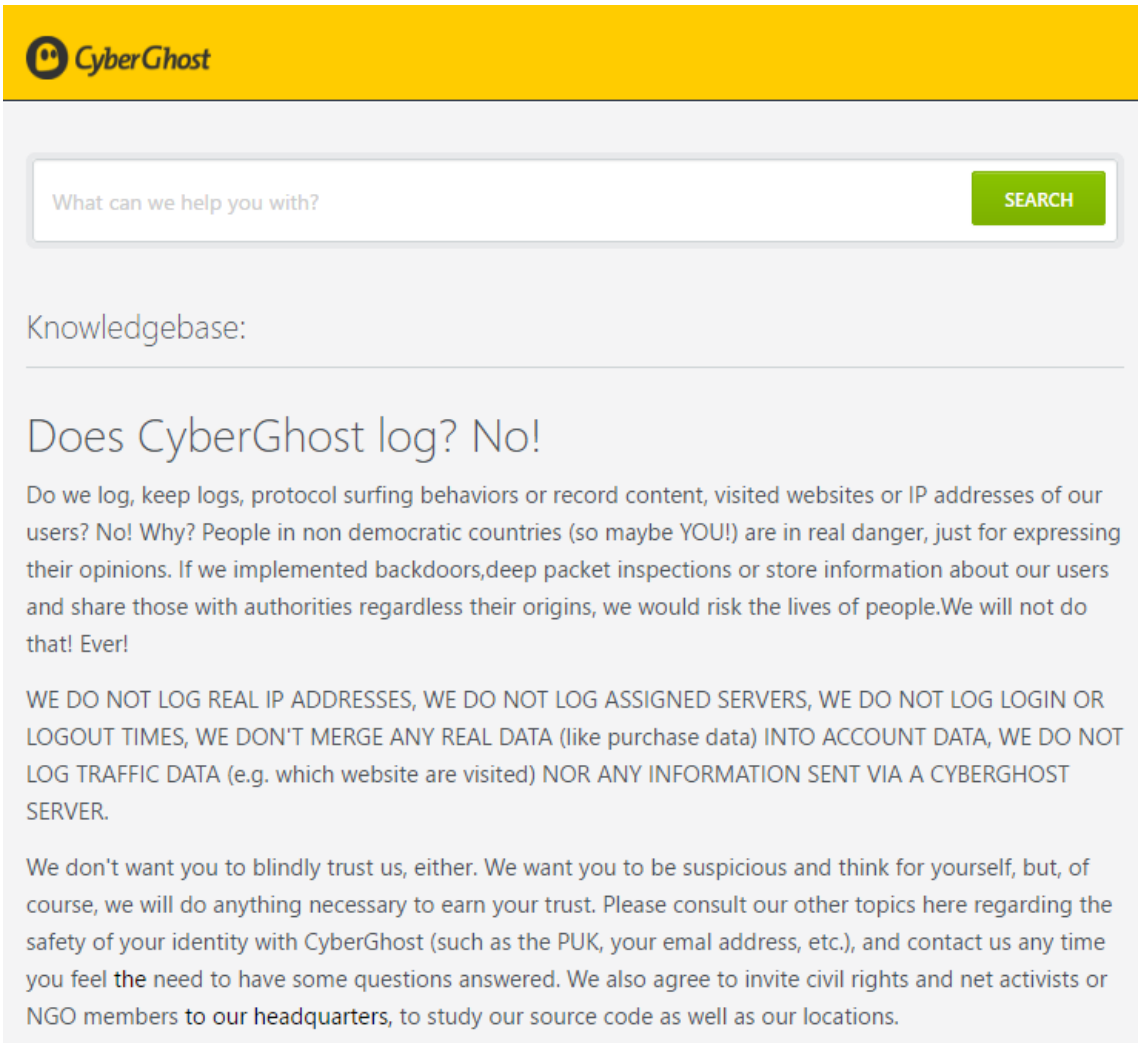


Figura 3. Captura de declaración sobre logs de CyberGhost VPN

4.2. Proxy

Un servidor proxy (intermediario en inglés) es un servicio –puede ser software o un equipo diferente– que se interpone entre los dos extremos de una conexión y que realiza o maneja las peticiones en nombre de quien origina la conexión. Es habitual que trabajen a nivel de aplicación en la pila del protocolo TCP/IP, y habitualmente son específicos para un protocolo en concreto. Su uso más extendido es el proxy HTTP o proxy web que canaliza todas las peticiones http de los clientes y origina nuevas conexiones hacia el servidor. Cuando las peticiones son atendidas, el servidor proxy recibe las respuestas y las encamina hacia el cliente.

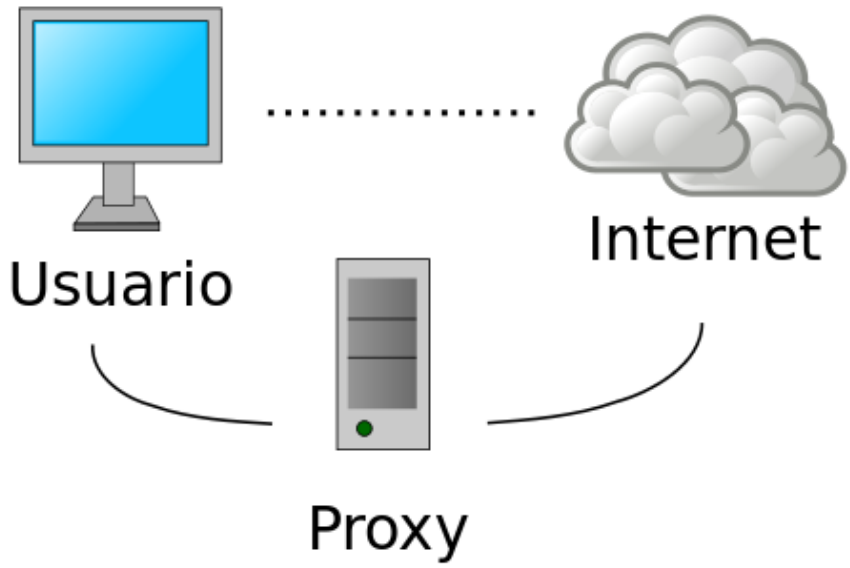


Figura 4. Esquema general del proxy. Autor Randomicc (Own work) [CC BY-SA 3.0]

El uso de proxies es muy habitual en redes grandes en las que es necesario compartir una única salida hacia internet. De esta forma, todas las conexiones de los clientes de dentro de la red quedan anoninimizadas ya que de cara hacia fuera todas se originan desde la misma IP. Otra característica que pueden ofrecer –si se utilizan técnicas de autentificación– es la de filtrado de contenidos y perfiles de navegación que permiten asignar roles a los usuarios con el fin de gestionar los recursos de manera óptima.

Para el caso particular de anonimización de conexiones en internet, existen diferentes proveedores –muchos de ellos gratuitos– que permiten la navegación web a través de un proxy, ya sea mediante la redirección del tráfico http y https directamente desde el equipo, o bien a través de un acceso a una web intermediaria que permite consultar la página final, actuando de proxy.

Sin embargo, la utilización de proxies limita el uso de servicios que se pueden utilizar, y por extensión, también limita el número de aplicaciones que se pueden usar de forma anónima, ya que como se ha dicho con anterioridad, la mayoría de proxies están especializados en el protocolo HTTP. No obstante, muchas aplicaciones y servicios permiten el encapsulado de protocolos diversos (como por ejemplo el protocolo de transferencia de archivos FTP) en paquetes HTTP que extienden las funcionalidades de los proxies web.

4.3. Red TOR

La red TOR (The Onion Router) es un sistema distribuido de equipos que permite anonimizar conexiones mediante el uso de técnicas de criptografía –para mantener la privacidad de la información– y la elección de caminos o rutas aleatorias a través de su red. Fue creada en 2003 por Roger Dingledine y otros a partir del proyecto Onion Routing. Actualmente es usada en muchos lugares donde existe censura, donde la libertad de expresión está coartada o bien donde es necesario mantener el anonimato

por seguridad. Por otra parte, también es usada con fines criminales en parte de la deepweb.

La red funciona como un conjunto de nodos que permiten crear rutas distintas para cada conexión, de modo que, en principio, no es posible monitorizar la red para tratar de discernir las conexiones entre los extremos. La estructura de la red está formada por los siguientes elementos:

- Nodos OR (Onion Router) que son los encargados de transmitir los paquetes a través de la ruta creada por el host que inicia la comunicación.
- Nodos OP (Onion Proxy), hacen de interfaz entre el cliente y la red TOR. Se encargan de obtener información del servicio del directorio, crear las rutas y gestionar las conexiones de las aplicaciones de usuario (paso de TCP a SOCKS).
- Servicio de directorio, ofrece información –de forma centralizada, aunque está replicado– sobre los nodos.

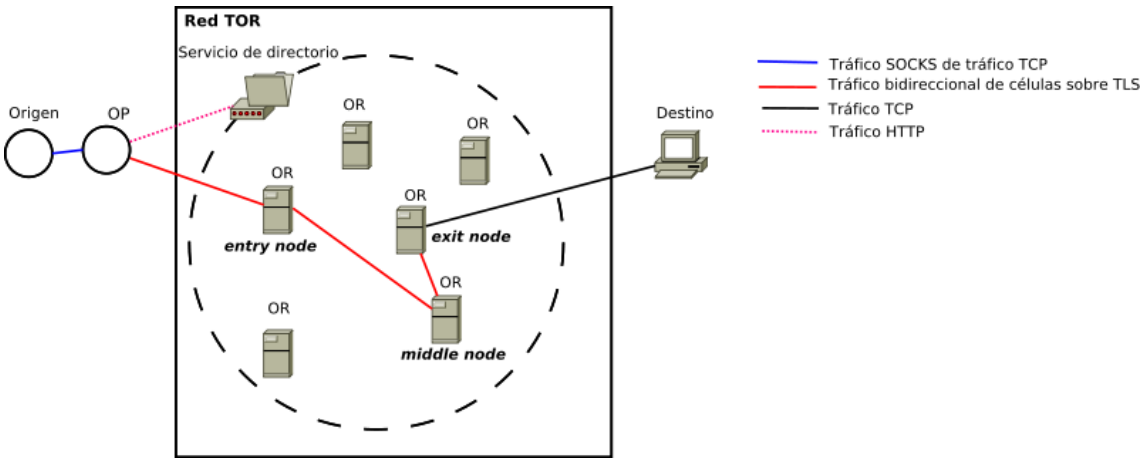


Figura 5. Esquema de la red TOR. Autor Fercufe. Fuente Wikimedia. CC BY-SA 3.0

El funcionamiento es el siguiente: El nodo OP (normalmente la aplicación en el sistema cliente) se pone en contacto con el servicio de directorio, con esta información crea una ruta y negocia las claves de sesión con cada uno de los nodos del circuito creado. El cliente va cifrando y empaquetando la información con la clave de cada nodo en sentido inverso al camino que va a seguir (empieza por el último nodo) formando las distintas capas que dan el nombre por la analogía de la cebolla. Después, envía los paquetes al nodo de entrada, que descifra la primera capa y pasa el paquete al siguiente nodo. El último nodo –el nodo de salida– establece una comunicación a través del protocolo TCP con el destinatario, y con todos los paquetes de respuesta hace el proceso contrario de encriptar y pasar al siguiente nodo.

En general, la red TOR permite mantener el anonimato de una forma muy eficiente, siempre que todas las conexiones (incluidas las peticiones DNS) se hagan a través de TOR, de lo contrario podrían monitorizarse las peticiones al servidor y re-identificar al cliente. Sin embargo, complementos o *pluggins* de los navegadores (por ejemplo Flash) pueden establecer conexiones directas a servidores sin pasar por la red TOR, lo que podría comprometer el anonimato. También se ha hablado de otros ataques que se

basan en el control de los nodos de salida, pero si el número de nodos es lo suficientemente alto, es difícil de llevar a cabo.

4.4. Red I2P

La red I2P (Invisible Internet Project) es una red de equipos distribuidos y descentralizados que actúan como pares iguales entre ellos. Las conexiones están cifradas de extremo a extremo, e incluyen cuatro niveles o capas de cifrado de la información. Es un proyecto de código abierto iniciado en 2003, es multiplataforma y actúa a modo de red superpuesta, es decir que usa la red Internet como infraestructura subyacente y como punto fuerte, ofrece compatibilidad con los protocolo TCP y UDP (al contrario de TOR que solo permite TCP). Además, ofrece servicios en la propia red como alojamiento y navegación web, canales de chat, etc.

En la red I2P todos los participantes actúan como nodos y son capaces de transferir tráfico a otros nodos. El directorio de nodos disponible se basa en tablas distribuidas (DHT) al igual que otros protocolos P2P. El protocolo para la comunicación se basa en la creación de túneles para el envío de datos y túneles distintos para la recepción de los datos. De esta forma el cliente crea una ruta de ida distinta a la ruta de vuelta, lo que hace que sea más resistente a cierto tipo de ataques como el análisis de tráfico.

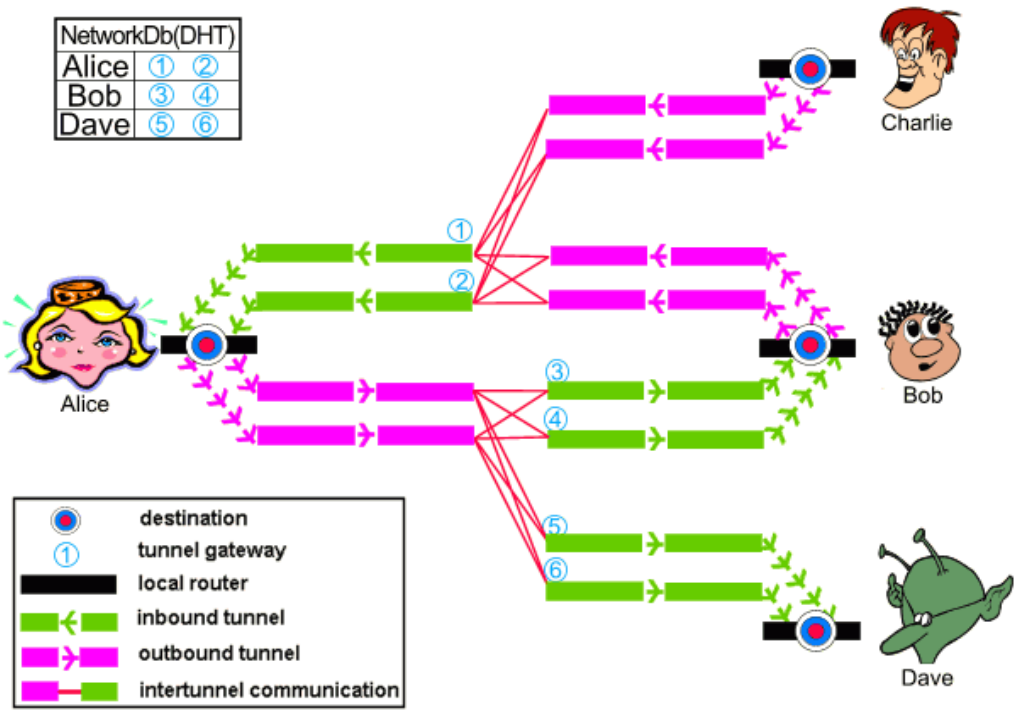


Figura 6. Esquema de funcionamiento de la red I2P. Autor: I2P Team. Fuente Wikimedia. CC-BY-SA 4.0

Cada usuario que quiere comunicar mediante la red I2P crea, al menos, un túnel de entrada y un túnel de salida. Se pueden crear varios túneles de entrada y salida –para

mejorar el rendimiento– que serán compartidos por los distintos flujos de datos. Cada túnel está compuesto por dos nodos. Los datos de salida pasan de la aplicación en el cliente al router local I2P que inicia el cifrado (se usa el protocolo Garlic que permite empaquetar más de un datagrama IP en cada paquete) y pasa los datos al túnel de salida. Los datos viajan hasta el punto de salida del túnel en el segundo nodo, y a partir de ahí van hacia la entrada del túnel o túneles de entrada del destinatario o servidor. De esta forma, a través de los dos nodos que forman el túnel de entrada, los paquetes de información llegan hasta el router local del destinatario que serán pasados a su aplicación correspondiente.

La red I2P se creó como una red en sí misma y no como un proxy que ofrece privacidad y anonimato hacia el internet público, por lo tanto tiene integrados dentro de su propia red muchos de los servicios habituales en internet. No obstante, es posible utilizarla para navegar y usar casi cualquier servicio que se encuentra en la internet pública o *Surface web* a través de los Outproxy que hacen de puerta de enlace anónima. Sin embargo, hay un número limitado y no muy alto de estos servidores de salida, por lo que puede ser vulnerable a ataque de análisis de tráfico que permitan.

4.5. Anonimización del navegador

A nivel de aplicación también se usan distintas técnicas para evitar el rastreo y posible identificación de los usuarios. Dentro del protocolo HTTP, en las peticiones o respuesta se incluyen una serie de datos que ofrecen información específica del cliente, y que en conjunto puede suponer una huella digital del dispositivo, y por extensión del usuario. Cada vez que se accede a una página web, se establecen conexiones con varios servidores con distintos fines (normalmente para obtener datos de uso). En la siguiente figura, se muestra una captura de LightBeam que permite ver las conexiones extra (los triángulos) cuando se visita una página web determinada (los círculos).

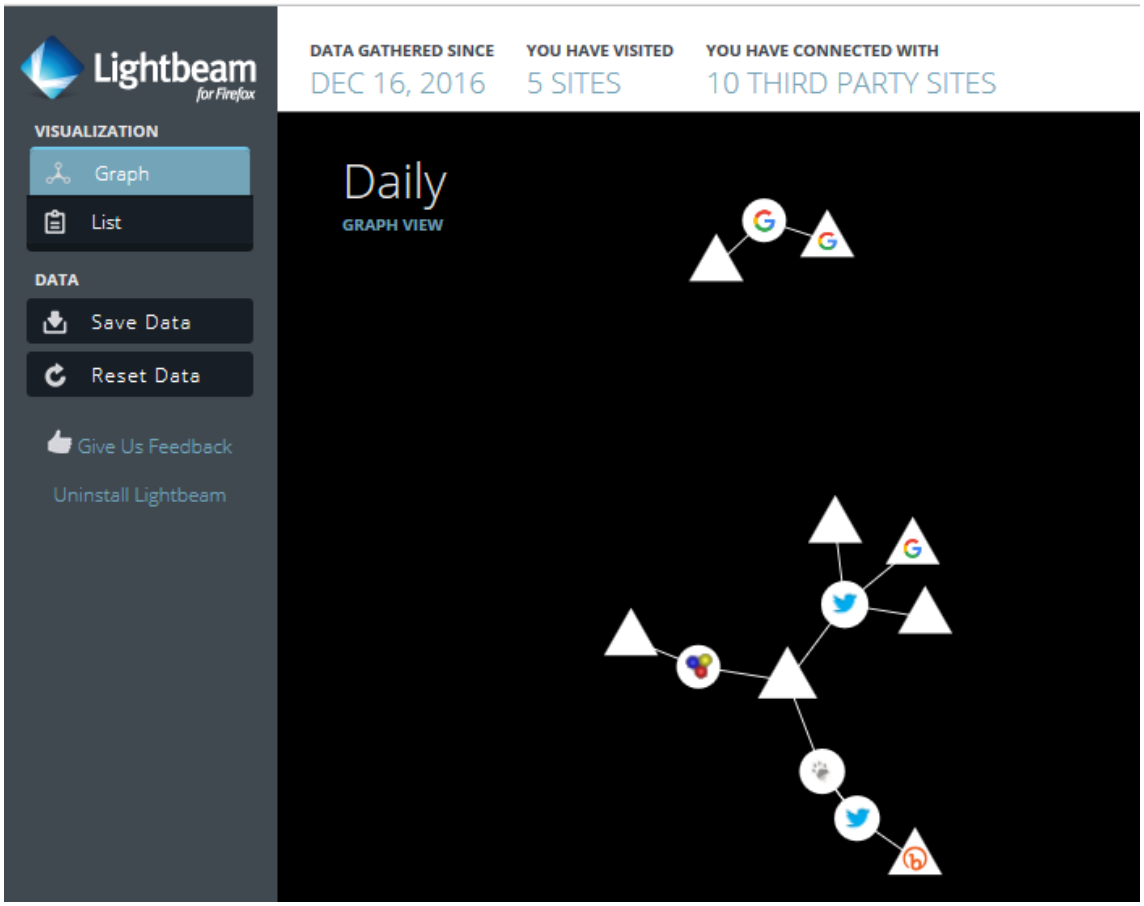


Figura 7. Captura de pantalla del complemento de Firefox LightBeam

Cabeceras HTTP

Algunas de los campos o cabeceras del protocolo HTTP que ofrecen información variada y que se envían en cada petición son:

- Do not Track: permite, de manera explícita, indicar al servidor que no se desea que se realice un seguimiento de la navegación, y que no se comparta con terceros. Sin embargo, la decisión de hacerlo o no recae sobre el servidor, de modo que no se puede tener la certeza de que se cumpla. Al igual que otras opciones, la mayor parte de navegadores modernos, ya sean de PCs o de dispositivos móviles permiten habilitar esta opción.

- Accept: Indica qué tipo de contenido acepta el navegador codificado en formato MIME, desde texto plano, a imágenes y contenido para complementos, etc.

- User-Agent: La cadena “User-Agent” es una de las cabeceras habituales que se envía en una petición HTTP, y que informa al servidor de multitud de información sobre el cliente: sistema operativo, arquitectura del sistema, modelo y versión del navegador, etc. El propósito de esta información es poder ofrecer al cliente una versión adecuada del contenido web. No obstante, debido a la variedad de combinaciones de sistemas y navegadores, el agente de usuario se convierte en

una huella digital con cierta precisión a la hora de identificar a un equipo, e incluso permite saber si se trata de un dispositivo móvil o un equipo de escritorio.

- Accept-Language: Indica los lenguajes que acepta el navegador; tienen un orden de preferencia, de modo que se envía cuál es el lenguaje principal para que si el servidor tiene distintas versiones del contenido, se envíe la más adecuada.

Un método para evitar la identificación a través de estos campos es no enviarlo, o bien falsear (*spoofing*) la información que se envía. Con este fin, existe una gran variedad de complementos que permiten simular un agente de usuario determinado (se podría usar uno muy común en base a estadísticas) o bien cambiarlo de forma aleatoria en cada conexión.

Cookies

El protocolo HTTP es un protocolo sin estado por definición, es decir, cada transacción o conexión se trata como si fuera nueva, aunque tenga que ver con una operación anterior. Para poder añadir las propiedades de un protocolo con estado, se crearon las *cookies* que son archivos que se almacenan en el equipo del cliente, y que el servidor puede consultar y/o actualizar; estos archivos ofrecen información al servidor sobre conexiones anteriores. También pueden demostrar que un equipo accedió a un servicio remoto, por lo que tienen propiedades identificativas.

La mayoría de navegadores web modernos permiten desactivar el uso de cookies, lo que aporta un grado más en la privacidad de las conexiones. También existen complementos para los navegadores que realizan esta función, e incluso permiten conocer qué información se guarda en las cookies.

JavaScript

JavaScript es un lenguaje de programación interpretado que se utiliza –habitualmente– para producir o manejar contenido dinámico en la navegación web a través de la ejecución de scripts o conjunto de comandos en el lado del cliente. Como puede ejecutar código en el equipo del cliente, se puede utilizar para obtener información del mismo, que puede permitir la identificación, o al menos colaborar a crear la huella digital.

Para mejorar la navegación anónima, es posible deshabilitar la ejecución de JavaScript de casi cualquier navegador, de modo que no se permita ofrecer esa vía. Si bien, la experiencia o la navegación en muchos sitios puede no funcionar correctamente, al menos se consigue poner una barrera para que no se pueda conseguir saber qué complementos tiene activados el navegador, que configuraciones admite, como la resolución de pantalla y otras características que suman datos a la huella digital.

Complementos (plug-ins)

Los complementos de los navegadores permiten extender las funcionalidades básicas que ofrecen de forma que permiten interpretar distintos tipos de contenido que no están soportados por los estándares del protocolo HTML. Un ejemplo muy extendido –aunque cada vez más en desuso– es el complemento de Adobe Flash que ha permitido ejecutar contenido enriquecido como vídeos y animaciones. Sin embargo, los complementos pueden realizar conexiones a servidores sin el control del usuario; por ejemplo, el complemento Flash puede realizar conexiones directas a un servidor cuando se está trabajando con el navegador TOR, de forma que podría mostrar la IP real del usuario.

La anonimización del navegador pasa por desactivar todos los complementos, o al menos mantener un control detallado de cada uno de ellos para evitar fugas de información o conexiones paralelas que pueden evidenciar la identidad.

4.6. Correos electrónicos desechables

El correo electrónico supone una fuente para la identificación de usuarios, además de que el acceso a su contenido, sea por vía legal mediante una orden judicial o por otros medios que supongan comprometer la seguridad, puede revelar mucha información. Además de los servidores de correo privados (de corporaciones o administraciones) y los servidores de correo públicos (Gmail, Hotmail, etc.) existen proveedores de correo electrónico que ofrecen cuentas desechables o DEA (*disposable email address*)

Las cuentas de correo desechables son direcciones de correo normales, con la peculiaridad de que tienen una vigencia determinada, de modo que cuando llega ese momento la cuenta y su contenido son destruidos. Algunos ejemplos de este tipo de servicio son:

- Guerrilla Mail: permite crear cuentas de varios dominios y alias aleatorio, cada sesión dura 60 minutos.
- Temp Mail: Las cuentas son destruidas cuando el usuario quiera, sin límite de cuentas.
- Trash Mail: Las cuentas que no se usen en tres meses son destruidas.

4.7. Sistemas operativos

Existen algunas distribuciones de software –sistema operativos junto con aplicaciones– que tiene como objetivo mantener el anonimato en la red, a todos los niveles posible, desde la capa de internet a la capa de aplicación. Este tipo de sistemas va más allá incluso al permitir la ejecución desde una unidad de almacenamiento externa como un *pendrive* o un CD o DVD. De esta forma, el sistema operativo y toda la sesión se ejecutan desde la memoria RAM del ordenador, sin guardar ningún dato ni el soporte que lo contiene ni en la unidad de almacenamiento interna (HDD, SSD o cualquier otra).

La distribución más extendida y completa en este sentido es Tails [14]. Es una distribución Linux basada en Debian, con GNOME como sistema de escritorio. Por defecto utiliza la red TOR para todas las comunicaciones –aunque también dispone de acceso a la red I2P–, con el navegador TOR también como opción de navegación predeterminada (con opciones para protección de elementos de JavaScript, solicitud de versión HTTPS de las páginas web, y AdBlock Plus para evitar el rastreo y publicidad) y una completa suite de herramientas de cifrado para archivos, mensajería instantánea y correo electrónico.

4.8. Otros

Ya se han descrito las principales técnicas y redes que permiten mantener el anonimato mediante la ocultación de la dirección IP real, y que son las nombradas por la Interpol en su último informe IOCTA, además de aquellos que proporcionan información que ayuda a construir una huella digital del dispositivo y sistema que se utiliza. Sin embargo, y aunque menos sofisticados, existen otros métodos que permiten que no se pueda identificar al equipo o usuario que se ha conectado a cierto servicio.

En este sentido, el uso de puntos de acceso públicos que no requieren autentificación (como redes WiFi sin encriptación) o aquellos que sí utilizan protocolos de encriptación (WEP, WPA), pero que no realizan ningún registro de las conexiones, como pueden ser puntos de acceso en cafeterías, bibliotecas públicas y otro tipo de establecimiento puede servir para este fin. Así, este tipo de redes, como la mayoría, utilizan NAT (Network Address Translation) para la salida hacia internet, de modo que se podría conocer la red desde la que se ha realizado la conexión (IP pública), pero podría llegar a ser muy difícil saber quién era el usuario (IP privada o dirección MAC) sin otros métodos (como cámaras de seguridad u otros).

En el caso de dispositivos móviles o tabletas (o en general cualquier dispositivo que tenga un receptor de señales de posicionamiento tipo GPS/GLONASS/GALILEO), para evitar que se envíen datos relativos a la posición geográfica, se puede deshabilitar esta característica. En ocasiones, la información sobre la localización no forma parte de los datos explícitos que se manejan, no obstante, puede suponer un riesgo para el anonimato y la privacidad (ver caso John Mcfee [15]) ya que es habitual que la localización se incluya como metadatos en las fotografías.

5. Técnicas de identificación

Las técnicas de identificación tratan de discernir, o al menos obtener más información, sobre la persona que se encuentra detrás de un pseudónimo o alias. Lo ideal sería obtener la dirección IP de la conexión, sin embargo gracias a los métodos citados en el apartado anterior, es posible que la IP no identifique al usuario. En el caso de twitter, existen diferentes campos en los tweets y en el usuario asociado a cada uno de ellos que pueden dar algo de información. Además, existen métodos que se basan en la búsqueda cruzada por alias en otras plataformas que quizá puedan dar más información. Por otro lado, conseguir una cuenta de correo asociada a algún usuario malicioso puede ser de gran ayuda para las fuerzas de seguridad.

5.1. Datos obtenidos de Twitter

La red social Twitter ofrece una API muy completa de la que se puede sacar mucha información. La estructura principal está formada por cuatro objetos que se relacionan entre sí:

- Tweets: Es el objeto fundamental que forma los tweets o micropublicaciones. Cada tweet tiene un identificador (ID) único, y además del texto del propio tweet tiene otros atributos asociados como el autor, la marca de tiempo, lenguaje, coordenadas, etc.
- Users: Cada cuenta dada de alta en la red tiene un objeto de este tipo asociado. Contiene campos como el nombre descriptivo, la fecha de creación, el idioma preferido, si tiene imagen de perfil, los seguidores y los usuarios a los que sigue entre otros.
- Entities: Son objetos que contienen meta datos de los objetos tweet y user. Contiene atributos que informan sobre las menciones a otros usuarios (@usuario), sobre los *hashtag* (#etiqueta), sobre los enlaces http y sobre contenido multimedia.
- Places: Objetos que representan lugares que pueden ser incluidos en los tweets. Contiene atributos tales como la dirección, coordenadas, etc.

Los objetos de los que se puede sacar información identificativa, y los que se verán en detalle, son los objetos *tweet* y los objetos *user*, ya que ofrecen ciertos campos o atributos que pueden arrojar algo de información del usuario que hay detrás de cada cuenta.

Twitter ofrece varias formas de acceder a su contenido en función de la información que se desea obtener. Para la consulta de objetos de los tipos fundamentales ofrece una API REST que permite, a través de peticiones HTTP, obtener la información requerida. El formato en el que la información es devuelta es en el formato estructurado JSON, de forma que los datos se pueden tratar de una forma sencilla. Además de este tipo de información, ofrece una API Streaming que permite consumir los tweets que se van publicando en tiempo real; de esta forma, es posible monitorizar los tweets en función de algún filtro, como los tweets de un usuario o bien aquellos que incluyen un *hashtag* determinado.

A partir de un objeto de tipo Tweet, se pueden obtener los siguientes datos relevantes:

`coordinates`: si este campo está informado por el usuario o bien por la aplicación que gestiona la cuenta de Twitter se puede ubicar al usuario en el momento en el que publicó el tweet en cuestión.

`created_at:` marca de tiempo en formato UTC, indica a la fecha y hora a la que se ha publicado el tweet, no se puede modificar.

`in_reply_to_screen_name, in_reply_to_status_id:` si el tweet es una repuesta estos campos informan del nombre de usuario y el id respectivamente del usuario destinatario. Puede ser útil si se está monitorizando la actividad de un usuario y hacia ese usuario.

`lang:` muestra el identificador del lenguaje del equipo desde el que se ha publicado el tweet, en formato BCP 47. Permite obtener al menos algo de información del sistema del usuario.

`place`: contiene un objeto de tipo *places*. Puede indicar que el tweet se publicó desde un lugar concreto lo que ubicaría al usuario. Sin embargo, se puede indicar un lugar cualquiera, por lo que no es vinculante.

`source`: indica la fuente desde la que se ha publicado el tweet. Se podría sacar información sobre qué dispositivos tiene, o al menos utiliza un usuario.

`text`: es el contenido publicado en formato UTF-8. Se puede utilizar para filtrar por palabras clave.

Los atributos más relevantes de los objetos de tipo usuarios son:

`created_at:` Marca de tiempo con la fecha de creación de la cuenta.

`default_profile, default_profile_image`: Indican si se ha modificado el perfil o la imagen por defecto. Pueden arrojar algo de información sobre el tipo de usuario.

`description:` opcional, muestra una descripción sobre el usuario.

`followers_count:` indica el número de seguidores. Da una idea de la popularidad de la cuenta.

`friends_count`: indica el número de usuarios a los que sigue la cuenta.

`geo_enabled:` Si el valor es verdadero los tweets podrán ser localizados mediante coordenadas.

`id:` identificador único del usuario, es un valor entero de 64 bits.

`lang:` identificador de lenguaje indicado por el propio usuario, puede ofrecer información sociocultural del usuario.

`location:` es la localización geográfica declarada por el usuario.

`name:` el nombre de usuario de la cuenta.

`screen_name:` alias con el que se identifica la cuenta. Útil para comparar o buscar usuarios con el mismo alias en otras plataformas.

`time_zone:` zona horaria declarada por el usuario.

Dado un usuario particular, se podría buscar toda la información a través de la función `statuses/user_timeline` (una de las funciones de la REST API que permite obtener hasta los 3200 últimos tweets de un usuario) y tratar de sacar la información disponible de los campos indicados con anterioridad. Si un usuario malicioso no ha tomado las medidas suficientes, es posible que haya pasado por alto algún campo que pueda dar algo de información personal, como por ejemplo la ubicación, la zona horaria o bien se puede tomar el alias de la cuenta, que se podría utilizar para buscar información en otras plataformas.

Por otra parte, con la API Stream se puede monitorizar todos los tweets que se publican, de forma que se podría usar para:

- Buscar tweets por palabras clave, y a partir de los tweets identificar a los usuarios que los publican.
- Monitorizar todos los tweets de un usuario particular, de esta forma si utiliza la geolocalización se podría ver en tiempo real su ubicación o trayectoria seguida mediante la representación en algún servicio de cartografía.
- Establecer relaciones entre usuarios para identificar la pertenencia a un grupo a partir de las menciones o las respuestas.
- Identificar hábitos horarios a partir de las marcas de tiempo de los tweets, de esta forma se podría establecer o descartar determinadas zonas horarias cuando hablamos de usuarios de todo el mundo.

5.2. Coincidencias del alias en otras plataformas

El alias o nombre de usuario es, en muchos casos, una seña de identidad en internet, por lo que muchos usuarios suelen utilizar el mismo alias para cuentas en distintas plataformas. Teniendo en cuanta esta situación, a partir de este dato se pueden buscar perfiles iguales en distintas redes, foros, o en general servicios en internet que tengan un acceso público. A pesar de no ser concluyentes, en cuanto a que no siempre cuentas de distintos servicios con el mismo alias pertenecen al mismo usuario, sí que son un punto de partida para los analistas.

Esta técnica de ayuda a la identificación busca obtener datos de fuentes abiertas que ofrezcan más información que la que se tiene de partida, en este caso de la que ofrece Twitter a través de su API. De esta forma, es posible que otro servicio permita obtener mayor precisión en datos identificativos, bien porque el servicio en cuestión de esta información (como por ejemplo la fecha de nacimiento en la red social Facebook) o bien porque el usuario haya compartido más información en otras partes sin tener en cuenta que alguien podría establecer esta relación.

Para encontrar esta información se pueden recurrir a varios métodos en función de la plataforma concreta que se analice. En muchas de ellas, es posible comprobar si una cuenta existe mediante una simple petición HTTP, ya que el alias se incluye directamente en la URL, como es el caso de Twitter, Facebook y otras:

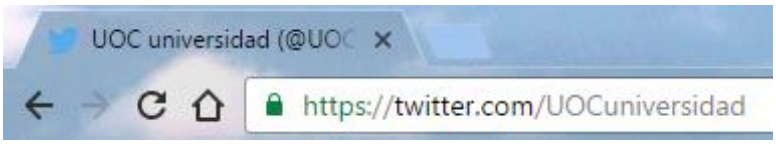


Figura 8. Captura de la URL de una cuenta de Twitter

En otras, sin embargo, la URL no ofrece la información de forma tan clara, o bien es necesario acceder mediante una cuenta valida, y ya una vez dentro se puede buscar por cuenta de usuario.

Es importante que el método utilizado consiga que no se informe al usuario objetivo de que se ha realizado una búsqueda sobre su perfil, ya que de otra forma podría levantar sospechas. Como el objetivo es conocer si la cuenta existe o no, con tener esa certeza es suficiente, sin tener que realizar más indagaciones sobre el contenido de la cuenta.

Existen servicios que permiten comprobar la existencia de un Nick o alias en muchas plataformas a la vez, y con una interfaz web muy amigable, como por ejemplo CheckUserNames [16] o Namech_k [17] que permiten ver si un alias está disponible en las principales redes sociales y foros. Además, ofrecen una API –de pago– con el fin de poder usar el servicio mediante programas o scripts. Por otra parte, también hay disponibles desarrollos de código abierto que permiten la misma funcionalidad, como por ejemplo el paquete OSRFramework de i3visio [18] en Python con licencia GPLv3+, que incluye la utilidad o módulo Usufy que permite realizar una búsqueda por alias en más de doscientas plataformas de distinto carácter.

5.3. Patrones de correo a partir del alias

Al igual que es habitual usar el mismo alias en diferentes plataformas, también es habitual utilizarlo como nombre para cuentas de correo electrónico. De esta forma, el alias se convierte en premisa para la búsqueda en otra fuente de información con muchas posibilidades.

En función del servicio de correo, se pueden utilizar distintos métodos para comprobar la existencia de una cuenta sin que el propietario sea notificado. Un primer acercamiento es a través de las opciones de recuperación de contraseña que ofrece la mayoría de plataformas. Al acceder a esta opción, es posible poner un nombre de usuario, y si existe, el sistema ofrecerá las opciones de recuperación de contraseña; en caso de no existir, la mayoría de plataformas informan mediante un mensaje de error. Como ejemplo, se muestra el comportamiento del acceso al servicio de correo Gmail con una cuenta existente, y una que no existe.

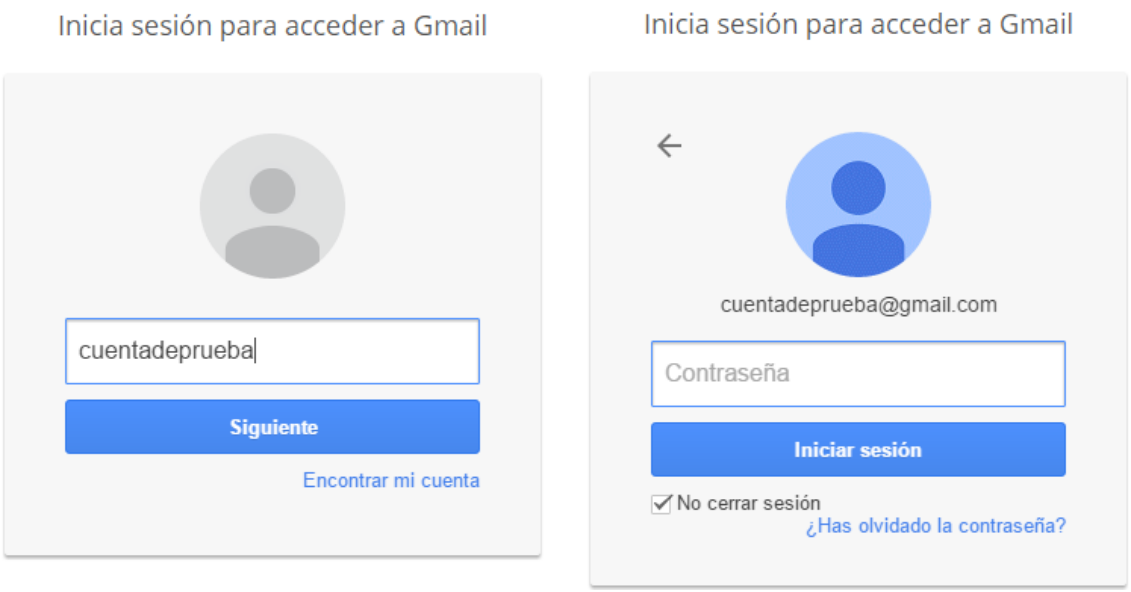


Figura 9. Captura del proceso de acceso al servicio Gmail con una cuenta existente

Figura 10. Captura del proceso de acceso a Gmail cuando la cuenta no existe

Además de mostrar si una cuenta existe, mediante la opción de recuperación de contraseña, es posible ver parte de otros medios de comunicación alternativos relacionados con la propia cuenta y que sirven para el propósito de recuperación de contraseña, a continuación se muestra la información que ofrece el servicio de recuperación de contraseñas de Microsoft (muestra las primeras letras y el dominio completo de la cuenta de correo alternativa y los dos últimos números del teléfono si es que está informado):

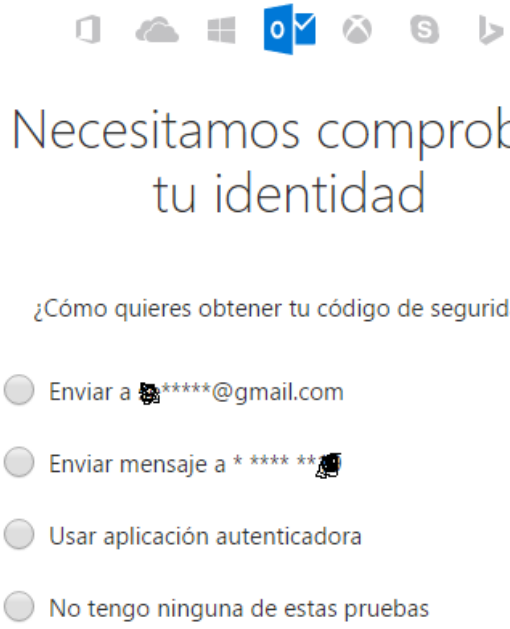


Figura 11. Captura del proceso de recuperación de contraseña de Microsoft

Al igual que en el caso de los alias, existen varios servicios o plataformas que ofrecen la búsqueda o validación de correos en diferentes plataformas. Por ejemplo, el servicio Verify Email [19] permite hacer consultas e correo a través de una interfaz web, aunque también ofrece una API de pago; este servicio se basa en los mensajes del protocolo SMTP (Simple Mail transfer Protocol), presumiblemente en la respuesta a la orden `RCPT TO` que indica si la cuenta existe en ese dominio o no.

Sin embargo, no es habitual que los usuarios maliciosos utilicen cuentas de correo de dominios con carácter privativo como los de empresas u organizaciones. Por el contrario, suelen utilizar correos de plataformas públicas como Gmail, Yahoo, Hotmail, etc. En este sentido, el paquete mencionado con anterioridad, OSRFramwork, ofrece una librería llamada Mailfy, que permite comprobar la existencia de direcciones de correo en varios dominios de correo públicos, y que puede ser utilizado en aplicaciones o scripts de Python con este fin.

5.4. Obtener información a partir de una cuenta de correo electrónico

A partir de una cuenta de correo, es posible buscar más información identificativa en directorios públicos en internet. De este modo, el servicio *whois* permite comprobar toda la información de registro de un dominio en internet. Entre esta información se encuentra el nombre de la persona o empresa que lo registra, teléfono de contacto, dirección y también dirección de correo electrónico.

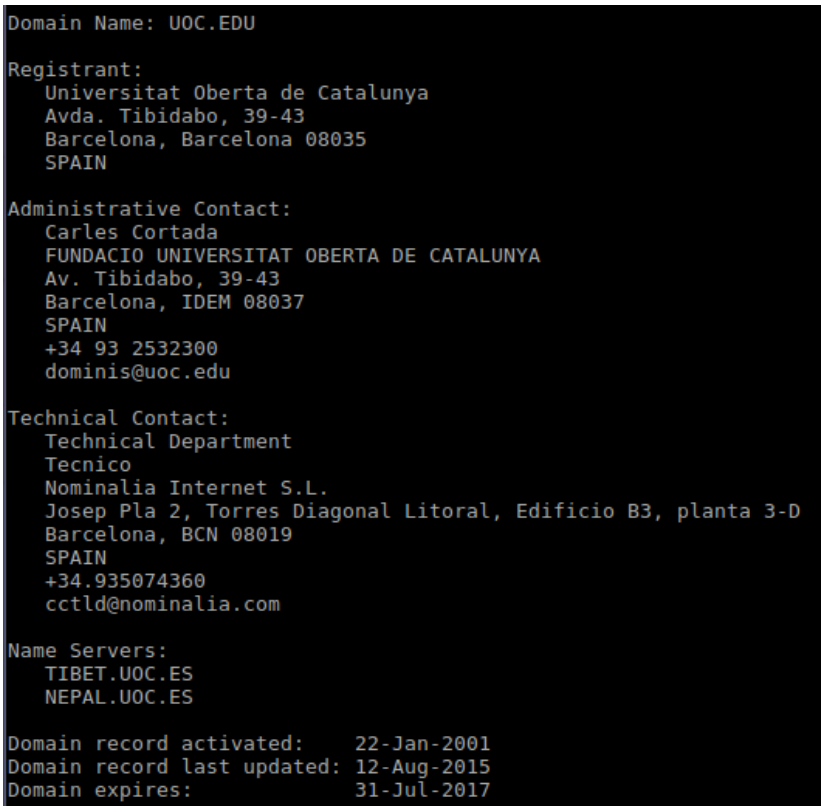


Figura 12. Respuesta de Whois (GNU/Linux) a la consulta sobre el dominio uoc.edu

De manera habitual, las consultas se realizan contra un nombre de domino, sin embargo es posible realizar búsquedas inversas a través del correo electrónico para ver que dominios están registrados a nombre de una dirección de correo concreta. Un ejemplo de servicio que ofrece esta posibilidad es View DNS [20] que mediante la opción “Reverse Whois lookup” permite realizar este tipo de búsquedas.

De esta forma, a partir de la dirección de correo –y en caso de encontrar algún dominio registrado mediante ella– se podría obtener información más precisa, que en un caso ideal podría incluir el nombre completo, un teléfono e incluso una dirección postal.

6. Desarrollo del prototipo

El prototipo del producto consiste en un script interactivo desarrollado en Python. Este desarrollo se divide en una primera etapa de recogida de requisitos. En segundo lugar, se lleva a cabo un análisis de las herramientas y librerías disponibles que se van a utilizar. Para el diseño se utiliza el estándar UML con un modelo de casos de uso y los diagramas de actividades correspondientes.

6.1. Requisitos

Para el desarrollo del proceso de desanonimización de usuarios en Twitter según las técnicas ya mencionadas, será necesario poder cumplir al menos con los siguientes requerimientos:

1. Acceso a los datos de Twitter, tanto a los tweets como a información referente a los usuarios, sobre todo los alias.
2. Localización de cuentas con el mismo alias en distintas plataformas públicas, como otras redes sociales, foros, etc.; para ofrecer fuentes de investigación adicionales.
3. Búsqueda de direcciones de correo electrónico de servidores públicos asociadas a los alias.

6.2. Análisis

A continuación, se presenta un breve análisis de las herramientas que se van a utilizar en el prototipo. La implementación del script se va a realizar en Python 2.7, ya que es un lenguaje de alto nivel con acceso a librerías muy completas que hacen de interfaz con distintas aplicaciones y servicios. Además, el lenguaje Python tiene una gran aceptación en el ámbito de la seguridad e informática forense.

Como ya se ha introducido en puntos anteriores, Twitter ofrece una completa API para el acceso y desarrollo de aplicaciones que consuman sus datos. Esta API es de tipo REST, por lo que a través de mensajes HTTP se puede interaccionar y realizar todas las operaciones. Hay que distinguir entre dos tipos principales de acceso a los datos de Twitter que implican también distintos propósitos:

- API REST: Permite hacer consultas sobre los distintos objetos de Twitter, permite obtener los tweets ya publicados por un usuario –su *timeline*– o bien realizar consultas sobre tweets según distintos parámetros, como por ejemplo por contenido o *hashtag.* El acceso a esta API está limitado por tiempo y número de consultas, permite realizar 15 consultas cada 15 minutos, además del límite en el número de objetos devueltos.

- Streaming API: Permite obtener un flujo de los tweets que se publican en tiempo real. Al contrario que la API REST, una vez realizada la conexión con el punto de consumo no hay límite en el número de tweets que se leen.

En ambos casos, es necesario conectarse a Twitter mediante el protocolo Oauth basado en *tokens* que permite la autorización de un tipo específico de cuenta destinado a las aplicaciones. Para crear una cuenta de tipo app solo es necesario disponer de una cuenta de Twitter válida y dar de alta una app en: https://apps.twitter.com.

Con el propósito de poder acceder a la API de twitter, existen desarrollos de librerías y módulos en distintos lenguajes que facilitan esta tarea, algunos de ellos llevan el mantenimiento por parte de Twitter y otros por terceros. Para el desarrollo del prototipo se va a utilizar el módulo de Python Tweepy [21], que permite manejar la autenticación, así como el acceso a la REST API y a la Stream API.

Por otra parte, para la consulta y búsqueda de perfiles de usuarios en otros medios es necesaria una herramienta que permita el acceso automatizado a esta información. Ya describieron algunas plataformas web que ofrecen esta información, sin embargo, el acceso a la API que ofrecen es de pago. Por lo tanto, para este propósito se va a utilizar el conjunto OSRFramework de código abierto y con licencia GPLv3 que ofrece las opciones necesarias.

El concreto, para la búsqueda de alias en varias plataformas se va a utilizar la aplicación usufy. Está implementada en Python, y ofrece la búsqueda en más 200 plataformas distintas. Su funcionamiento se basa en la respuesta web que se obtiene cuando se accede al perfil de un usuario a través de una URL directamente. Si el usuario no existe en la plataforma, es posible analizar el código HTML de la página mostrada en busca de un mensaje de error. La aplicación cuenta con una librería de *wrappers* o adaptadores a las plataformas.

Para la tarea de búsqueda de direcciones de correo electrónico en plataformas de correo públicas se va a utilizar la herramienta de OSRFramework mailfy. También implementada en Python, permite la búsqueda de correos en más de 20 dominios de correo públicos. Se basa en las respuestas del protocolo de correo SMTP, en concreto en la respuesta a RCPT TO que informa si la cuenta de correo existe en el servidor. No obstante, hay servidores de correo que aceptan los mensajes para cualquier cuenta aunque no exista, tal vez para evitar la obtención automática de direcciones válidas con fines maliciosos como el envío masivo de correos (spam).

En ambos caso, tanto usufy como mailfy están diseñados como aplicaciones independientes y no como módulos para importar en otros proyectos Python, aunque es posible realizar la parametrización y pasarla a la función main a través de una llamada.

6.3. Diseño

Para el diseño del script según los requisitos previos, se propone el diagrama de casos de uso y los diagramas de flujo generales correspondientes a cada uno de ellos.

6.3.1. Casos de uso

Caso 1: Monitorizar el *stream* de Twitter y filtrar por palabras clave

Caso 2: Monitorizar la actividad de un usuario

Caso 3: Obtener tweets del *timeline* de un usuario

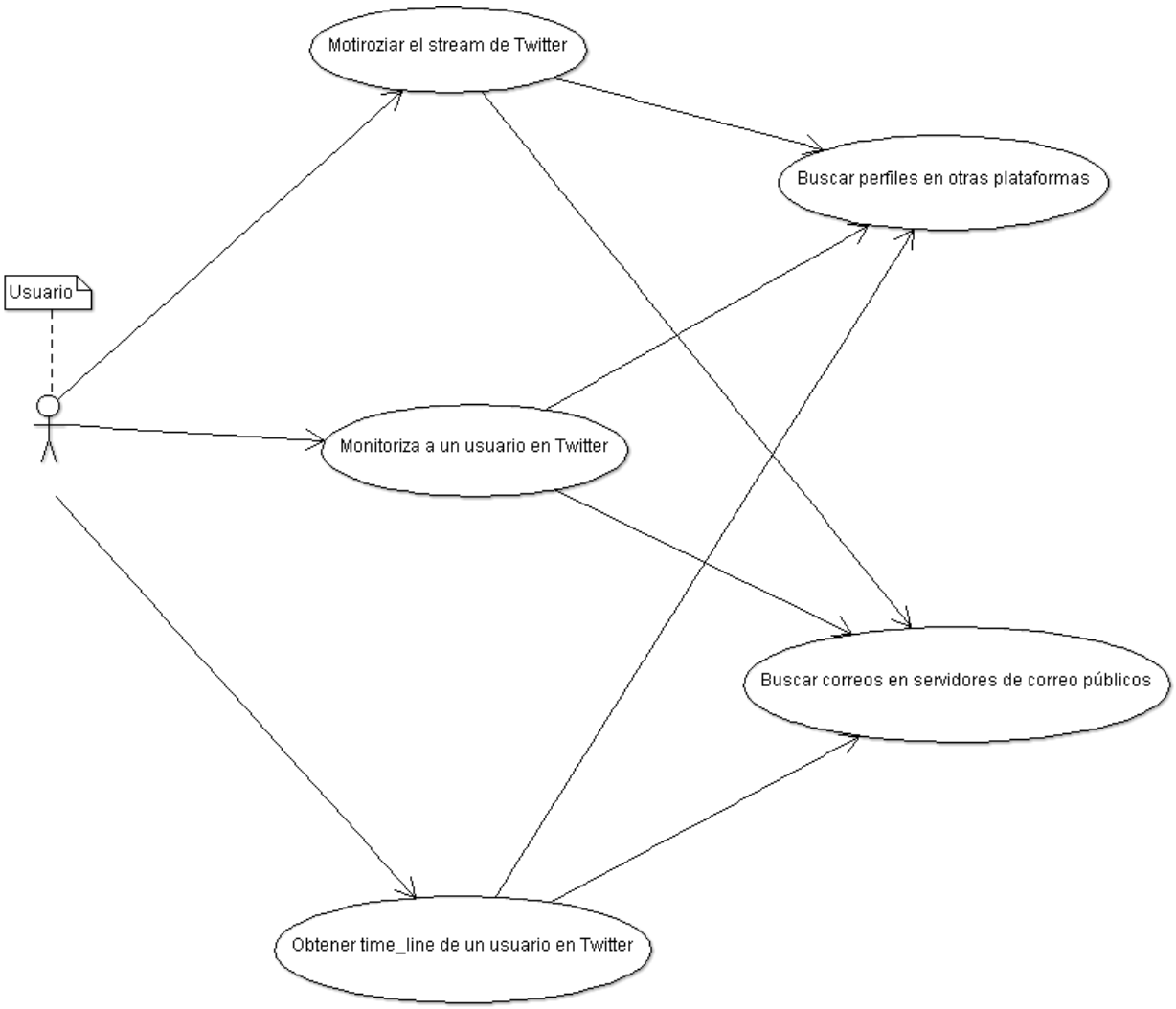


Figura 13. Diagrama de casos de uso

6.3.2. Diagramas de actividades

Caso 1: Monitorizar el *stream* de Twitter y filtrar por palabras clave

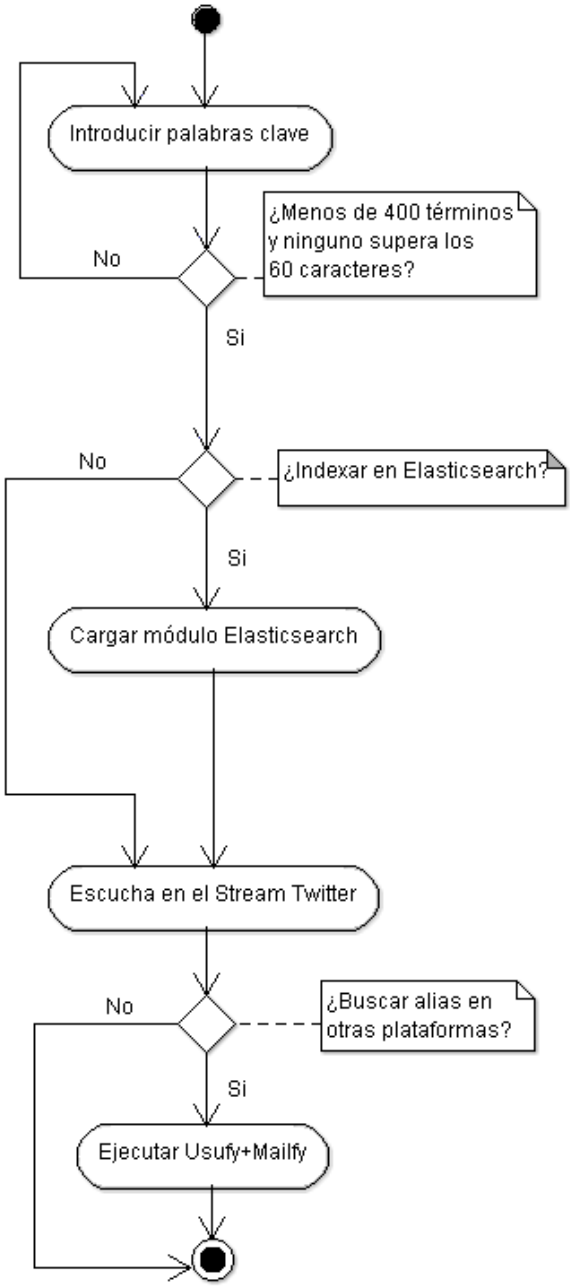


Figura 14. Diagrama de estados caso 1

Caso 2: Monitorizar la actividad de un usuario

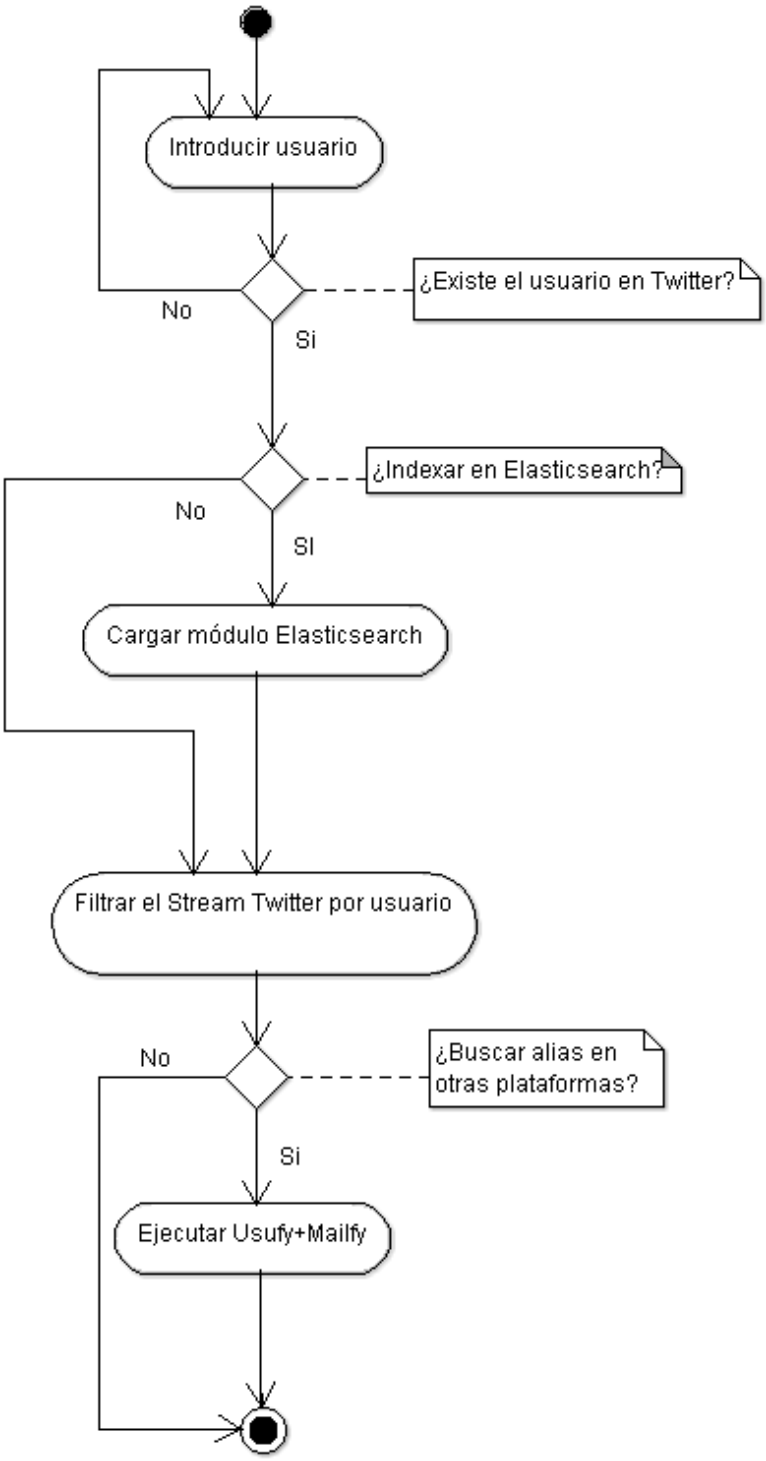


Figura 15. Diagrama de estados caso 2

Caso 3: Obtener tweets del *timeline* de un usuario

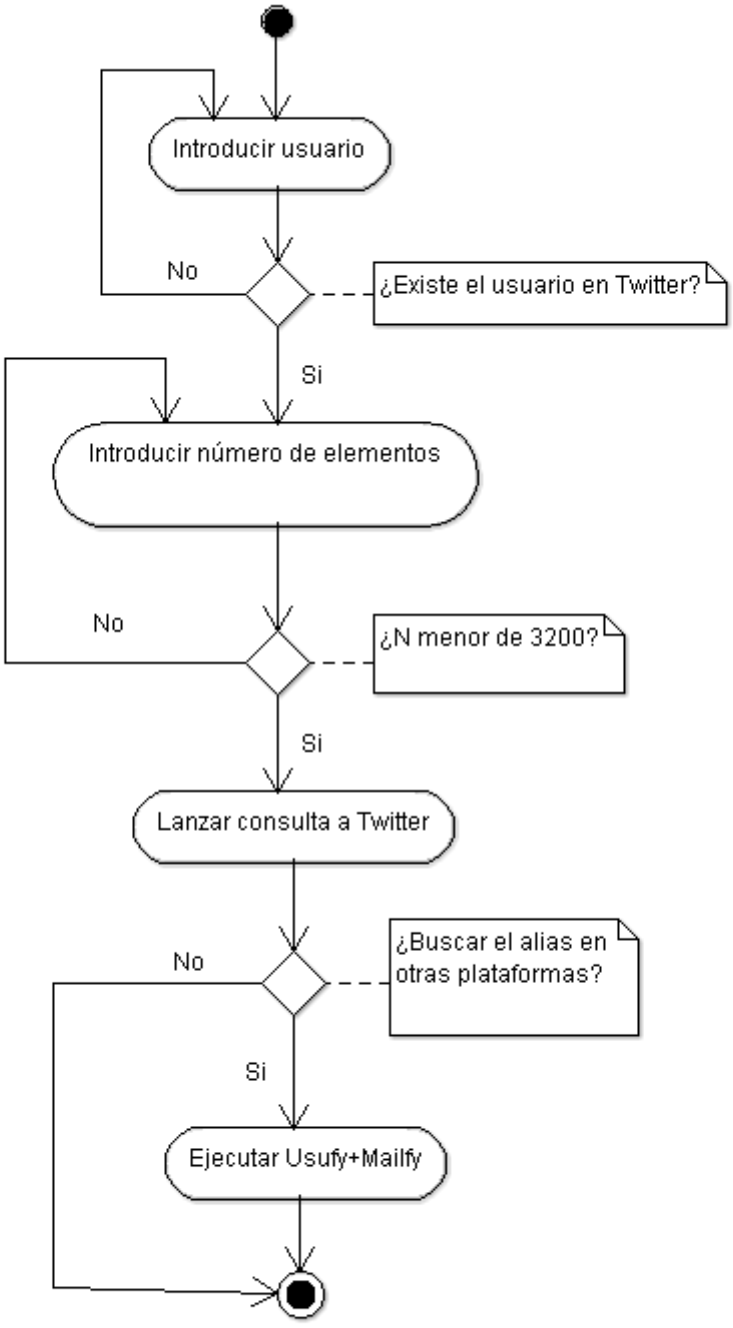


Figura 16. Diagrama de estados caso 3

6.4. Implementación

La implementación del prototipo se ha realizado para asegurar la compatibilidad con sistemas GNU/Linux, con el único prerrequisito de que dispongan del intérprete de Python (versión 2.7).

Se ha seguido una metodología ágil, con constante revisiones y pruebas para comprobar que se cumple con los requisitos planteados. Debido a que es un script interactivo, el estilo de programación seguido ha sido funcional, sin embargo se podría adaptar a la orientación a objetos de forma sencilla.

El código está debidamente comentado para que tanto su comprensión como su seguimiento sea sencillo; además, el propio programa ofrece una ayuda que permite conocer su funcionamiento y las opciones disponibles. De cualquier forma, en los anexos se ha incluido una breve guía con la explicación de la instalación y funcionamiento. La misma información puede encontrarse en el archivo “README.txt” incluido en el paquete con el código fuente.

La aplicación se llama “muot” (Monitoring users on Twitter). El código fuente se entrega adjunto a la memoria. Además, puede encontrarse en el siguiente repositorio:

https://github.com/Sergioln/muot

6.5. Interfaz de visualización

La interfaz principal del programa es un menú textual interactivo, sin embargo, para la visualización opcional de los datos en tiempo real se van a utilizar las siguientes aplicaciones:

- **Elasticsearch**[22]: Es un indexador de información, que permite además de introducir datos, realizar consultas. Contiene algunos tipos de datos muy interesantes, como el tipo `geo_point`, que define coordenadas que pueden ser exportadas a algún servicio de mapas. La interfaz de comunicación es una API de tipo REST; es posible utilizar un nodo local o bien un nodo o conjunto de nodos distribuidos. En este caso la opción será la de nodo local.

- **Kibana**[22]: Es un *frontend* para el consumo de datos desde Elasticsearch. ofrece una interfaz web muy visual, desde la que se pueden generar gráficos a partir de los datos. Además ofrece el uso de *dashboards* o paneles de control en los que se pueden consultar distintos gráficos de un solo vistazo.

El motivo para elegir estas aplicaciones es, en primer lugar, que son herramientas de código abierto que continúan con la filosofía global de este trabajo. Por otra parte, ofrecen un módulo para Python con el que es muy sencillo enviar la información consumida en el *stream* de Twitter para poder prepararla y mostrarla.

Las versiones específicas utilizadas son:

- Elasticseach 2.4.3
- Kibana 4.6.3 (linux-x86)

7. Prueba de concepto

Para evitar posibles conflictos éticos y legales por el tratamiento de datos personales reales obtenidos de Twitter, en la PoC (prueba de concepto) se van a utilizar unas cuentas de Twitter creadas a propósito para este fin, ya que permiten hacer una demostración en circunstancias similares a la realidad.

El entorno utilizado para la PoC es una máquina virtual VirtualBox con sistema operativo Ubuntu 16.04 de 32 bits. Se parte de la configuración necesaria para ejecutar la aplicación, con todas las dependencias y complementos opcionales.

La estructura de la PoC es la siguiente:

Caso de uso 1: Monitorizar el *stream* de Twitter mediante la aplicación de un filtro por palabra clave

- Se monitoriza el *stream* de Twitter por una palabra clave elegida: test_muot07
- Se publica en Twitter mediante las cuentas de prueba: test_tfg_sec2 y test_tfg_sec3
- Se muestran los resultados

Caso de uso 2: Monitorizar la actividad de un usuario en Twitter

- Se monitoriza la actividad del usuario: test_tfg_sec2
- Se muestran los resultados
- Se realiza la búsqueda manual de correos en las plataformas en las que no ha sido posible hacerlo mediante la aplicación

Caso de uso 3: Obtener *tweets* del *timeline* de un usuario

- Obtener los *tweets*
- Mostrar los resultados

Toda la prueba de concepto queda reflejada en el vídeo captura de pantalla adjunto a la memoria. También puede descargarse mediante el siguiente enlace:

http://edumedia.uoc.edu:8081/download/586a490afa9b48d825cb84fb

8. Conclusiones

El proceso de desanonimización de identidades es una tarea compleja y que requiere de diversos factores. Como ya se ha visto, el primer paso, y tal vez el más importante, pasa por conocer la dirección IP. Sin embargo, otros elementos menos intuitivos también intervienen en la identidad que se transmite cuando alguien se conecta a internet. No obstante, muchos de los métodos de identificación requieren disponer de acceso a los servidores de las plataformas, y que además se recojan y almacenen los datos necesarios (como los datos de navegación web); de modo que es necesario explorar técnicas alternativas.

Por una parte, Twitter ofrece información en tiempo real que permite obtener un tweet de un usuario con todos los objetos asociados a este, y que pueden servir para la identificación o al menos para crear un perfil que lo distinga en cierto grado, como la ubicación, el cliente que utiliza –que puede informar del modelo de teléfono–, etc. Además, es posible acceder a un historial que permite obtener información de los *tweets* publicados en el pasado que podrían contener información identificativa.

Por otro lado, es habitual que los usuarios utilicen el mismo alias en distintas plataformas, ya sean redes sociales o bien foros públicos de debate. Así, a partir de un nombre de usuario, es posible localizar perfiles en multitud de plataformas en las que se podrían encontrar también datos identificativos (incluso fotografías). Y si atendemos a direcciones de correo electrónico, la situación es la misma, ya que mediante métodos que no notifican al usuario es posible encontrar direcciones en muchas plataformas de correo públicas con distintos métodos sencillos de implementar.

Mediante el desarrollo del modelo, se ha podido demostrar que es sencillo realizar un seguimiento en Twitter de un usuario, o de un tema mediante palabras clave, ya que hay muchas posibilidades de acceso a la API de Twitter; y que además existen desarrollos de software que permiten poner en práctica los métodos de identificación explorados (tanto privativos como de código abierto).

A la vista de los resultados, y aunque no era el objetivo del trabajo, se ha podido comprobar que el anonimato en la red es un trabajo que requiere controlar distintos protocolos, aplicaciones y servicios: ocultación de la IP, enmascaramiento de los datos de navegación, uso de alias exclusivos para cada plataforma y el uso de correos electrónicos distintos para cada caso (mediante las direcciones desechables).

Para finalizar, cabe destacar que las técnicas descritas a lo largo de este trabajo se pueden extender a cualquier plataforma pública de internet. Además, es posible complementarlas con la investigación de más fuentes abiertas en internet, ya que cuantos más datos se aporten más completo será el perfil que se pueda construir a partir de los datos recogidos.

8.1. Líneas futuras y propuestas de mejora

El prototipo propuesto cubre parte de las técnicas de desanonimización presentadas durante la fase de investigación, sin embargo, no se ha implementado la comprobación automática de correos en, al menos, los servidores de Microsoft (@hotmail.com,

@outlook.com, @msn.com) y Yahoo (@yahoo.com), de modo que queda pendiente como mejora de la funcionalidad de la aplicación.

Como continuación de la mejora de funcionalidades, se podría estudiar la implementación de la búsqueda de dominios registrados por correo electrónico a través de fuentes abiertas o mediante algún desarrollo de código abierto existente.

Por otra parte, se puede crear una interfaz gráfica que sustituya al menú textual interactivo para que pueda ser utilizada de una forma más sencilla e intuitiva por aquellos usuarios que no se sientan cómodos con la línea de comandos. También, junto a la interfaz gráfica, se podría modificar la aplicación para admitiera el paso de parámetros por línea de comandos de modo que pudiera ser integrada en *scripts* de automatización.

9. Fuentes de Información

Wikipedia - https://en.wikipedia.org

Twitter Developer Documentation – API https://dev.twitter.com/overview/api

[1] Interactive Advertising Bureau - Estudio anual de redes sociales 2016 http://www.iabspain.net/wp-content/uploads/downloads/2016/04/IAB_EstudioRedesSociales_2016_VCorta.pdf

[2] Twitter Inc – Investor Relations https://investor.twitterinc.com/index.cfm

[3] Internet Lives Stats - Twitter usage (Consultado el 07/11/2016) http://www.internetlivestats.com/twitter-statistics/

[4] Edward Snowden - Wikipedia https://en.wikipedia.org/wiki/Edward_Snowden

[5] Europol - Internet Organised Crime Threat Assessment (IOCTA) 2016 https://www.europol.europa.eu

[6] TOR Project https://torproject.org

[7] Invisible Internet Project – I2P *www.i2p2.de*

[8] Am I unique? https://amiunique.org/

[9] Panopticlick - https://panopticlick.eff.org/

[10] Narayanan and Shmatikov - De-anonymizing Social Networks https://www.cs.cornell.edu/~shmat/shmat_oak09.pdf

[11] Nilizadeh, Kapadia, Ahn - Community-Enhanced De-anonymization of Online Social Networks http://www.yongyeol.com/papers/nilizadeh-deanon-2014.pdf

[12] Wondracek, Holz, Kirda, Kruegel - A Practical Attack to De-Anonymize Social Network Users http://hgi.ruhr-uni-bochum.de/media/emma/veroeffentlichungen/2011/06/07/deanonymizeSN-Oakland10.pdf

[13] B. Schneier, Mudge - Cryptanalysis of Microsoft's PPTP Authentication Extensions (MS-CHAPv2) https://www.schneier.com/academic/archives/1999/09/cryptanalysis_of_mic_1.html

[14] Tails - https://tails.boum.org/

[15] Graham Cluley - Fugitive John McAfee's location revealed by photo meta-data screw-up https://nakedsecurity.sophos.com/2012/12/03/john-mcafee-location-exif/

[16] CheckUserNames http://checkusernames.com/

[17] NameCh_k - https://namechk.com/

[18] OSRFramework - https://github.com/i3visio/osrframework

[19] Verify Email - http://verify-email.org/

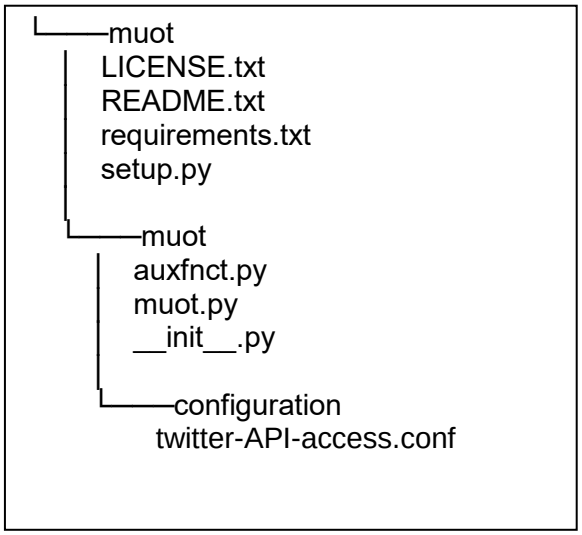
[20] View DNS - http://viewdns.info/

[21] Tweepy - http://tweepy.readthedocs.io/en/v3.5.0/

[22] Elasticsearch - https://www.elastic.co/

10. Anexos

Contenido del paquete de software



Instalación

Para la instalación de la aplicación es suficiente con descargar el paquete completo e instalar las dependencias necesarias para la ejecución, en particular se trata de los paquetes Python:

- osrframework
- tweepy
- elasticsearch

La forma más sencilla de instalar las dependencias es a través del gestor de paquetes PyPI y el archivo requirements.txt:

```
pip install -r requirements.txt
```

Para instar las aplicaciones opcionales Elasticserach y Kibana, se pueden descargar de la fuente original:

```
wget
https://download.elastic.co/elasticsearch/release/org/elasticsearch/
distribution/tar/elasticsearch/2.4.3/elasticsearch-2.4.3.tar.gz
```

```
wget https://download.elastic.co/kibana/kibana/kibana-4.6.3-linux-x86.tar.gz
```

Para ejecutar la aplicación hay que ejecutar el archive muot.py a través de python o directamente si se le ha dado permiso de ejecución:

```
python muot.py
```

```
./muot.py
```

Nota importante: Antes de la ejecución de la aplicación, es necesario rellenar el archivo de configuración twitter-API-access.conf con los datos de acceso a la API de de Twitter que se pueden obtener en https://apps.twitter.com.

Descripción de las funciones

Clases y funciones principales:

Class tweepy.Streamer:

La conexión con el *stream* de Twitter se realiza a través de la clase tweepy.Streamer, de modo que se ha extendido la clase para que realice las funciones que se requieren, en concreto los siguientes métodos:

- **__init__** : Extensión del constructor para que acepte nuevos parámetros que serán pasados por referencia.
- **on_connect**: Es llamado al realizar la conexión con el *stream* de Twitter, para el propósito de la aplicación solo se encarga de mostrar información por la salida estándar.
- **on_status**: Este método es llamado cada vez que se publica un tweet nuevo según el filtro aplicado al *stream* (por contenido o por usuario en la aplicación). En concreto, se encarga de grabar una fila nueva en el archivo en el que se guardan los tweets, y en caso de habilitar el módulo que se comunica con Elasticsearch (ver punto 6.4) pasa los datos del tweet a este para que sean indexados.
- **on_error**: Es invocado si se produce algún error en el *stream*. Para la aplicación se controlan los errores producidos por sobrepasar el límite de intentos de conexión.

Función monitorStrem()

Es el método llamado en el caso de uso 1 –monitorizar el *stream* de Twitter por palabras clave–. Recibe las palabras clave por la entrada estándar, comprueba que no se superen los límites impuestos por Twitter (60 caracteres por término y un máximo de 400 términos).

Crea el objeto que gestiona la autentificación en Twitter, el objeto para conectar con la Streamer API y lanza la escucha activa mediante el filtrado según los términos introducidos.

El control para finalizar la escucha se realiza mediante la gestión de la interrupción por teclado. Una vez finalizada la escucha activa, controla la llamada para iniciar la búsqueda de perfiles y correos a través de las aplicaciones usufy y mailfy.

Función UserTrack()

Método llamado al ejecutar el caso de uso 2 –Monitorizar la actividad de un usuario en Twitter–. Recibe como parámetro el nombre de usuario, comprueba si el usuario existe (a través de la API REST) antes de continuar.

Al igual que la función anterior, crea el objeto para gestionar la autentificación en Twitter, y también el objeto para la conexión al *stream*. La escucha activa se activa mediante el filtro por usuario.

El control para la finalización de la escucha se lleva a cabo a través de la captura de la interrupción por teclado, y permite elegir si se buscan perfiles y correos relacionados con el alias introducido.

Función readUserTweets()

Función implementada para caso de uso 3, obtener n tweets de un usuario. Recibe el nombre de usuario por la entrada estándar y comprueba que exista en Twitter. También recibe como parámetro el número de tweets a recuperar (de un máximo de los últimos 3200 impuesto por Twitter).

La recuperación de los tweets se realiza a través de la función user_timeline, que permite obtener 200 tweets como máximo por cada consulta. Para indicar hasta que tweet se quiere recuperar se le pasa el ID máximo que se quiere es esa consulta.

La función showHelp()

Muestra un texto de ayuda sobre el funcionamiento de la aplicación, y un resumen de lo que realiza cada opción.

Funciones auxiliares:

optionsMenu()

Se encarga de mostrar por la salida estándar el banner de la aplicación y las opciones del menú principal

lookForUsu()

Realiza la parametrización y llamada a la aplicación usufy que realiza búsquedas en distintas plataformas del alias.

lookForMail()

Igual que la función anterior para la aplicación mailfy.

createTwitterCSV()

Recibe como parámetros el nombre de archivo y la ruta; se encarga de crear el archivo CSV que se utilizará para escribir la información de los tweets.

timeStr()

Devuelve una cadena con la marca de tiempo personalizada para acompañar a los logs y demás información mostrada por la salida estándar.

useElasticsearch()

Controla la activación del módulo de comunicación con el indexador Elasticsearch (ver punto 6.4)

createIndexES()

Crea el índice pasado como parámetro según el *template* predefinido

getUserTweets()

Devuelve el número de tweets del usuario que se ha pasado como parámetro

twitterOauthHandler()

Toma los datos del archivo de configuración en el que se han especificados los datos de acceso a la API de Twitter y crea el gestor OAuth necesario.