UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de Telecomunicación

Estudio de los artefactos forenses generados por la aplicación WhatsApp Web en el entorno Chromium

Trabajo Fin de Grado

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

AUTOR/A: Moreno Martínez, Álvaro

Tutor/a: López Patiño, José Enrique

CURSO ACADÉMICO: 2022/2023

Resumen

El estudio sobre la adquisición de evidencias y su significado en las aplicaciones de mensajería instantánea se ha enfocado, principalmente, en el ámbito de los dispositivos móviles. Sin embargo, el uso de estas aplicaciones se está extendiendo a los ordenadores a través del navegador o de una aplicación de escritorio. La documentación de los artefactos forenses que las aplicaciones de mensajería instantánea generan en estos equipos carece de un análisis profundo. Así pues, este trabajo pretende realizar un estudio de la información que se puede obtener, concretamente, de WhatsApp Web cuando se ejecuta en un navegador basado en Chromium. Se ha realizado una recopilación y actualización de toda la documentación y estudios relevantes en este ámbito. Se han estudiado los artefactos generados y se ha identificado y probado que los mensajes enviados y recibidos se almacenan en las bases de datos locales del equipo. Junto a estos, se ha llegado a obtener imágenes, vídeos y audios encriptados, los cuales se han descifrado y relacionado con sus respectivas conversaciones, así como con marcas de tiempo de envío y recepción. Por último, se ha profundizado en la extracción de estos artefactos de forma automática, creando una serie de scripts en Python, tanto para las versiones antiguas de WhatsApp Web como para las nuevas.

Resum

L'estudi sobre l'adquisició d'evidències i el seu significat en les aplicacions de missatgeria instantània s'ha centrat, principalment, en l'àmbit dels dispositius mòbils. No obstant això, l'ús d'aquestes aplicacions s'està estenent als ordinadors a través del navegador o d'una aplicació d'escriptori. La documentació dels artefactes forenses que les aplicacions de missatgeria instantània generen en aquests equips manca d'un anàlisi profund. Així doncs, aquest treball pretén realitzar un estudi de la informació que es pot obtenir, concretament, de WhatsApp Web quan s'executa en un navegador basat en Chromium. S'ha realitzat una recopilació i actualització de tota la documentació i estudis rellevants en aquest àmbit. S'han estudiat els artefactes generats i s'ha identificat i provat que els missatges enviats i rebuts s'emmagatzemen en les bases de dades locals de l'equip. Juntament amb aquests, s'ha aconseguit obtenir imatges, vídeos i àudios encriptats, els quals s'han desxifrat i relacionat amb les seues respectives converses, així com amb marques de temps d'enviament i recepció. Finalment, s'ha aprofundit en l'extracció d'aquestos artefactes de manera automàtica, creant una sèrie de scripts en Python, tant per a les versions antigues de WhatsApp Web com per a les noves.

Abstract

The study on the acquisition of evidence and its significance in instant messaging applications has mainly focused on the realm of mobile devices. However, the use of these applications is expanding to computers through web browsers or desktop applications. The documentation of forensic artifacts generated by instant messaging applications on these devices lacks in-depth analysis. Therefore, this work aims to conduct a study of the information that can be obtained specifically from WhatsApp Web when running on a Chromium-based browser. A compilation and update of all relevant documentation and studies in this field have been carried out. The generated artifacts have been examined, and it has been identified and tested that sent and received messages are stored in

the local databases of the device. Additionally, encrypted images, videos, and audios have been obtained, which have been decrypted and linked to their respective conversations, along with timestamps of sending and receiving. Finally, the extraction of these artifacts has been further explored through automated means, creating a series of Python scripts for both old and new versions of WhatsApp Web.

A mis padres, por su apoyo incondicional, su motivación y esfuerzo.

Índice general

Índice de figuras

Índice de tablas

Parte I

Memoria

Capítulo 1

Introducción

1.1. Antecedentes

El número de aplicaciones de mensajería instantánea (IM) ha aumentado considerablemente en los últimos años. Este tipo de aplicaciones han supuesto un gran avance en cuanto a la forma en que nos comunicamos. Entre ellas, la más conocida y utilizada es WhatsApp [1]. A través de ella, los usuarios son capaces de enviar y recibir mensajes, audios, fotos, vídeos y documentos. Bien, a través de grupos, o por chats privados. Además, incorpora un cifrado extremo a extremo, con la finalidad de aportar seguridad y privacidad [2].

El desarrollo de WhatsApp permite que, no sólo se pueda ejecutar en un teléfono móvil sino también en un ordenador, bien a través del cliente Web, o bien a través de la aplicación de Escritorio.

El número de investigaciones que se apoyan en WhatsApp está en aumento debido a su amplio y diverso uso. Entre las pruebas, pueden aportarse mensajes, imágenes, vídeos, documentos o marcas de tiempo. Las evidencias digitales que proveen las aplicaciones de mensajería instantánea tienen, de esta manera, un gran interés en el ámbito forense.

Por tanto, el **objetivo principal** de este estudio es localizar y aportar información del contenido de los artefactos que genera WhatsApp (WhatsApp Web, concretamente) cuando se ejecuta, o se ha ejecutado, en un navegador.

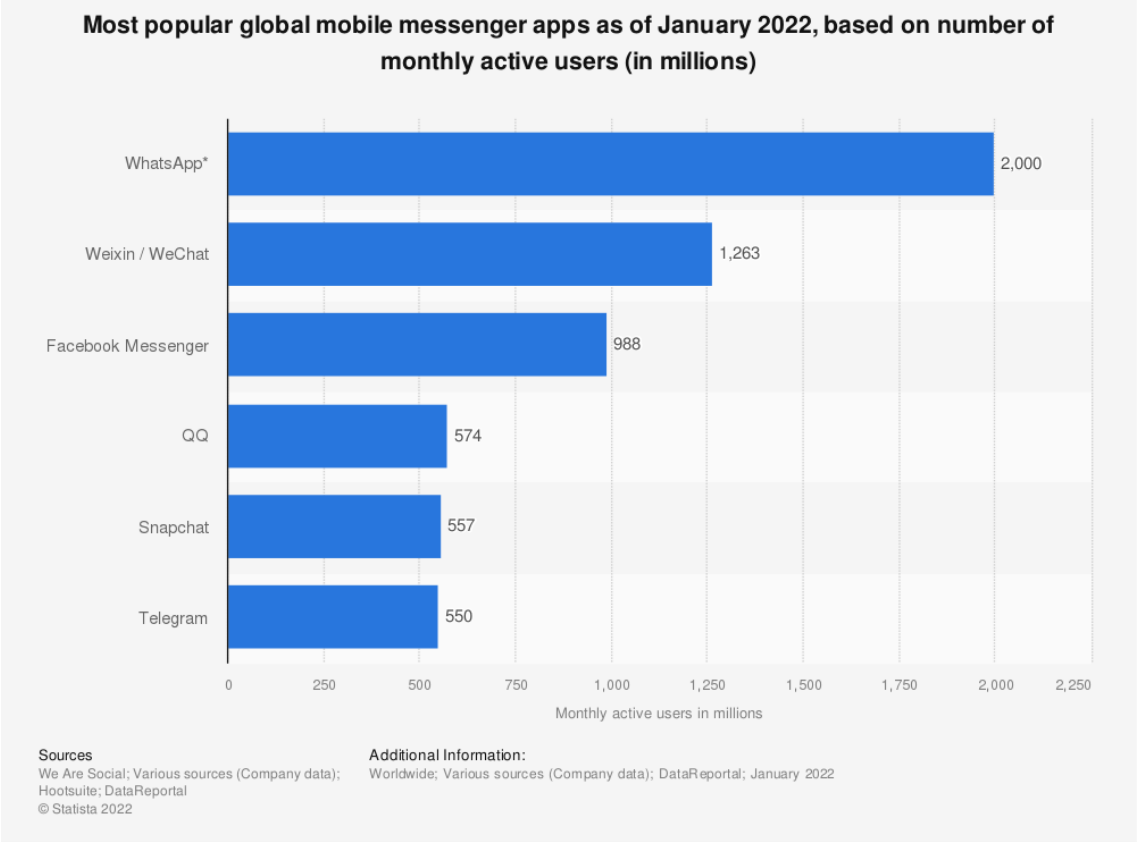


Figura 1.1: Aplicaciones de mensajería móvil más populares del mundo en enero de 2022, según el número de usuarios activos mensuales (en millones)[1].

1.2. Alcance

En este estudio se ha abordado únicamente la aplicación de mensajería instantánea de WhatsApp. Se han utilizado dos dispositivos móviles Android, con diferentes versiones tanto del sistema operativo como de la aplicación.

El sistema operativo sobre el que se realizará el estudio es Windows 10. Como navegador para ejecutar el cliente Web de WhatsApp, se ha elegido Google Chrome. En ambos casos la elección se ha realizado basándose en el número de usuarios que los utilizan.

En el primer caso, Windows es el sistema operativo con más cuota de mercado con un 69.61 % en diciembre de 2021 [3]. Google Chrome, a su vez, ocupa el primer puesto como navegador más utilizado con una cuota de mercado de 63.82 % [4].

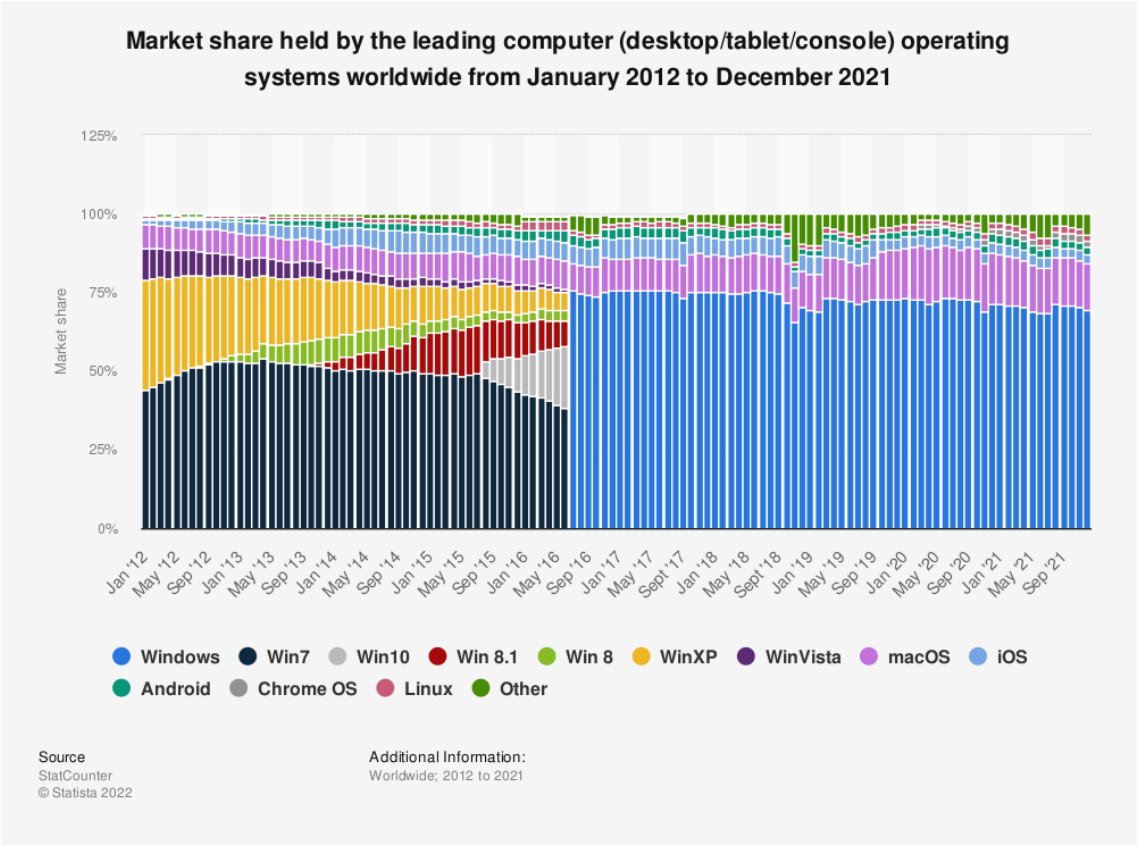


Figura 1.2: Cuota de mercado de los principales sistemas operativos para ordenadores (de sobremesa/mesa/consola) en todo el mundo desde enero de 2012 hasta diciembre de 2021 [3].

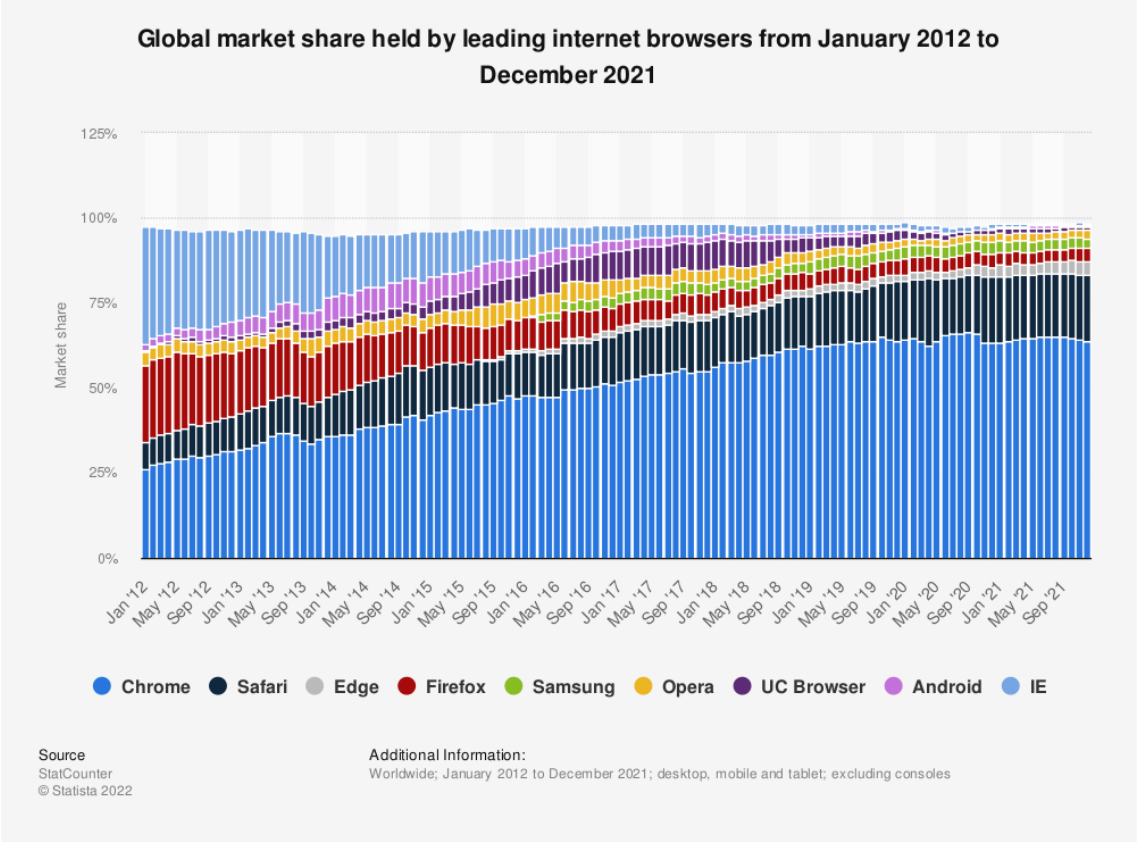


Figura 1.3: Cuota de mercado global de los principales navegadores de Internet desde enero de 2012 hasta diciembre de 2021 [4].

Además, otro aspecto importante para la elección de Google Chrome es que, al estar basado en Chromium, la metodología y herramientas empleadas en este estudio se extienden a todos los navegadores que se basan en esta tecnología [5].

Algunos navegadores basados en Chromium son [6][7]:

- Brave
- Microsoft Edge
- Opera
- Vivaldi
- Yandex
- Blisk
- Epic Privacy Browser
- SlimBrowser

1.3. Preguntas de la investigación

Con el objetivo del estudio presente, se han planteado unas preguntas que serán resueltas a lo largo del mismo:

- ¿Dónde se localizan los artefactos que genera WhatsApp Web?
- ¿Qué tipo de artefactos son?
- ¿Se pueden obtener mensajes?
- En caso de que se guarden mensajes:
  - ¿Cada mensaje queda identificado con el modelo del dispositivo?
  - ¿Se puede identificar cuándo un mensaje se envía desde el teléfono móvil, desde el cliente Web o a través de la aplicación de Escritorio?
  - ¿Se puede ver el contenido en texto plano de los mensajes?
- ¿Se pueden obtener archivos multimedia?
- ¿Se pueden obtener los contactos?

Capítulo 2

Estado del arte

La **informática forense** puede ayudar a resolver crímenes en los que se vea involucrado prácticamente cualquier dispositivo electrónico. Especialmente ordenadores y dispositivos móviles. Los **artefactos** que se pueden encontrar en estos dispositivos pueden ser, por ejemplo, carpetas, ficheros, historiales, logs o cualquier dato que contenga información fiable que ayude a apoyar o refutar una hipótesis.

En el caso que comprende este estudio, los artefactos de interés son los que generan los navegadores. A la rama de la informática forense que comprende a los navegadores se le conoce como ***Browser forensics***.

Los navegadores permiten a los usuarios leer noticias, ver vídeos, películas o series, conectarse a redes sociales, realizar cursos online, etc. Todo esto genera información de diferentes tipos (cache, logs, cookies, historial, bases de datos, etc) y en diferentes lugares. Esto dependerá del navegador.

Una posible forma de identificar el navegador que se está utilizando es buscando en los archivos de precarga de Windows (*Windows Prefetch files*):

C:\Windows\Prefetch [8]

Este estudio se centra en el **navegador Chrome en Windows**. Por tanto, es necesario conocer dónde se ubican sus artefactos. Se ha visto, y la literatura lo corrobora [8], que se pueden encontrar en:

C:\user\{USERNAME}\AppData\Local\Google\Chrome\User Data\Default

Se conocen herramientas, tanto open-source como de pago, que analizan los directorios y ficheros donde los navegadores almacenan información. Algunos ejemplos son: Nirsoft Tools, Autopsy o Browser forensic tool. No obstante, estas herramientas no serán utilizadas en el estudio.

Por otro lado y, al contrario que en el caso de estudio, el análisis forense en las aplicaciones IM en los **dispositivos móviles** es bastante popular. Hay abundante literatura al respecto, sobre todo de

WhatsApp. Los trabajos de este ámbito que pueden resultar relevantes para este estudio se comentarán a continuación.

En primer lugar, en ‘Análisis del modelo de datos de la red social WhatsApp y sus aplicaciones al peritaje‘ [9] y en ‘WhatsApp in Plain Sight: Where and How You Can Collect Forensic Artifacts‘ [10] se hace un estudio de las bases de datos de WhatsApp, sus tablas y sus campos, entre otra información. De estos, se extrae que las bases de datos más importantes, y de las cuales se podrá extraer información para el estudio, son las que reciben el nombre ‘**wa.db**‘ y ‘**msgstore.db**‘.

En ‘wa.db‘ hay una tabla a partir de la cual se pueden obtener los contactos, **wa_contacts**, así como el ID de cada uno (campo **jid**), su nombre o su estado.

En ‘msgstore.db‘ la tabla más importante es ‘**messages**‘, la cual alberga información como el número del contacto, el mensaje, el estado, marcas de tiempo e información sobre ficheros adjuntos. El campo **key_id** refleja el identificador único del mensaje, el cual será relevante más adelante.

En segundo lugar, a través del curso de ‘Forense Móviles‘ [11], se mostró una manera de **extraer las bases de datos de WhatsApp** sin necesidad de ser root en el dispositivo. Se realiza a través de un método conocido con el nombre de ***APK downgrade***. Los comandos realizados se pueden encontrar en el anexo ‘Extracción de las Bases de Datos de WhatsApp‘.

Esta información será necesaria para buscar números de teléfono, identificadores únicos de mensajes o identificadores de usuario en los artefactos que se obtengan.

En cuanto al **estudio de WhatsApp Web**, la literatura es muy limitada y se basa, principalmente, en cuatro estudios.

En ‘Forensic investigation on WhatsApp web using framework integrated digital forensic investigation framework version 2’ [12], se realiza un estudio en el cual se analiza WhatsApp Web. Para ello, también se lleva a cabo la extracción de las bases de datos de WhatsApp en el dispositivo móvil. Se localiza dónde almacena **Google Chrome** los **artefactos** de interés. Además, hace uso de la herramienta Wireshark, con la cual intercepta unos ficheros con la extensión ‘**.ENC**‘ (ficheros encriptados). Termina el estudio tratando de desencriptar estos ficheros a través de la clave maestra de la base de datos obtenida del teléfono, sin éxito.

En ‘WhatsApp Forensics: Locating artifacts in Web and Desktop clients‘ [8], se estudia qué artefactos se pueden recuperar desde el punto de vista forense cuando se utiliza WhatsApp en la **versión Web** y en la de **Escritorio**. Utiliza las herramientas **FTK, AXIOM y Autopsy** en seis entornos distintos. Tres Windows y tres en Mac OS. En estos se analizó el cliente Web en Google Chrome, Mozilla Firefox y Safari. Tanto en el entorno Web como en el de Escritorio, afirman que pudieron recuperar prácticamente todo, al menos de forma parcial (alguna marca de tiempo o un log). También indican la localización de los artefactos. En una de las tablas, indican los artefactos que se han podido extraer de un **fichero log**.

En ‘Browser Forensic Investigations of WhatsApp Web Utilizing IndexedDB Persistent Storage‘ [13], se realiza un estudio sobre los artefactos que se pueden encontrar en las bases de datos **IndexedDB y LevelDB**. Tras esto, visualiza la base de datos generada por Google Chrome al utilizar el cliente Web de WhatsApp y realiza un listado detallado sobre los artefactos que se pueden encontrar. Termina realizando un software de nombre **BrowSwEx** a través del cual se obtienen algunos artefactos de interés de WhatsApp Web en Chrome. Entre estos se puede encontrar la hora de inicio de sesión o si el usuario está cerca o lejos del equipo. Si ha recibido o enviado algún mensaje o cierta información sobre si se carga o reproducen ficheros multimedia.

En ‘A Deep-dive Analysis on WhatsApp Artifacts and their Relevance in Crime Investigation‘ [14], se centra en la adquisición forense de los artefactos de WhatsApp de un dispositivo móvil, aunque, hacia el final del reporte, indica la localización de los artefactos generados por WhatsApp Web en Google Chrome y en Mozilla Firefox, junto con los generados por la aplicación de Escritorio. A su vez, informa, de forma resumida, los mensajes que aparecen el fichero de log. También menciona que, aunque, en este se muestra información sobre mensajes o ficheros multimedia compartidos, **su contenido no está disponible**.

En adición a los estudios comentados, se han analizado dos trabajos con referencias a WhatsApp Web.

Primeramente, el trabajo realizado por sigalor en ‘WhatsApp Web reverse engineered‘ [15]. En este, se provee lo necesario para **replicar WhatsApp Web**. El **backend** (fichero ‘backend‘ del repositorio de GitHub) es lo que se utilizará en este estudio. En él se encuentra lo necesario para el **desencriptado de ficheros multimedia**.

El otro trabajo que tiene relevancia es ‘Backing up WhatsApp data through the multi-device web client‘ [16]. Este trabajo está enfocado en **extraer** tanto **mensajes** como **ficheros multimedia** del cliente de WhatsApp Web **a través del navegador**. Para ello, desarrolla un script en JavaScript que, al copiarlo en la consola del navegador (presionando F12 y seleccionando ‘consola‘) y ejecutándolo, descarga todos los mensajes, así como los ficheros multimedia. En el reporte explica, de forma detallada, la investigación realizada para la elaboración del script.

Finalmente, algo que puede ser relevante, en este punto, es **IndexedDB** [17]. Es un almacén de clave-valor donde las claves y los valores son objetos de JavaScript. A su vez, otro término importante es **LevelDB**. Este es otro almacén de clave-valor puro [18]. Es decir, contiene claves y valores binarios, sin procesar. Como IndexedDB se implementa sobre LevelDB, es necesario que se le aporte la ‘traducción‘ de la estructura, los metadatos y registros sin procesar de LevelDB. Es decir, IndexedDB hace la función de una API, de manera que abstrae a los navegadores del sistema de archivos subyacente [19].

Esto se explica de forma detallada en los post de CCL Solutions Group [19][18]. Además, proveen una **herramienta** desarrollada en Python para la **extracción de los pares clave-valor** de los ficheros ‘**.LDB**‘. Este fichero ‘.LDB‘, es donde LevelDB almacena su contenido. El cual se mantiene

bloqueado para proteger su información [13]. En la misma ubicación del archivo '.LDB', se podrá encontrar un fichero '.LOG' y un fichero MANIFEST.

Capítulo 3

Metodología

3.1. Diseño de la investigación

Teniendo presente que el objetivo de este estudio es localizar y estudiar los artefactos que genera WhatsApp Web y resolver las preguntas iniciales, se ha procedido al análisis de los artefactos que se generan.

Destacar que, como la información sobre este ámbito es tan escasa y las versiones de WhatsApp tan cambiantes, no se ha seguido una metodología como tal. Se ha trabajado en base a los resultados que se han ido obteniendo y, de acuerdo con las necesidades, se ha buscado información y herramientas que pudieran satisfacerlas.

Así pues, esta investigación se ha compuesto de dos partes:

1. En la **primera**, se ha realizado el análisis sobre una **versión sin actualizar**, lo cual ocurrió de forma inesperada y se explicará más adelante. Esta se desarrolló en el entorno de pruebas 2 (ver 3.3 Entornos de pruebas).

   Concretamente, en este primer análisis se ha estudiado la herramienta BrowSwEx para la extracción de los datos de uno de los ficheros que genera el navegador. Se ha probado que la herramienta no funciona dado que el desarrollador no la ha actualizado. En este punto se ha procedido a estudiar el fichero del que obtiene los datos, el cual es un fichero ‘.log‘. Una vez analizado y estudiado se han realizado las correcciones pertinentes en la aplicación BrowSwEx para que se ejecute correctamente.

   Durante el proceso de estudio se ha visto que el navegador utiliza dos ficheros para almacenar la información y la aplicación BrowSwEx sólo extrae información de uno de ellos. Por tanto, se ha procedido al estudio y a la extracción de los datos del otro fichero, el cual tiene formato ‘.ldb‘.

   Finalmente, se ha realizado un script en Python para la automatización del proceso de extracción y análisis de los datos de los fichero ‘.log‘ y ‘.ldb‘.

2. En la **segunda** parte, la investigación se ha centrado en una **versión actualizada** y cercana a la realidad en el entorno de pruebas 1 (ver 3.3 Entornos de pruebas).

   En este segundo análisis se ha partido de la experiencia y conocimiento del análisis inicial. En primer lugar, se ha buscado la información almacenada a través de las herramientas de desarrollador del navegador para comprobar si se encuentran los mensajes almacenados en las bases de datos que almacena el navegador. Tras esto, se ha estudiado la nueva base de datos. Se ha verificado que los mensajes se encuentran en esta, pero cifrados. También se han podido obtener unos enlaces, los cuales descargan ficheros con extensión '.enc'. Estos ficheros se han podido descifrar y se ha comprobado que son los diferentes archivos multimedia que se han enviado y recibido en el dispositivo.

3.2. Recursos software utilizados

A continuación se realizará una breve descripción de las herramientas empleadas en este estudio.

- **DB Browser for SQLite**: es un software utilizado para administrar y visualizar bases de datos SQLite
- **Visual Studio Code** : es un entorno de desarrollo integrado (IDE) utilizado para escribir, depurar y compilar código
- **Wireshark**: es una herramienta de análisis de protocolos de red utilizada para examinar el tráfico de red en tiempo real
- **Python**: es el lenguaje de programación utilizado para el desarrollo de los scripts del proyecto.
- **7Zip**: es un programa de compresión de archivos utilizado para empaquetar y desempaquetar ficheros
- **Virtual Box**: es un software de virtualización que permite ejecutar múltiples sistemas operativos en una sola máquina. Esta versión específica de Virtual Box se refiere a la versión 6.1.32 instalada en el ordenador
- **XAMPP** : es un paquete de software que incluye un servidor web Apache, una base de datos MySQL y los intérpretes de PHP y Perl
- **BrowSwEx [20]**: es una de las herramientas analizadas en este estudio para la extracción de información de WhatsApp Web. Se verá en detalle más adelante
- **CyberChef**: es una herramienta de manipulación y análisis de datos utilizada en ciberseguridad y análisis forense. CyberChef proporciona funciones para decodificar, codificar y transformar datos en diversos formatos
- **WindowsMemoryExtractor** [21]: es una utilidad versátil, ya que puede extraer tanto módulos completos como regiones de memoria que cuenten con unas protecciones determinadas.
- **LibreOffice**: es una suite de código abierto que incluye aplicaciones como procesador de texto, hoja de cálculo y presentaciones, entre otras. Similar a la suite de Microsoft Office
- **Process Hacker**: es una herramienta de monitorización y gestión de procesos para sistemas Windows. Proporciona información detallada sobre los procesos en ejecución y permite realizar modificaciones en tiempo real, así como visualizar dónde está escribiendo cada uno
- **Sqlite Viewer SysTools**: es una herramienta de visualización y exploración de bases de datos SQLite
- **BurpSuite Community Edition**: es una herramienta para realizar pruebas de seguridad web
- **Avila Forensics 3.0** [22]: es una herramienta de forense digital móvil gratuita que permite la extracción de datos de dispositivos móviles
- **Microsoft Visual C++**: es un entorno de desarrollo integrado (IDE) y un conjunto de herramientas utilizado para desarrollar aplicaciones en lenguaje C++ en entornos Windows

3.3. Entornos de pruebas

Apreciaciones sobre los entornos de pruebas:

- Sólo se han realizado las pruebas con el sistema operativo Windows.
- El sistema operativo de los dispositivos móviles utilizados es Android.
- El navegador utilizado en todo momento ha sido Google Chrome.
- Se han utilizado dos versiones de WhatsApp en los dispositivos móviles.

Para este estudio se han utilizado 3 configuraciones. Se especifican a continuación:

Entorno de pruebas 1

Este entorno está destinado a utilizarse en el análisis 2 (2).

Se compone de un ordenador y un dispositivo móvil:

- Dispositivo Móvil
  - Pixel 4A (unrooted)
  - Android 12
  - WhatsApp 2.22.11.82
- Ordenador
  - Windows 10 Versión 22H1 Build 19044.1706
  - Google Chrome 102.0.5005.63
  - WhatsApp Web 2.2218.8
  - DB Browser for SQLite 3.12.2
  - Visual Studio Code 1.67.2
  - Wireshark 3.6.3
  - Python 3.9.6
  - 7Zip 21.07 (x64)
  - Virtual Box 6.1.32

Entorno de pruebas 2

Este entorno está destinado a utilizarse en el análisis 1 (1).

Se compone de una máquina virtual y un dispositivo móvil:

- Dispositivo Móvil
  - Huawei ALE-L21C432B634 (unrooted)
  - Android 6.0
  - WhatsApp 2.22.7.74
- Máquina Virtual
  - Windows 10 Versión 22H1 Build 19044.1645
  - Google Chrome 100.0.4896.12
  - WhatsApp Web 2.2212.8
  - XAMPP v3.3.0
  - BrowSwEx [20]
  - CyberChef v0.37.0
  - WindowsMemoryExtractor [21]
  - 7Zip 21.07 (x64)
  - DB Browser for SQLite 3.12.2
  - LibreOffice 7.3.2
  - Process Hacker 2.39.124
  - Sqlite Viewer SysTools 3.0.0
  - Visual Studio Code 1.66.2
  - Wireshark 3.6.3
  - Python 3.9.6
  - BurpSuite Community Edition v2022.2.4
  - Avila Forensics 3.0 [22]

Entorno de pruebas 3

Este entorno está destinado al desencriptado.

Se compone de una máquina virtual:

- Windows 10 Versión 22H1 Build 19044.1706
- Google Chrome 102.0.5005.63
- Microsoft Visual C++ 9.0
- Python 2.7.17 con los paquetes:
    - websocket-client
    - curve25519-donna
    - pycrypto
    - pyqrcode
    - protobuf

Para esta configuración, se han seguido las instrucciones de sigalor[15]. Además de clonar el repositorio de GitHub, para utilizar los componentes de la carpeta *backend*.

Capítulo 4

Resultados del Análisis 1

En este capítulo se presentarán los resultados obtenidos en la primera parte de la investigación.

Este primer análisis se ha llevado a cabo en el **entorno de pruebas 2**. Se ha realizado con un número de teléfono recién adquirido. El teléfono se ha restaurado al formato de fábrica y, tras esto se ha instalado la aplicación de WhatsApp. Se han agregado 3 contactos y se ha creado un grupo compuesto por el teléfono en cuestión y uno de los contactos. Se han generado mensajes, audios, imágenes y vídeos.

4.1. BrowSwEx

4.1.1. Extracción de las bases de datos del dispositivo móvil

Antes de la primera conexión a través del cliente Web, se ha llevado a cabo una extracción de la base de datos del dispositivo. Para ello, en primer lugar se ha realizado utilizando la herramienta 'Avilla Forensics 3.0' [22] (figura 4.1). Como no se ha obtenido ningún resultado, se optó por hacerlo de forma manual, tal y como se explica en el Anexo: Extracción de las BBDD de WhatsApp (figura 4.2).

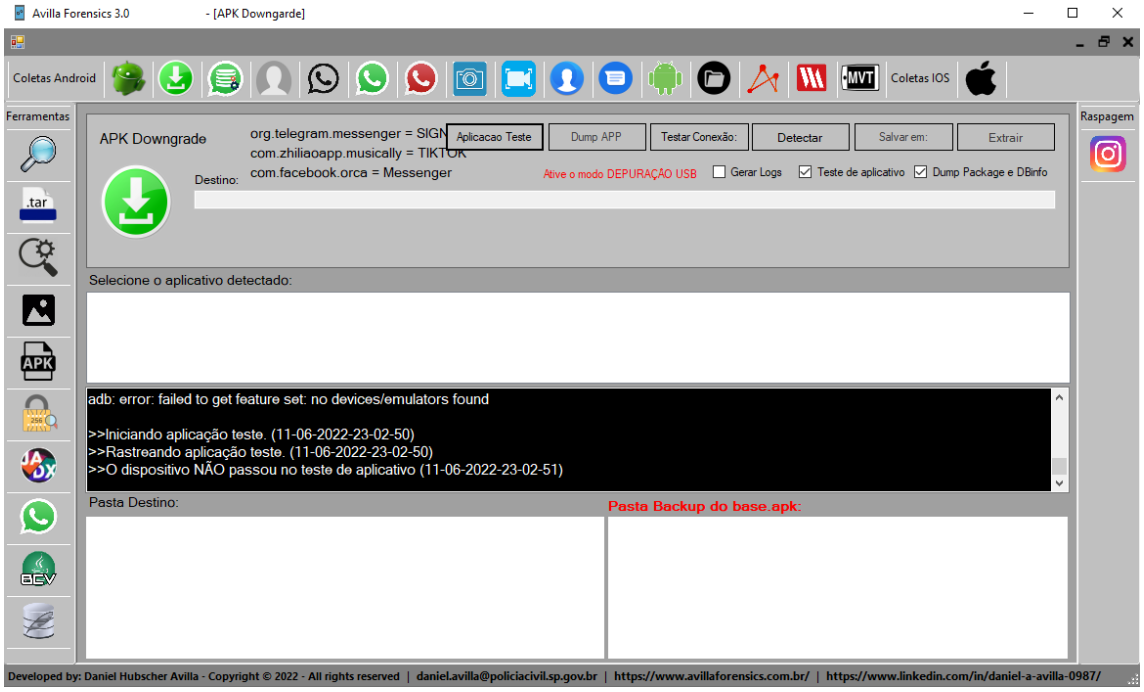


Figura 4.1: Página de inicio de APK Downgrade de la herramienta Avilla Forensics 3.0

```
                                    $ adb shell pm list packages -3 | grep com.whatsapp
package:com.whatsapp
                                    $ adb shell pm path com.whatsapp
package:/data/app/com.whatsapp-1/base.apk
                                    $ adb shell am force-stop com.whatsapp
                                    $ adb shell am kill com.whatsapp
                                    $ adb pull -a /data/app/com.whatsapp-1/base.apk
/data/app/com.whatsapp-1/base.apk: 1 file pul... skipped. 4.0 MB/s (39055777 bytes in 9.341s)
                                    $ adb shell pm uninstall -k com.whatsapp
Success
                                    $ adb install -r -d whatsapp_431.apk
Performing Push Install
whatsapp_431.apk: 1 file pushed, 0 skipped. 6.7 MB/s (18329558 bytes in 2.602s)
        pkg: /data/local/tmp/whatsapp_431.apk
Success
                                    $ adb shell monkey -p "com.whatsapp -c android.intent.cat
egory.LAUNCHER 1"
Events injected: 1
## Network stats: elapsed time=173ms (0ms mobile, 0ms wifi, 173ms not connected)
                                    $ adb shell "input keyevent 3"
                                    $ adb backup -f whatsapp.ab com.whatsapp
WARNING: adb backup is deprecated and may be removed in a future release
Now unlock your device and confirm the backup operation...
                                    $ adb install -r -d base.apk
Performing Push Install
base.apk: 1 file pushed, 0 skipped. 8.8 MB/s (39055777 bytes in 4.230s)
        pkg: /data/local/tmp/base.apk
Success
                                    $ java -jar abe.jar unpack whatsapp.ab whatsapp.tar
```

Figura 4.2: Extracción de las bases de datos de WhatsApp de forma manual

4.1.2. DevTools de Chrome

Tras la extracción de las bases de datos del dispositivo móvil, se ha iniciado sesión en WhatsApp Web y se ha hecho un análisis de lo que se puede visualizar a través de las 'DevTools'[1] (figuras 4.3 a 4.6).

Figura 4.3: 'Local Storage' del cliente de WhatsApp Web

En 'Local Storage', se puede encontrar información como (en orden de aparición según los datos resaltados en amarillo): marca de tiempo de la hora de inicio de sesión, el nombre del usuario, si ha marcado para que se le recuerde en el equipo, el sistema operativo del teléfono con el que se realizó la conexión y el número de teléfono. Además, aparecen unos campos (recuadrados en rojo) que, según 'sigalor' [15], parecen utilizarse en el desencriptado. Se pueden encontrar de nuevo en la figura 4.5.

Figura 4.4: Información almacenada en wawc/logs

[1] Se puede acceder a las 'DevTools' con la tecla F12 del navegador.

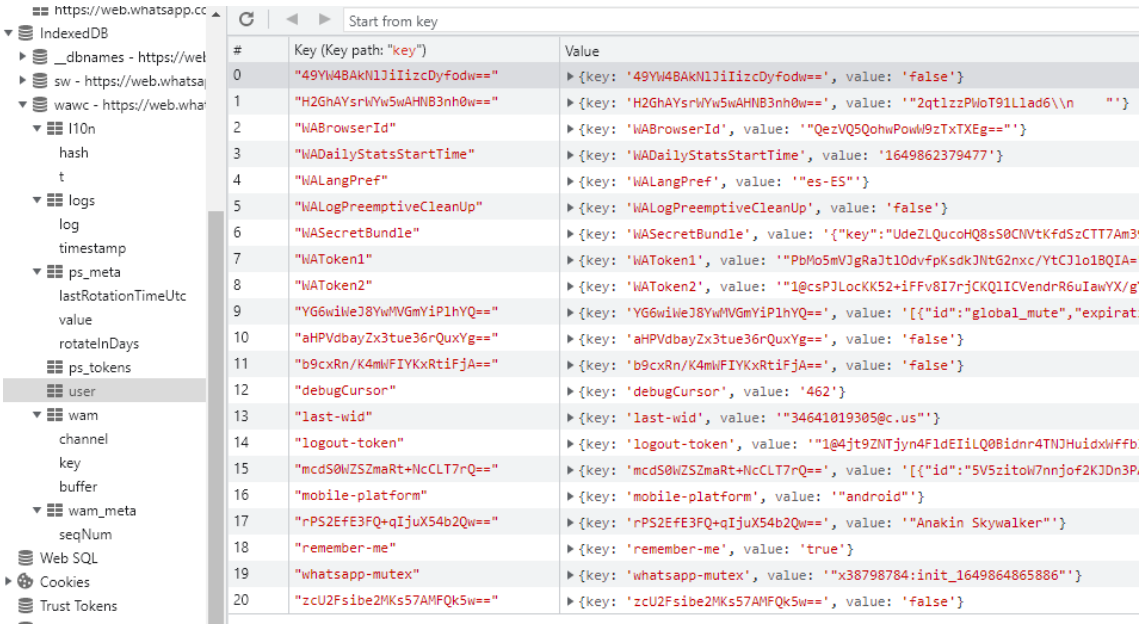


Figura 4.5: Información almacenada en wawc/user

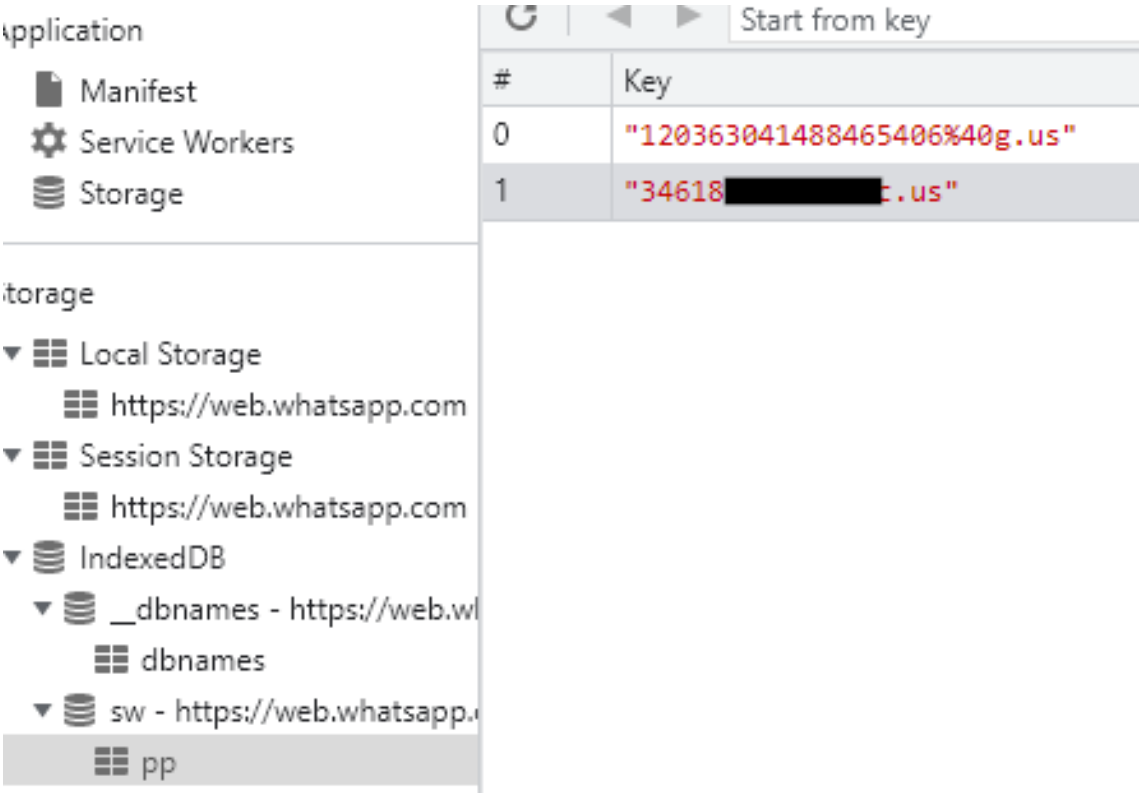


Figura 4.6: Chats que ha cargado

4.1.3. Localización de artefactos con Process Hacker

Con esa información, se procedió con la localización de los artefactos relevantes. Se utilizó la herramienta ***Process Hacker*** (figuras 4.7 y 4.8). La ruta es:

C:\Users\windows\AppData\Local\Google\Chrome\User Data\Default

IndexedDB\https_web.whatsapp.com_0.indexeddb.leveldb

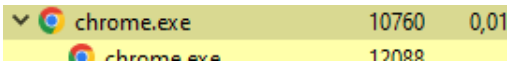


Figura 4.7: Identificador del proceso de Google Chrome

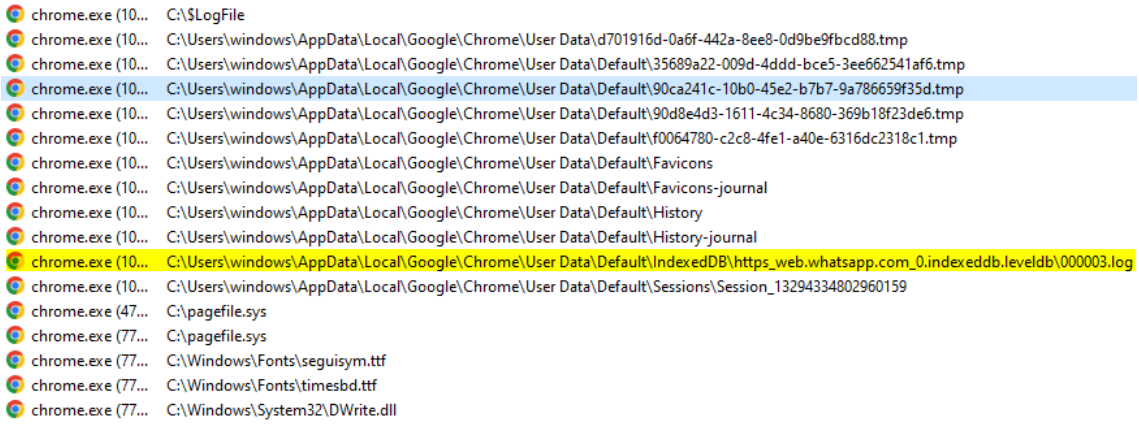


Figura 4.8: Localización de los artefactos generados por WhatsApp Web

4.1.4. Herramienta BrowSwEx

Tras la localización, el estudio se ha enfocado, en primer lugar, en la ejecución de la herramienta **BrowSwEx** [20]. Para ello se ha utilizado XAMPP[2] la cual es, según indica en su página, 'una distribución de Apache completamente gratuita, fácil de instalar que contiene MariaDB, PHP y Perl'.

La herramienta se ha descargado de su GitHub [20].

[2] `https://www.apachefriends.org/es/index.html`

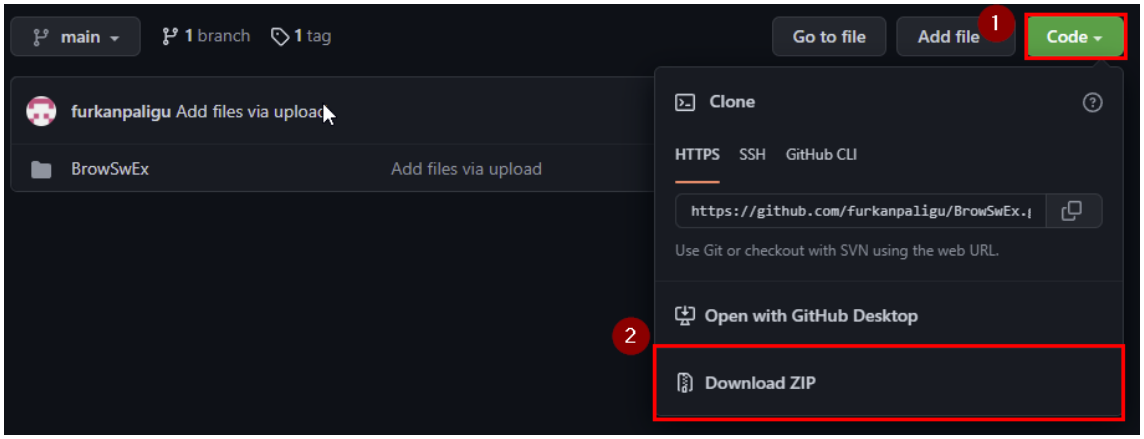


Figura 4.9: Descarga de la herramienta BrowSwEx

Se ha descomprimido el fichero obtenido en la ruta:

C:\xampp\htdocs\browswex

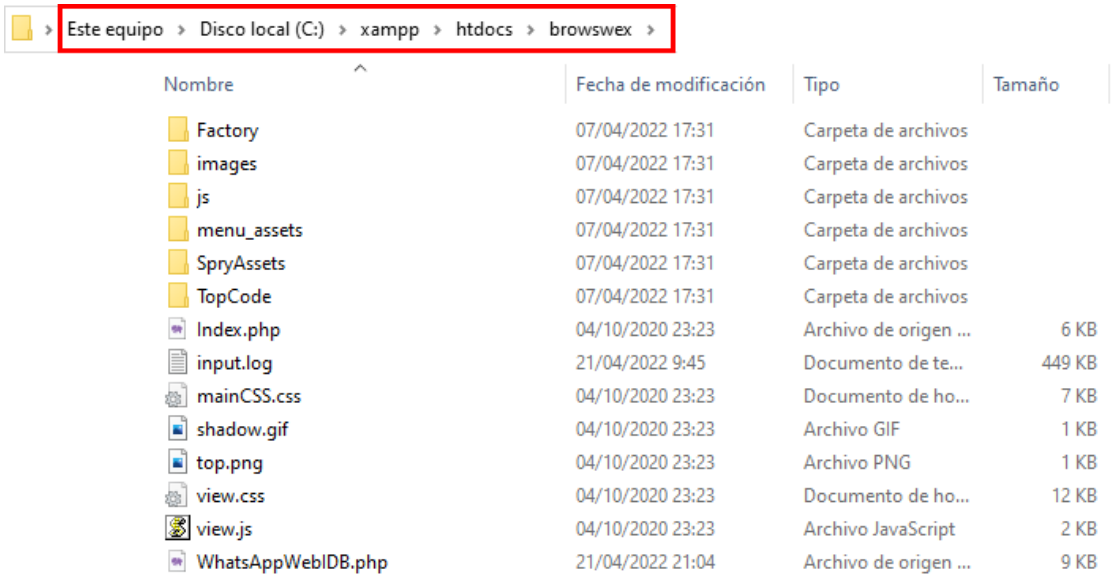


Figura 4.10: Ruta de instalación de la herramienta BrowSwEx

Como en la primera ejecución, se ha obtenido un resultado que no concordaba con la realidad, se ha procedido a analizar el código de la aplicación. En el fichero ***WhatsAppWebIDB.php*** se ha observado que estaba haciendo referencia a un fichero que tenía de muestra, llamado 'input.log' (figura 4.11). Este se encuentra en la ruta principal de la herramienta (figura 4.10). Por tanto, se ha sustituido este por el que se encuentra en la ruta:

C:\Users\windows\AppData\Local\Google\Chrome\User Data\Default

*IndexedDB\https_web.whatsapp.com_0.indexeddb.leveldb**000003.log***

El cual se ha renombrado para que tuviese el mismo nombre: ‘input.log’. Tras ver el resultado, seguía sin mostrar información relevante. Apenas muestra si el usuario estaba cerca del ordenador o el momento de inicio de sesión. Se ha continuado analizando el código contenido en ese fichero .php y se ha apreciado que aplica unas expresiones regulares (figura 4.12). En ese momento, se ha planteado la posibilidad de que estén desactualizadas, dado que WhatsApp está en continuo desarrollo.

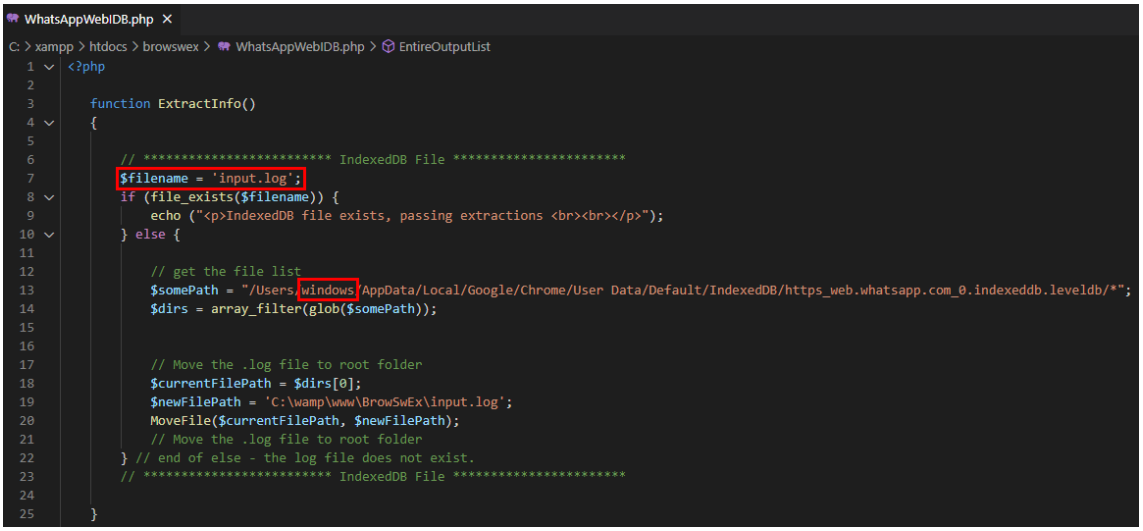


Figura 4.11: Extracto del código del fichero WhatsAppWebIDB.php (I)

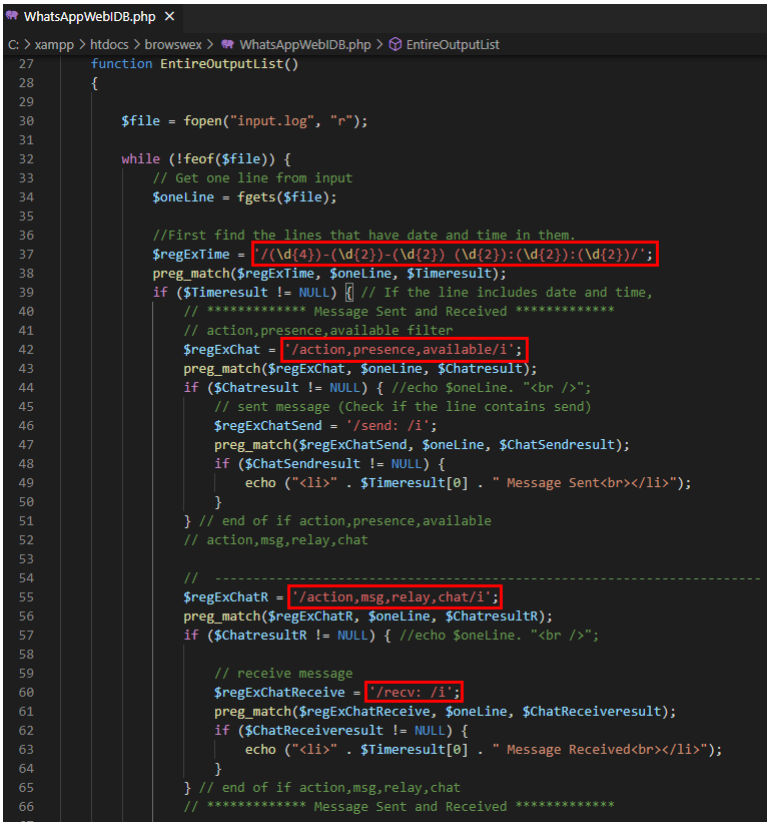


Figura 4.12: Extracto del código del fichero WhatsAppWebIDB.php (II)

4.1.5. Análisis del fichero '.log'

Se ha indagado en el fichero '.log', con la finalidad de actualizar las expresiones regulares de la herramienta. En un primer momento, se ha realizado un análisis manual. A través de este se han podido identificar:

- Elementos del *Local Storage* (figuras 4.13, 4.14)
- Mensaje recibido (figura 4.15)
- Inicio de sesión del usuario (figura 4.16)
- Chats (figura 4.17)
- Batería del dispositivo (figura 4.18)
- Imagen de perfil de uno de los chats (figura 4.19). Además, realizando una petición a través de Postman, se pudo obtener dicha imagen (figuras 4.20 y 4.21).

Figura 4.13: Extracto fichero 'log' (I). Elementos del Local Storage.

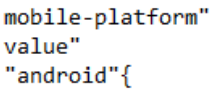


Figura 4.14: Extracto fichero 'log' (II). Elementos del Local Storage.

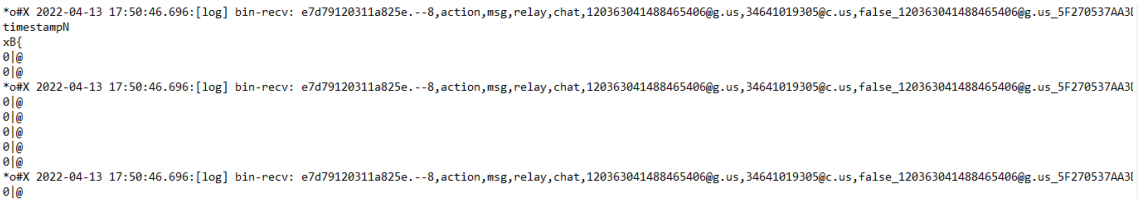


Figura 4.15: Extracto fichero 'log' (III). Mensaje recibido.

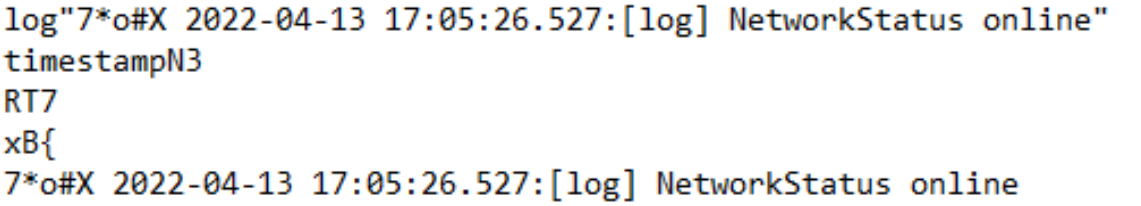


Figura 4.16: Extracto fichero 'log' (IV). Inicio de sesión.

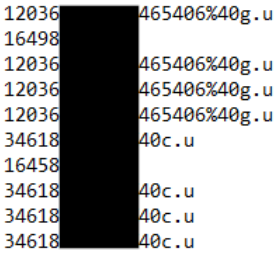


Figura 4.17: Extracto fichero 'log' (V). Chats.

```
log"F*o#X 2022-04-13 17:06:49.215:[log] bin-recv: 5,action,battery,49,false"
timestampN3#ch7
xB{
F*o#X 2022-04-13 17:06:49.215:[log] bin-recv: 5,action,battery,49,false
3#ch7
F*o#X 2022-04-13 17:06:49.215:[log] bin-recv: 5,action,battery,49,false
3#ch7
```

Figura 4.18: Extracto fichero 'log' (VI). Batería del dispositivo.

```
[{"id":"5V5zitoW7nnjof2KJDn3PA==","token":"3e3BezC0DfX5Oh+lEbtzHSqltwJ2TBgMKGZiEn3cPQQ=","tag":"1645844208","raw":null,"eurl":"https://pps.whatsapp.net/v/t61.24694-24
mcdS0WZSZmaRt+NcCLT7rQ=
mcdS0WZSZmaRt+NcCLT7rQ=
mcdS0WZSZmaRt+NcCLT7rQ=
B:>
lineI
log"R*o#X 2022-04-13 17:06:26.649:[log] models:profilePic:cache-save: profile_pic_thumb"
timestampNf
```

Figura 4.19: Extracto fichero 'log' (VII). Imagen de perfil.

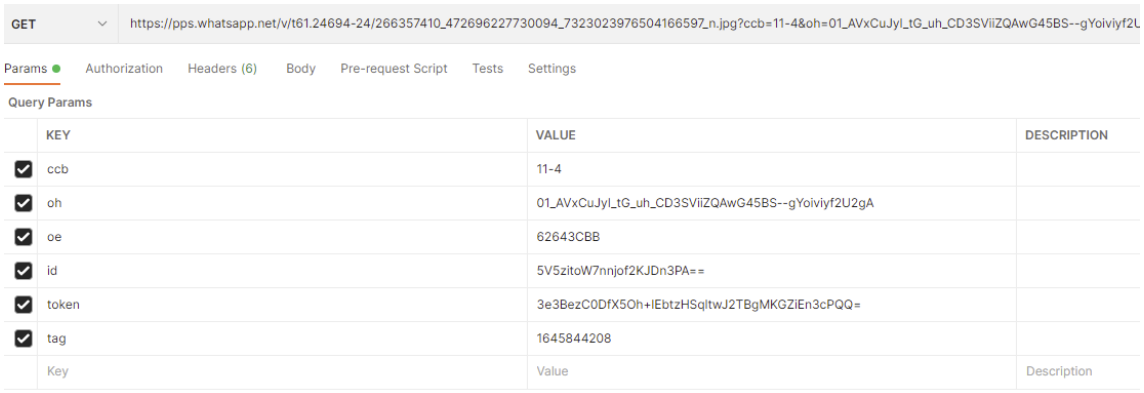


Figura 4.20: Petición realizada con Postman para obtener la foto de perfil.

Params ● Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

	KEY	VALUE
☑	ccb	11-4
☑	oh	01_AVxCuJyl_tG_uh_CD3SViiZQAwG45BS--gYoiviyf2U2gA
☑	oe	62643CBB
☑	id	5V5zitoW7nnjof2KJDn3PA==
☑	token	3e3BezC0DfX5Oh+lEbtzHSqltwJ2TBgMKGZiEn3cPQQ=
☑	tag	1645844208
	Key	Value

Body Cookies Headers (13) Test Results

Figura 4.21: Imagen de perfil obtenida a través de Postman.

4.1.6. Actualización de las expresiones regulares

Tras este análisis, se ha cargado el fichero '.log' en la herramienta CyberChef y, con la 'receta' *Regular expression*, se ha tratado de obtener los datos identificados anteriormente. Principalmente, a través de este método, se ha obtenido:

Usuario online

En la figura 4.22, se indica que el usuario se conectó a las 09:40:01 del día 21 de abril de 2022. Lo volvió a hacer a las 11:31:26, ese mismo día.

Regex
.*NetworkStatus online.*

^ and $

Output

2022-04-21 09:40:01.022:[log] NetworkStatus online" timestampN□éb□t□xB{□□□2□□l
2022-04-21 11:31:26.918:[log] NetworkStatus online" timestampNÍD□e·□xB{□□□2□□l

Figura 4.22: Usuario online

Recordar usuario

En la figura 4.23 se puede ver que, cuando se inició sesión, se había marcado la casilla para que se recordara el dispositivo.

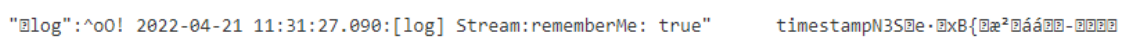


Figura 4.23: Recordar usuario

Mensaje recibido

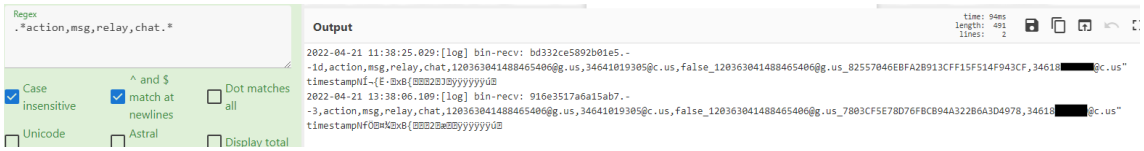


Figura 4.24: Mensaje recibido (I)

Para entender y corroborar los indicios sobre qué representaban los datos extraídos con esta expresión regular (figura 4.24), se ha procedido a la extracción de las bases de datos de WhatsApp del teléfono.

Así pues y, como se puede ver en las figuras 4.25 y 4.26, el campo *key_id* corresponde con el extraído de la base de datos msgstore (figura 4.25), así como los campos *timestamp*[3] y *from_me*. Además, se puede ver que los mensajes se han enviado a través de un grupo en el cual el usuario participa, según se puede ver en la figura 4.26.

De esta manera, se interpreta la información extraída de la siguiente manera:

<timestamp>: [log]: bin-recv: <desconocido>,action,msg,relay,chat,<raw_string>,

false_<raw_string>_<key_id>,<número teléfono emisor>@c.us

```
1  SELECT _id, from_me, key_id, origin, datetime(timestamp/1000, 'unixepoch') as timestamp,
2  message_type, text_data FROM message WHERE _id = '37' or _id = '40'
3
```

	_id	from_me	key_id	origin	timestamp	message_type	text_data
1	37	0	82557046EBFA2B913CFF15F514F943CF	0	2022-04-21 09:38:25	0	Sí
2	40	0	7803CF5E78D76FBCB94A322B6A3D4978	0	2022-04-21 11:38:06	0	Ok

Figura 4.25: Mensaje recibido (II)

[3]Los timestamps de la base de datos del teléfono se encuentran en formato UTC. Los timestamps extraídos del fichero .log o .ldb están en el mismo que el del sistema. En este caso UTC+2.

```
1  SELECT _id, user, server, type, raw_string
2  FROM jid WHERE user='120363041488465406'
```

	_id	user	server	type	raw_string
1	6	120363041488465406	g.us	1	120363041488465406@g.us

Figura 4.26: Mensaje recibido (III)

Mensaje enviado

En la figura 4.27 aparece que se han enviado dos mensajes. El primero a las 10:49:11 y el segundo a las 12:00:45 del día 21 de abril de 2022.

El formato de este log es, comparándolo con la información extraída de la base de datos del dispositivo (figura 4.28), la siguiente:

<timestamp>: [log] send: <key_id>, action, message,chat,,<key_id>"

Destacar que, tal y como se ha podido ver, **los mensajes enviados a través de WhatsApp Web**, se almacenan en la base de datos del dispositivo con un ***key_id* que comienza por '3EB0'**.

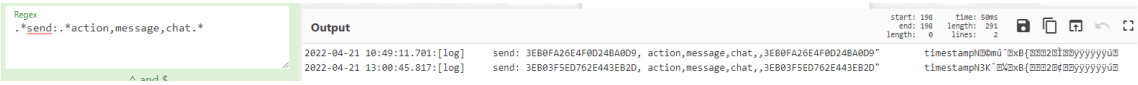


Figura 4.27: Mensaje enviado (I)

```
1  SELECT _id, from_me, key_id, origin, datetime(timestamp/1000, 'unixepoch') as timestamp,
2  message_type, text_data FROM message WHERE key_id='3EB0FA26E4F0D24BA0D9'
3  OR key_id='3EB03F5ED762E443EB2D'
```

	_id	from_me	key_id	origin	timestamp	message_type	text_data
1	35	1	3EB0FA26E4F0D24BA0D9	0	2022-04-21 08:49:12	0	hecho
2	39	1	3EB03F5ED762E443EB2D	0	2022-04-21 11:00:46	0	A que está muy chula?

Figura 4.28: Mensaje enviado (II)

Vídeo recibido

En la figura 4.29 aparece la recepción de un vídeo. Del mismo modo, aparece en la bases de datos del dispositivo (figura 4.30). El formato es el siguiente:

<timestamp>:[log] bin-recv:<desconocido>,action,msg,relay,video,<teléfono emisor>@c.us,<teléfono receptor>@c.us,false_<teléfono emisor>@c.us_<key_id>,"

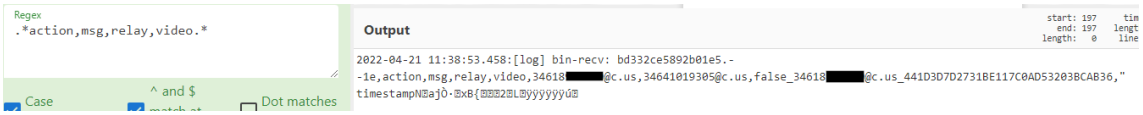
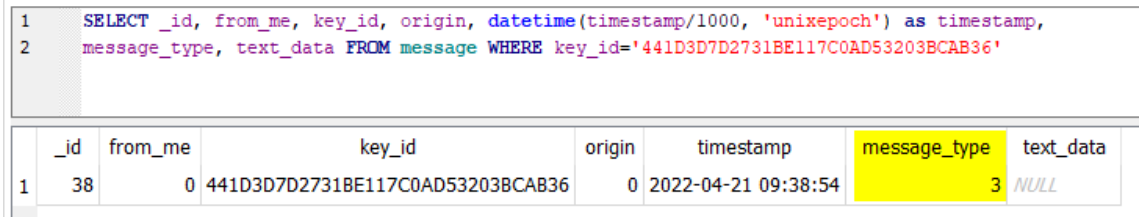


Figura 4.29: Vídeo recibido (I)



```
1  SELECT _id, from_me, key_id, origin, datetime(timestamp/1000, 'unixepoch') as timestamp,
2  message_type, text_data FROM message WHERE key_id='441D3D7D2731BE117C0AD53203BCAB36'
```

	_id	from_me	key_id	origin	timestamp	message_type	text_data
1	38	0	441D3D7D2731BE117C0AD53203BCAB36	0	2022-04-21 09:38:54	3	NULL

Figura 4.30: Vídeo recibido (II)

Usuario alejado del ordenador

En la figura 4.31 se puede ver los instantes en los que el dispositivo se encontraba alejado del ordenador. No se sabe exactamente la forma a través de la cual WhatsApp Web lo calcula.

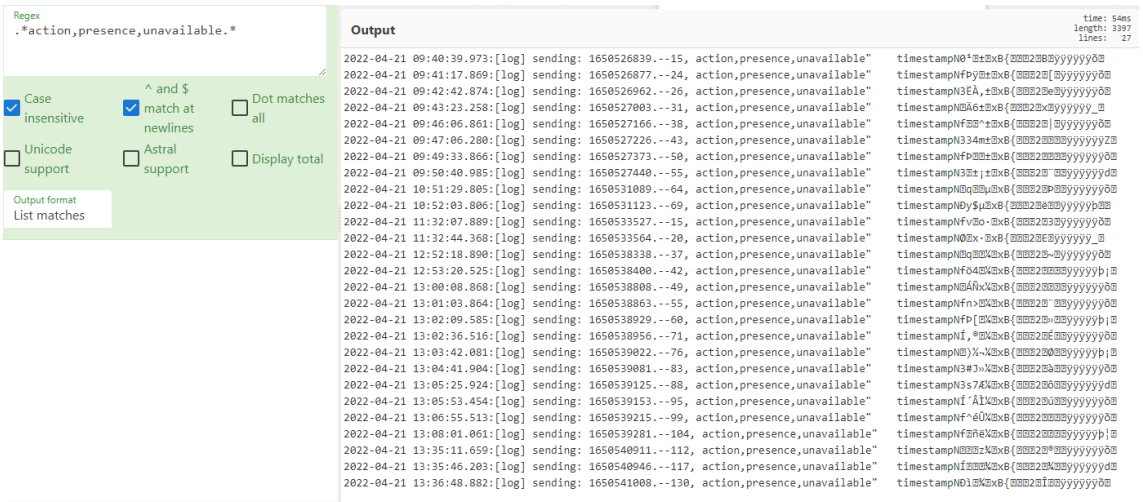


Figura 4.31: Usuario alejado del ordenador

Usuario cerca del ordenador

En la figura 4.32 se puede ver los instantes en los que el dispositivo se encuentra cerca del ordenador. No se sabe exactamente la forma a través de la cual WhatsApp Web lo calcula.

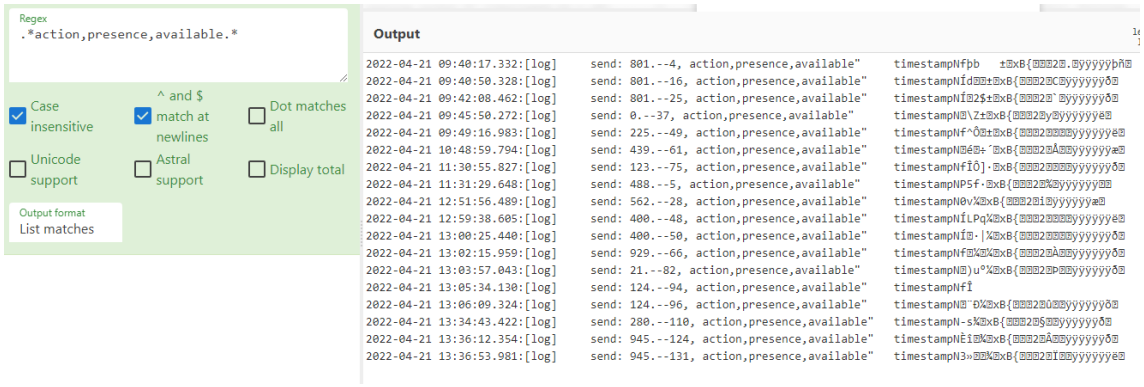


Figura 4.32: Usuario alejado del ordenador

	Expresión regular
Usuario online	.*NetworkStatus online.*
Recordar usuario	.*stream:rememberMe: true.*
Mensaje recibido	.*action,msg,relay,chat.*
Mensaje enviado	.*send:.*action,message,chat.*
Vídeo recibido	.*action,msg,relay,video.*
Usuario alejado del ordenador	.*action,presence,unavailable.*
Usuario cerca del ordenador	.*action,presence,available.*

Tabla 4.1: Expresiones regulares actualizadas

Algunas de las expresiones de la herramienta BrowSwEx no se han actualizado ya que no se ha encontrado información al respecto en el fichero '.log' analizado.

Tras esto se ha vuelto a ejecutar la herramienta BrowSwEx. El resultado se puede ver en la figura 5.7.

Figura 4.33: Resultados obtenidos tras la actualización de la herramienta BrowSwEx

4.1.7. Conclusiones sobre la herramienta BrowSwEx

El análisis de esta herramienta ha creado una línea de trabajo. En esta se ha estudiado el fichero '.log' y, utilizando expresiones regulares, se ha obtenido información sobre el envío y recepción de mensajes, marcas de tiempo sobre inicios de sesión o la presencia o no del dispositivo.

Se han actualizado los campos más relevantes en la herramienta y se ha comprobado que el resultado de su ejecución ha sido correcto y de acuerdo con la realidad del experimento.

4.2. LevelDB

4.2.1. Extracción del fichero '.ldb'

Habiendo realizado el estudio del fichero '.log' se ha realizado la siguiente pregunta: ¿Se podrá extraer información del fichero '.ldb'?

Tras un periodo de búsqueda, se han encontrado los artículos de 'CCL Solutions Group' [19][18]. En estos, además de la información, se ha aportado una serie de scripts desarrollados en Python 3 para la extracción de la información, tanto del fichero '.log', como del fichero '.ldb'. Se pueden encontrar en GitHub [23].

Como primera prueba, se ha clonado el repositorio y, en él, se ha copiado la carpeta con los artefactos generados por WhatsApp Web que se encontraba en:

C:\Users\windows\AppData\Local\Google\Chrome\User Data\Default

IndexedDB\https_web.whatsapp.com_0.indexeddb.leveldb

Quedando esta carpeta como se ve en la figura 4.34.

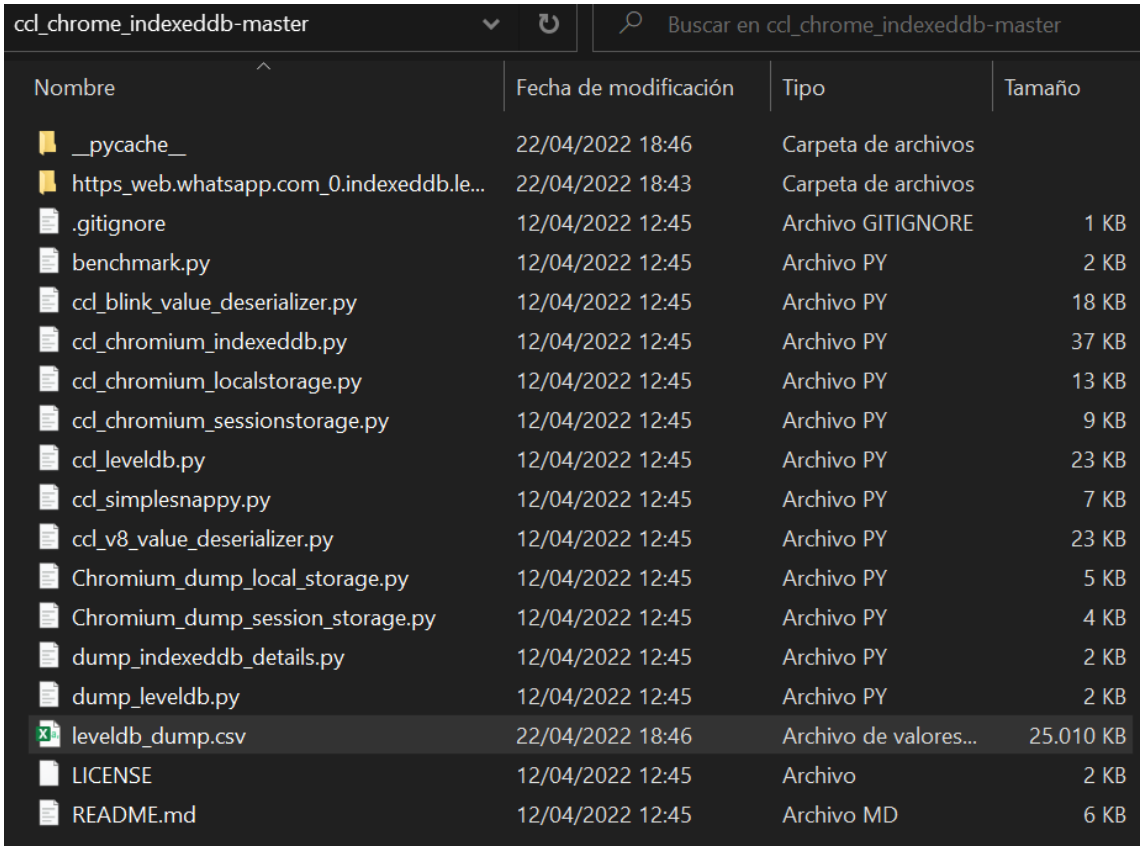


Figura 4.34: Extracción de la información del fichero '.log' y '.ldb'

Utilizando el comando (figura 4.35):

```
python dump_leveldb.py https_web.whatsapp.com_0.indexeddb.leveldb
```

Se ha generado el fichero ‘leveldb_dump.csv’, el cual contiene toda la información extraída de la base de datos *LevelDB*.

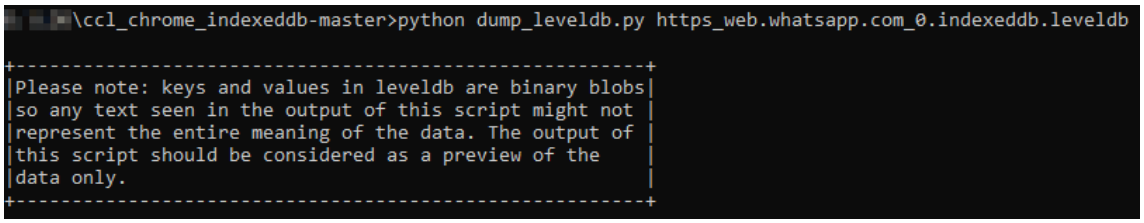


Figura 4.35: Comando para la extracción de la información del fichero ‘.log’ y ‘.ldb’

Este fichero consta de las siguientes columnas (figura 4.36):

- key-hex
- key-text
- value-hex
- value-text
- origin_file
- file_type
- offset
- seq
- state
- was_compressed

Los campos relevantes son ‘key-text’ y ‘value-text’. A partir de los cuales se podrá obtener información en un formato algo legible.

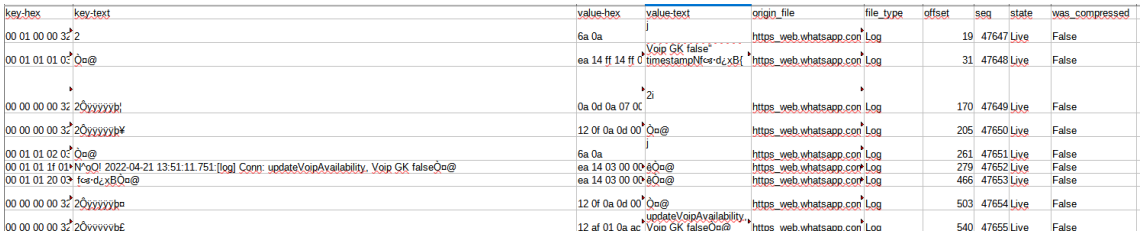


Figura 4.36: Columnas del fichero ‘leveldb_dump.csv’

Analizando esas dos columnas, se puede ver información como la que sigue:

Campos '*key*'

Se ha identificado la aparición, en múltiples ocasiones, de un campo codificado en Base64.

+ÿÿo"key"
658093837"buffer"ÀV0FNAzADB0AF1gAwCwiADQdXaW5kb3dzgBEIMi4yMjEyLjgQFVAvL0
VhYognARVDaHJvbWUgMTAwLjAuNDg5Ni4xMjcYeQI4wwICiAsDBkNocm9tZSgPA4g9AwZ
IVUFXRUmIPwMHaHdBTEUtSIhBAw9BTEUtTDIxQzQzMkI2MzSIQwMDNi4wCGsDKIMDiO
0DCTIuMjIuNy43NDh5BgQ4aQ1/iI8OCng5MDg1NTMxNDiImRECMTA5dAP/NgEKUC8xRW
FiSGkNgAA51gf2EgUSBoIIDnF1ZXJ5X3NldHRpbmdzMgdQMgNpQgKaAEYEoQE="chann
el"regular{

Figura 4.37: Campo '*key*'

ú□Ë®|ÓÝüß¶ï}÷«WAM0@Ö0
□
Windows□2.2212.8P//Eab□'Chrome 100.0.4896.127y8Ã□
Chrome(□=HUAWEI□?hwALE-H□AALE-L21C432B634□C6.0k(□□í 2.22.7.748y8i
□□
x908553148□□109tÿ6
P/1EabHi
□9Öö□query_settings2P2iB□Fjr§□ékz
¥j

Figura 4.38: Campo '*key*' decodificado

Entre la información obtenida destaca (resaltada en la figura 4.38 y en orden de aparición):

- Sistema operativo del ordenador que ejecuta el cliente de WhatsApp Web.
- Versión de WhatsApp que utiliza en la Web.
- Versión del navegador.
- Marca, modelo, número de compilación y versión de Android del dispositivo.
- Versión de WhatsApp que se ejecuta en el dispositivo móvil.

Imágenes de perfil de todos los chats en un mismo campo

øÿÿo"key"mcdS0WZSZmaRt+NcCLT7rQ=="value"□
[{"id":"nd+LgnU/dTFIRXNAUkLHuw==","token":"lm2VAWzDUnqSy2t8d6SywS72dGf4k8gVVz7d+ZZxkMM=","tag":"1650382977","raw":null,"eurl":"https://pps.whatsapp.net/v/t61.24694-24/240375823_1505769919825582_3434642598191681959_n.jpg?ccb=11-4&oh=3a96d776b7d7badcc5bb3e4739bd4557&oe=6270FC0A"},{"id":"cGCmQ9sNeyYOewSzgJfDXw==","token":"6rldSiuMGWuHcjk7lOSuUXFk5ln7F3g9JkLebObTV1I=","tag":"1634818208","raw":null,"eurl":"https://pps.whatsapp.net/v/t61.24694-24/247790098_355024473083769_645888699716905562_n.jpg?ccb=11-4&oh=01_AVw-3xx8U6NngRjjrAtbspLhKB-L3Pa21ghOBB25FgQVmQ&oe=6271EDE0"},{"id":"5V5zitoW7nnjof2KJDn3PA==","token":"kZ/gPF0IZTveBegEm1Ftvtn4ej3WNZrRbTGJEXX0FT8=","tag":"1645844208","raw":null,"eurl":"https://pps.whatsapp.net/v/t61.24694-24/266357410_472696227730094_7323023976504166597_n.jpg?ccb=11-4&oh=01_AVyEasJz3SQolkiofzvw7YUcADNOFKp8hXDf9N-asumAcw&oe=62716BBB"},{"id":"tz7qd+yKTl8qqgpgblQ9iQ==","token":"lCouaabQK32hVirpE1BvJU1cv6C0yeKkJalgd8NCktM=","tag":"1649851193","raw":null,"eurl":"https://pps.whatsapp.net/v/t61.24694-24/187788597_1169897387156517_2797890526478894753_n.jpg?ccb=11-4&oh=01_AVwHZQAZd-sM3jKJen-WijFHms99Q-0gSzSF4HEkkiV-rQ&oe=62735467"}]{

Figura 4.39: Campo con URL a la imagen de perfil de los chats

Del texto que aparece en la figura 4.39 se pueden extraer las URLs de la figura 4.40.

- https://pps.whatsapp.net/v/t61.24694-24/240375823_1505769919825582_3434642598191681959_n.jpg?ccb=11-4&oh=3a96d776b7d7badcc5bb3e4739bd4557&oe=6270FC0A
- https://pps.whatsapp.net/v/t61.24694-24/247790098_355024473083769_645888699716905562_n.jpg?ccb=11-4&oh=01_AVw-3xx8U6NngRjjrAtbspLhKB-L3Pa21ghOBB25FgQVmQ&oe=6271EDE0
- https://pps.whatsapp.net/v/t61.24694-24/266357410_472696227730094_7323023976504166597_n.jpg?ccb=11-4&oh=01_AVyEasJz3SQolkiofzvw7YUcADNOFKp8hXDf9N-asumAcw&oe=62716BBB
- https://pps.whatsapp.net/v/t61.24694-24/187788597_1169897387156517_2797890526478894753_n.jpg?ccb=11-4&oh=01_AVwHZQAZd-sM3jKJen-WijFHms99Q-0gSzSF4HEkkiV-rQ&oe=62735467

Figura 4.40: URLs de las imágenes de perfil obtenidas

Adicionalmente, se han encontrado, de nuevo:

- Usuario en línea
- Mensajes enviados y recibidos
- Vídeos recibidos
- Números de teléfono e IDs de grupos

- Batería del dispositivo
- Presencia del usuario

Por último, destacar que, al parecer, el fichero '.log' es rotativo, mientras que el fichero '.ldb' no lo es. Esto hace que si se analiza únicamente el fichero '.log', puede que falte información de días previos. En cambio, si se analiza el fichero '.ldb', se encontrará mucha más información.

4.2.2. Conclusiones sobre LevelDB

Lo más importante de esta sección es que se ha conseguido extraer información del fichero '.ldb'.

Además, contiene tanto la información del fichero '.log' como otra adicional de intervalos temporales anteriores a este. Destacar que se ha visto que el fichero '.log' es rotativo y elimina la información de días previos.

4.3. Script

Tras haber visto una forma de extraer la información de la base de datos LevelDB, de verificar que en ella se encuentran datos de relevancia y que, utilizando expresiones regulares, es posible seleccionar esta información y hacer una extracción, se ha ideado un script que realice esta labor de manera automática.

Este script se encuentra en la información anexa. Tiene como nombre:

whatsapp_parser_tipo_1.py.

En la figura 4.41 se encuentra una prueba del funcionamiento y la información que se obtiene.

```
PS C:\Users\windows\Desktop\Parser> & C:/Users/windows/AppData/Local/Programs/Python/Python39/python.exe c:/Users/windows/Desktop/Parser/whatsapp_parser_log.py
El usuario del que se extraerán los datos es: windows
El fichero .log tiene el nombre: 000004.log
Timestamps encontrados: 528
Total usuario online: 0
Total escribiendo: 0
Total enviado: 0
Total eliminado: 0
Total batería: 4
Total leidos: 1
Total status upload: 0
Total media files: 0
Total recibidos: 1
Total offline: 8
Total online: 5
El fichero .ldb tiene el nombre: 000005.ldb
Timestamps encontrados: 2625
Total usuario online: 3
Total escribiendo: 0
Total enviado: 3
Total eliminado: 0
Total batería: 27
Total leidos: 4
Total status upload: 0
Total media files: 0
Total recibidos: 4
Total offline: 30
Total online: 16
PS C:\Users\windows\Desktop\Parser>
```

Figura 4.41: Ejecución del script whatsapp_parser_tipo_1.py

Los resultados son los ficheros de las figuras 4.42 a 4.44.

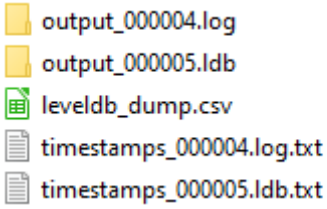


Figura 4.42: Ficheros obtenidos tras la ejecución de whatsapp_parser_tipo_1.py

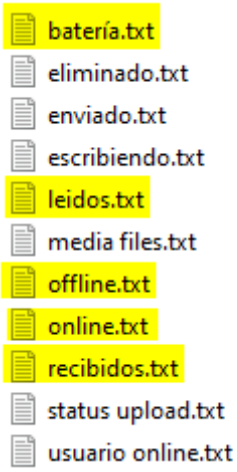


Figura 4.43: Ficheros de la carpeta 'output_000004.log' (los resaltados contienen información)

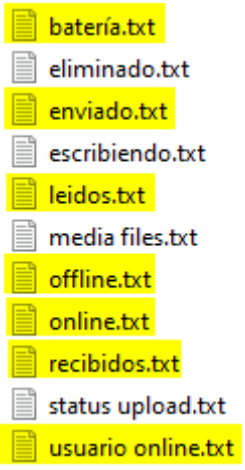


Figura 4.44: Ficheros de la carpeta 'output_000005.ldb' (los resaltados contienen información)

En las siguientes figuras se muestra la **información obtenida**.

```
2022-05-02 13:37:49.076:[log] bin-recv: 2,action,battery,83,false"      timestampNÍL[Pˆ□xB{□□    2 ¨□ □ÿÿÿÿÿÿú□
2022-05-02 13:37:49.169:[log] bin-recv: 3,action,battery,82,false"      timestampN P[Pˆ□xB{□□    2 ¨□ □ÿÿÿÿÿÿð□
2022-05-02 13:46:34.535:[log] bin-recv: 9,action,battery,81,false"      timestampN  ÌÐˆ□xB{□□    2 Ã□ □ÿÿÿÿÿÿú□
2022-05-02 14:03:24.496:[log] bin-recv: 5,action,battery,80,false"      timestampN3kbÇ‰□xB{□□    2 õ□ □ÿÿÿÿÿÿú□
```

Figura 4.45: Fichero output_000004.log/batería.txt. Campos: 4

```
2022-05-02 14:01:32.522:[log]     send: 533.--256, action,chat,read,{"fromMe":false,"remote":"34618[redacted]@c.us",
"id":"3EB0C1A874B4B011475A","_serialized":"false_34618[redacted]@c.us_3EB0C1A874B4B011475A"},1"     timestampNš±Ü«‰□xB{□□    2 Ù□ □ÿÿÿÿÿÿÒ□
```

Figura 4.46: Fichero output_000004.log/leídos.txt. Campos: 1

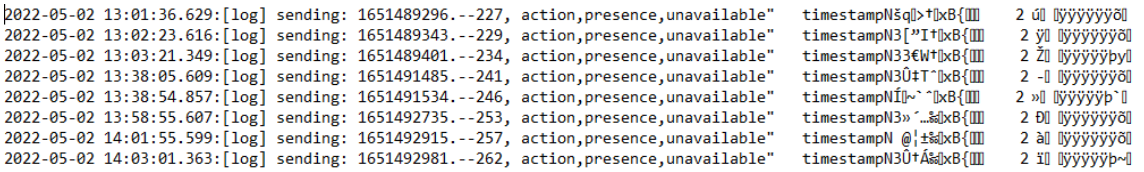


Figura 4.47: Fichero output_000004.log/offline.txt. Campos: 8

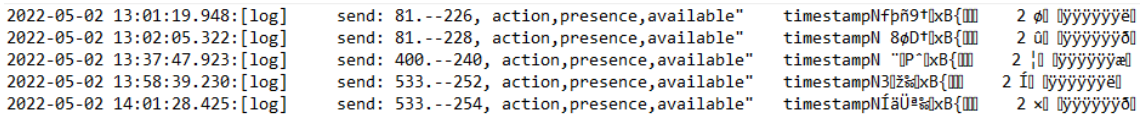


Figura 4.48: Fichero output_000004.log/online.txt. Campos: 5

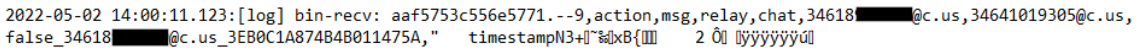


Figura 4.49: Fichero output_000004.log/recibidos.txt. Campos: 1

Figura 4.50: Fichero output_000005.ldb/batería.txt. Campos: 27

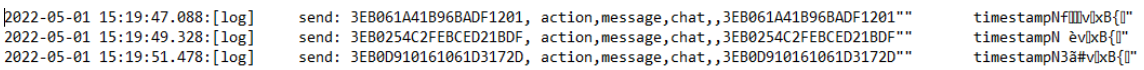


Figura 4.51: Fichero output_000005.ldb/enviado.txt. Campos: 3

Figura 4.52: Fichero output_000005.ldb/leídos.txt. Campos: 4

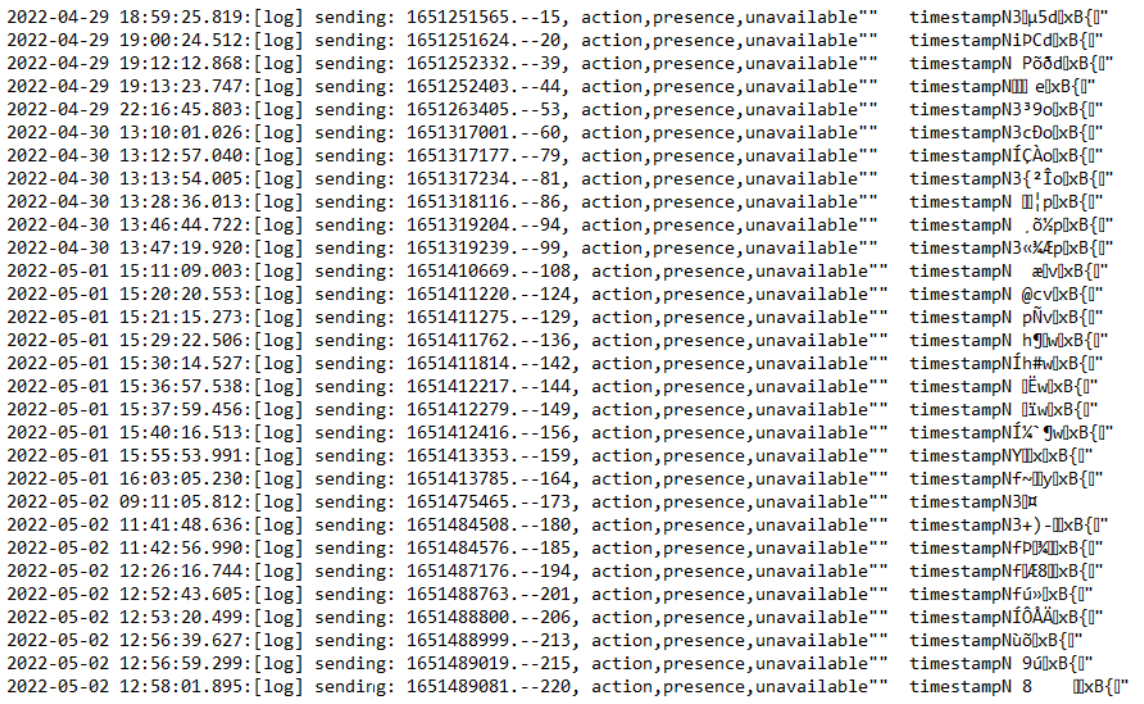


Figura 4.53: Fichero output_000005.ldb/offline.txt. Campos: 30

```
2022-04-29 18:58:55.233:[log]     send: 533.--4, action,presence,available""    timestampN 8(.d□xB{□"
2022-04-29 19:11:42.347:[log]     send: 623.--25, action,presence,available""   timestampNÍtaéd□xB{□"
2022-04-30 13:09:35.444:[log]     send: 403.--59, action,presence,available""   timestampN®~o□xB{□"
2022-04-30 13:11:26.423:[log]     send: 403.--61, action,presence,available""   timestampN3Sªo□xB{□"
2022-04-30 13:13:37.616:[log]     send: 403.--80, action,presence,available""   timestampNfÊo□xB{□"
2022-04-30 13:46:28.240:[log]     send: 114.--93, action,presence,available""   timestampN 8Ë¹p□xB{□"
2022-05-01 15:19:32.153:[log]     send: 664.--115, action,presence,available""  timestampN3□xv□xB{□"
2022-05-01 15:29:06.109:[log]     send: 274.--135, action,presence,available""  timestampNÑ□w□xB{□"
2022-05-01 15:29:41.259:[log]     send: 274.--137, action,presence,available""  timestampNf""□w□xB{□"
2022-05-01 15:36:40.416:[log]     send: 274.--143, action,presence,available""  timestampN Àxw□xB{□"
2022-05-01 15:39:41.156:[log]     send: 278.--155, action,presence,available""  timestampNÍ¤-w□xB{□"
2022-05-01 15:55:38.260:[log]     send: 278.--157, action,presence,available""  timestampN 8Fx□xB{□"
2022-05-02 11:41:26.019:[log]     send: 465.--179, action,presence,available""  timestampN3s§□xB{□"
2022-05-02 12:52:24.792:[log]     send: 175.--200, action,presence,available""  timestampNÍôE·□xB{□"
2022-05-02 12:56:22.738:[log]     send: 799.--212, action,presence,available""  timestampNÍLMñ□xB{□"
2022-05-02 12:56:43.430:[log]     send: 799.--214, action,presence,available""  timestampN 0lö□xB{□"
```

Figura 4.54: Fichero output_000005.ldb/online.txt. Campos: 16

```
2022-04-30 13:11:46.683:[log] bin-recv: 26e2b272bc9d1cde.--2,action,msg,relay,chat,120363041488465406@g.us,
34641019305@c.us,false_120363041488465406@g.us_3EB0B5CE938F6855D044,34618[redacted]@c.us""  timestampNÉy¯o□xB{□"

2022-04-30 13:12:04.971:[log] bin-recv: 26e2b272bc9d1cde.--3,action,msg,relay,chat,120363041488465406@g.us,
34641019305@c.us,false_120363041488465406@g.us_3EB091F8B2B582F25F77,34618[redacted]@c.us""  timestampNYÞ³o□xB{□"

2022-04-30 13:12:18.380:[log] bin-recv: 26e2b272bc9d1cde.--4,action,msg,relay,chat,120363041488465406@g.us,
34641019305@c.us,false_120363041488465406@g.us_3EB0049EDAB678A96536,34618[redacted]@c.us""  timestampNÍ-·o□xB{□"

2022-04-30 13:13:08.482:[log] bin-recv: 26e2b272bc9d1cde.--5,action,msg,relay,chat,34618[redacted]@c.us,
34641019305@c.us,false_34618[redacted]@c.us_3EB0E0581265959F8E62,"" timestampNÍlqÃo□xB{□"
```

Figura 4.55: Fichero output_000005.ldb/recibidos.txt. Campos: 4

```
2022-04-29 18:58:28.512:[log] NetworkStatus online""    timestampN3{¹'d□xB{□"
2022-04-29 22:16:40.291:[log] NetworkStatus online""    timestampN àë□o□xB{□"
2022-04-30 13:28:34.723:[log] NetworkStatus online""    timestampN P□¦p□xB{□"
```

Figura 4.56: Fichero output_000005.ldb/usuario online.txt. Campos: 3

4.4. Conclusión

En este primer análisis, se ha utilizado la herramienta **BrowSwEx**. Se ha visto que no funcionaba correctamente y se ha inspeccionado el código. Se ha detectado que utilizaba **expresiones regulares** sobre el fichero '**.log**'. Así pues, **se ha analizado** este fichero. Con los resultados obtenidos, se han actualizado algunas de las expresiones regulares y la herramienta ha proporcionado un resultado de acuerdo con la realidad.

Como se ha visto que BrowSwEx únicamente utiliza el fichero '.log', se ha decidido buscar información para **extraer** los **datos** del fichero '**.ldb**'. Se han encontrado unos **scripts** desarrollados en **Python 3**, los cuales realizan esta función, la extracción de datos de los ficheros '**.ldb**'.

Tras esto se ha decidido llevar a cabo un script en Python que incorporase los anteriores y que **automatice** la labor de **extracción** empleando expresiones regulares sobre la información obtenida.

No obstante, llegados a este punto, y no antes, se ha visto que **algo estaba ocurriendo que no era normal**. Es necesario tener el dispositivo conectado a la misma red y que se encuentre encendido. De otro modo, no es posible la sincronización de WhatsApp Web. Pero esto no debería ocurrir dado que, a partir de la versión 2.21.23.5 [24] esto no sería necesario. La versión que se está utilizando en el dispositivo es la 2.22.7.74. ¿Qué ocurre entonces?

No se tiene una respuesta a esa pregunta, realmente. La **hipótesis** que se baraja es que, al realizar en primer lugar, la extracción de las bases de datos de WhatsApp a través de APK Downgrade, algo debió modificarse en el dispositivo, haciendo que la versión Web de WhatsApp detectase el dispositivo como desactualizado y, por compatibilidad, utilizase una versión diferente a la actual.

Por tanto, el script adjunto de nombre 'whatsapp_parser_tipo_1.py' sólo podrá ser utilizado en aquellos casos en los que se necesite el dispositivo conectado para que WhatsApp Web se ejecute. Lo que debería ocurrir en versiones de WhatsApp anteriores a la 2.21.23.5 (Android).

Tras esta aclaración, queda finalizado el primer análisis.

Capítulo 5

Resultados del Análisis 2

En esta sección se van a presentar los resultados obtenidos en la segunda parte de la investigación.

El principal cambio con respecto al análisis anterior es la versión de WhatsApp Web que se ejecuta. En esta versión no es necesario tener el dispositivo conectado para poder interactuar con la aplicación. Por tanto, la base de datos LevelDB, cambia.

Se parte de la información que ha aportado el análisis previo sobre cómo extraer la información de la base de datos LevelDB. Tras su estudio, se emplearán expresiones regulares para obtener la información relevante.

5.1. DevTools de Chrome

Buscando información relacionada sobre WhatsApp Web en Chrome, se ha encontrado el artículo 'Backing up WhatsApp data through the multi-device web client'. En él, utiliza una herramienta que el autor desarrolla en JavaScript para la extracción de los mensajes y ficheros multimedia a través del navegador. También explica el proceso que sigue. Esta herramienta se ha probado y funciona correctamente. No obstante, como este estudio está enfocado en los artefactos, no se va a explicar con detalle. Simplemente decir que, gracias a su artículo, se han empleado las 'DevTools' para poder dar respuesta a la pregunta sobre si **se guardaban los mensajes en la base de datos LevelDB**.

A través de las herramientas de desarrollador, se han podido ver contactos (figura 5.1), grupos y otra información sin ningún tipo de cifrado. Eso quiere decir que, todo lo que aparezca ahí sin cifrar, se podrá encontrar del mismo modo en la base de datos LevelDB. A continuación se explica cómo se ha comprobado esta afirmación.

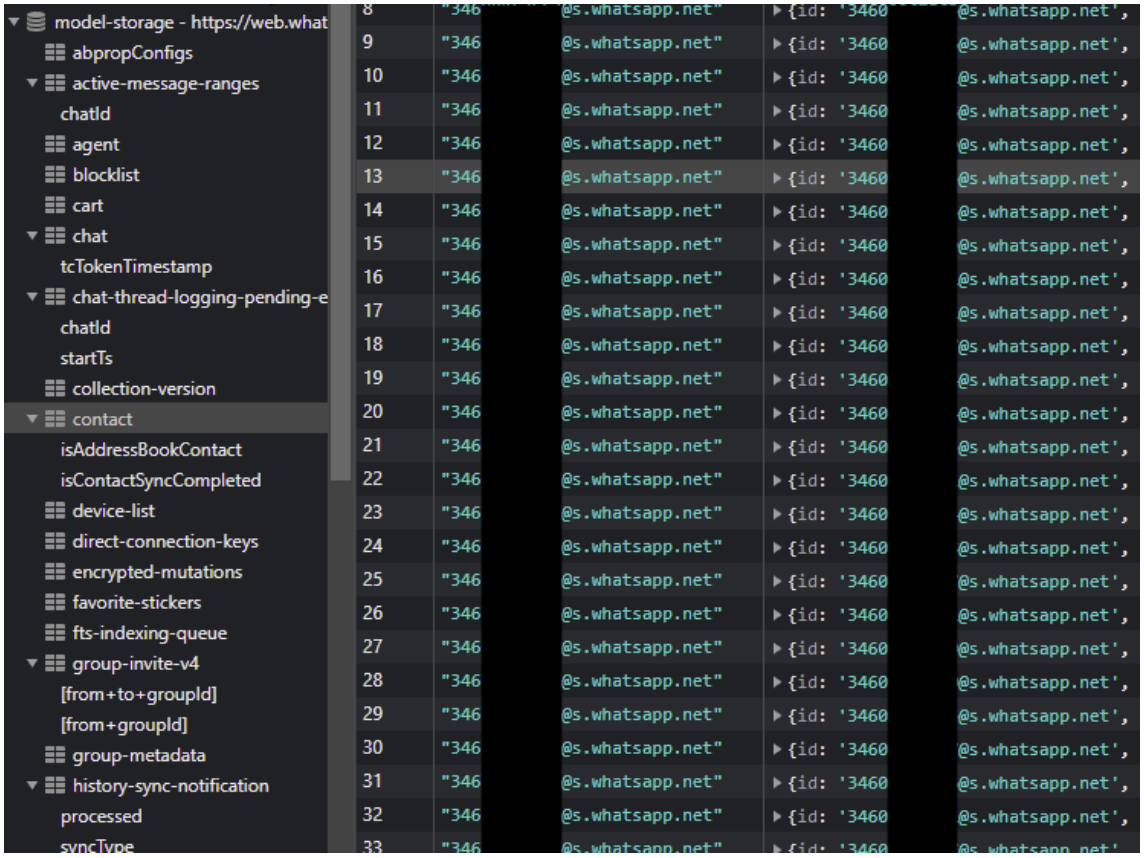


Figura 5.1: Visualización de contactos a través de 'DevTools'

En el apartado '*message*' (figura 5.2), se pueden ver los mensajes. No obstante, se encuentran cifrados (figura 5.3).

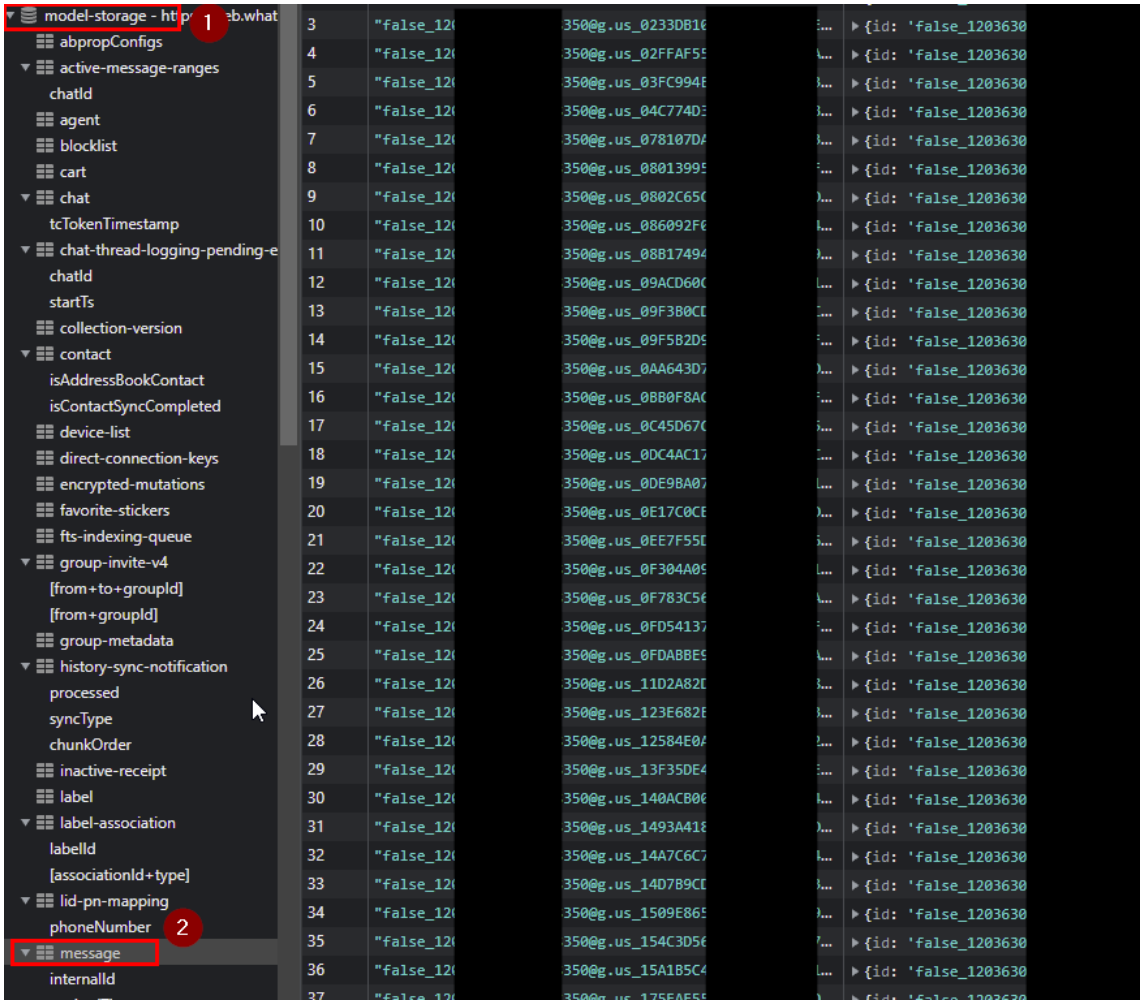


Figura 5.2: Base de datos de mensajes visualizada a través de 'DevTools'

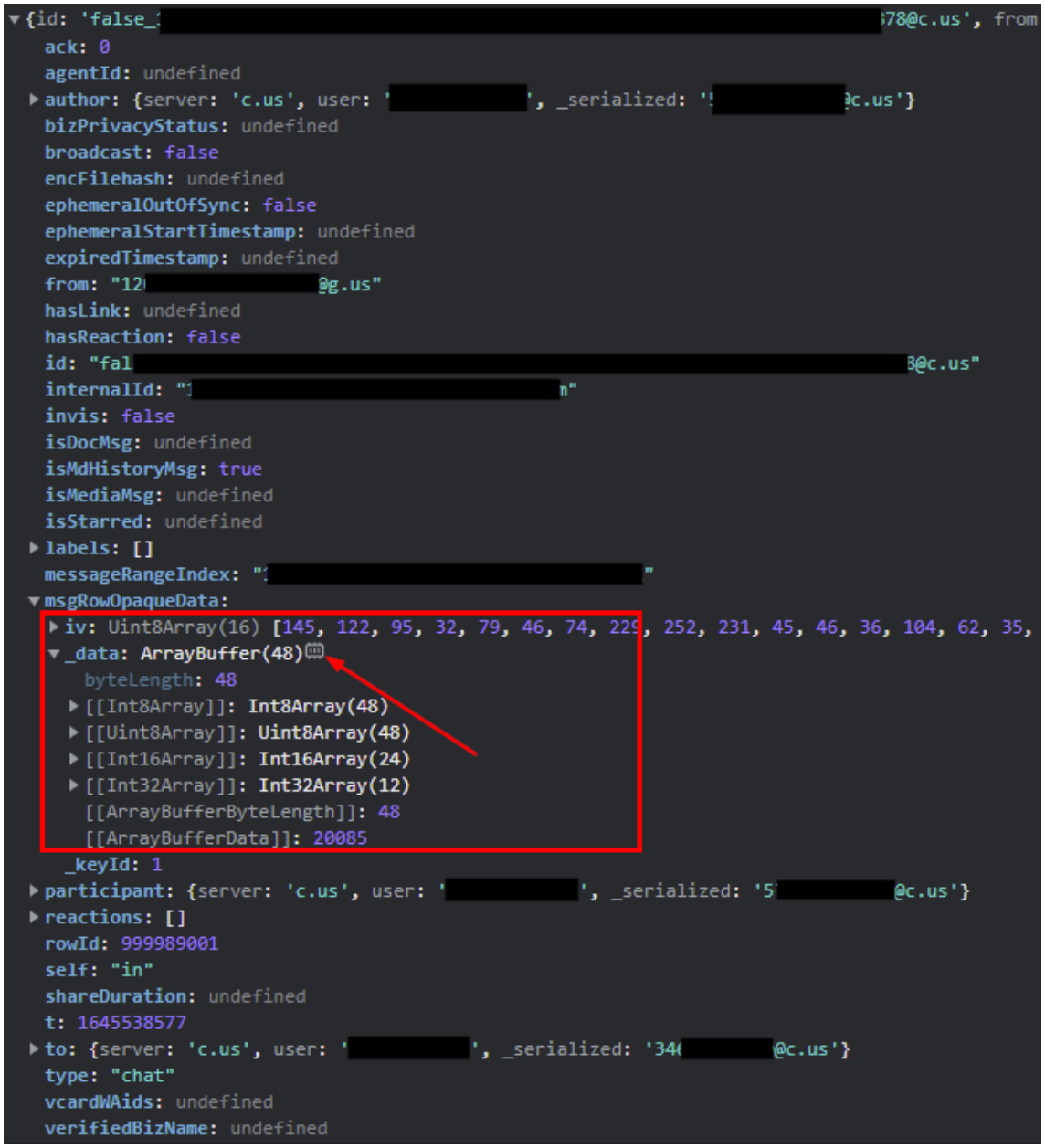


Figura 5.3: Mensaje visualizado a través de 'DevTools'

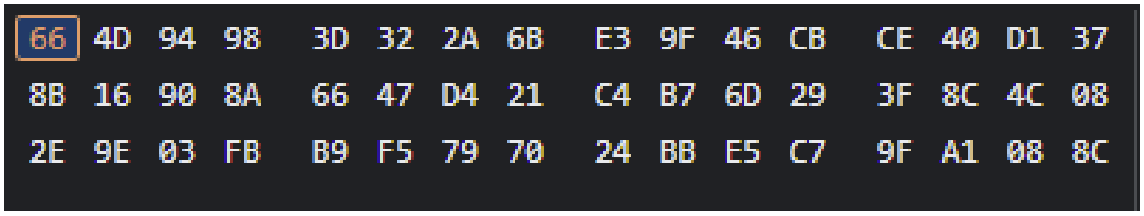


Figura 5.4: Mensaje cifrado (campo '*_data*') visualizado a través de 'DevTools'

Para saber si los mensajes se encuentran en la base de datos LevelDB, se ha ejecutado el script con un pequeña modificación, de manera que en vez de extraer los datos convertidos a texto, únicamente se muestran en hexadecimal (figura 5.5).

```
151  def dumpea_leveldb(ruta, output_path):
152      input_path = ruta
153      # output_path = "leveldb_dump.csv"
154
155      leveldb_records = ccl_leveldb.RawLevelDb(input_path)
156
157      with open(output_path, "w", encoding="utf-8", newline="") as file1:
158          writes = csv.writer(file1, quoting=csv.QUOTE_ALL)
159          writes.writerow(
160              [
161                  "value-hex",
162              ])
163
164          for record in leveldb_records.iterate_records_raw():
165              # Realmente saca datos del .ldb y del .log
166              # Con el if sólo se obtendrán los que pertenezcan al .ldb
167              if (record.file_type.name == 'Ldb'):
168                  writes.writerow([
169                      record.value.hex(" ", 1),
170                  ])
```

Figura 5.5: Modificación del script *whatsapp_parser_tipo_2.py* para la extracción de la base de datos en hexadecimal

Buscando en la información obtenida, se ha encontrado una cadena que coincide completamente con lo que, a través de 'DevTools', se mostraba como '*_data*' del mensaje.

```
153569  58 50 72 65 76 69 65 77 54 79 70 65 49 00 22 0f 64 6f 4e 6f 74 50 6c 61 79 49 6e 6c 69 6e 65 5f 22 09 74 65 78 74 43 6f 6c 6f 72 5f 22 0f 62 61 63 6b 67 72
153570  ad 9b 3c 0e d0 aa e1 5d 51 99 91 46 e0 15 14 06 22 02 69 76 42 10 03 ad bb c8 a7 a9 24 d3 9f 61 a1 f2 53 3c cd a6 56 42 00 10 f0 47 22 06 5f 6b 65 79 49 64
153571  51 42 30 66 4d 94 98 3d 32 2a 6b e3 9f 46 cb ce 40 d1 37 8b 16 90 8a 66 47 d4 21 c4 b7 6d 29 3f 8c 4c 08 2e 9e 03 fb b9 f5 79 70 24 bb e5 c7 9f a1 08 8c 22
153572  c0 fc 95 f0 a2 7a d1 00 32 5a d1 88 fb 4f 23 31 ff 00 2f 43 0d cf bd 7a 54 35 1c 66 14 a5 97 24 15 81 0b f0 f5 ae d7 80 b6 f9 ca fe 87 98 4f e7 22 02 69 76
153573  58 50 72 65 76 69 65 77 54 79 70 65 49 00 22 0f 64 6f 4e 6f 74 50 6c 61 79 49 6e 6c 69 6e 65 5f 22 09 74 65 78 74 43 6f 6c 6f 72 5f 22 0f 62 61 63 6b 67 72
153574  4b 7d 39 26 d0 91 ad 88 86 a1 c2 62 c8 fc ae fc 18 5c fd 80 20 55 50 22 68 75 1a b7 4b 5f 95 6c 22 02 69 76 42 10 43 b0 de 2c 0b a4 7f 3e 7c 83 df b5 2f a1
153575  1b 2f e4 a3 03 70 86 3e 97 6f d2 9e 93 30 1a 59 21 c6 57 6c b2 e3 b3 98 d2 2d 66 d9 66 dd 8a a1 39 29 22 0c de aa ca 2a b5 8b 52 58 22 c8 78 5b 22 02 69 76
153576  3b 68 61 73 52 65 61 63 74 69 6f 6e 46 22 07 61 67 65 6e 74 49 64 5f 22 11 64 65 70 72 65 63 61 74 65 64 4d 6d 73 33 55 72 6c 22 4d 68 74 74 70 73 3a 2f 2f
153577  58 50 72 65 76 69 65 77 54 79 70 65 49 00 22 0f 64 6f 4e 6f 74 50 6c 61 79 49 6e 6c 69 6e 65 5f 22 09 74 65 78 74 43 6f 6c 6f 72 5f 22 0f 62 61 63 6b 67 72
153578  58 50 72 65 76 69 65 77 54 79 70 65 49 00 22 0f 64 6f 4e 6f 74 50 6c 61 79 49 6e 6c 69 6e 65 5f 22 09 74 65 78 74 43 6f 6c 6f 72 5f 22 0f 62 61 63 6b 67 72
153579  48 92 ec 08 35 4a 0a 6c 11 bb 68 6c 4f 1b 70 3c 20 65 7d b6 2d 71 76 76 d9 e1 f2 6c 15 8e 75 d4 22 02 69 76 42 10 0d 63 36 fc e9 4e 14 e7 19 95 65 3d f2 8c
153580  35 d1 ab 12 48 21 a9 7d 2c 8f 69 03 32 00 40 a5 22 02 69 76 42 10 6c 8c a2 2f b1 83 b4 31 74 74 69 b6 d1 87 80 09 56 42 00 10 f0 47 22 06 5f 6b 65 79 49 64
153581  73 52 65 61 63 74 69 6f 6e 46 22 07 61 67 65 6e 74 49 64 5f 22 11 64 65 70 72 65 63 61 74 65 64 4d 6d 73 33 55 72 6c 22 4d 68 74 74 70 73 3a 2f 2f 6d 6d 67
153582  bc 38 69 13 0f 7e e9 50 76 c8 51 0e 84 2e d7 30 be 53 07 f7 4c 68 f8 16 22 15 a9 ff 7f 72 3f 4d 22 02 69 76 42 10 22 38 cf 91 5d c4 e7 8c 9a 8a ed 7a 26 97
```

Figura 5.6: Mensaje cifrado encontrado en la extracción de LevelDB

Por tanto, **los mensajes sí que se guardan en la base de datos LevelDB**, aunque se encuentran **cifrados**.

Por tanto, todo lo que se puede ver a través de 'DevTools' en 'IndexedDB', se podrá encontrar en la base de datos 'LevelDB' almacenada en el equipo.

5.2. Script

El script que aquí se propone tiene como finalidad la extracción automática de la información que almacena la nueva versión de WhatsApp Web en el equipo.

Este script se encuentra en la información anexa. Tiene como nombre:

whatsapp_parser_tipo_2.py.

En las siguientes figuras se muestra una prueba de su funcionamiento.

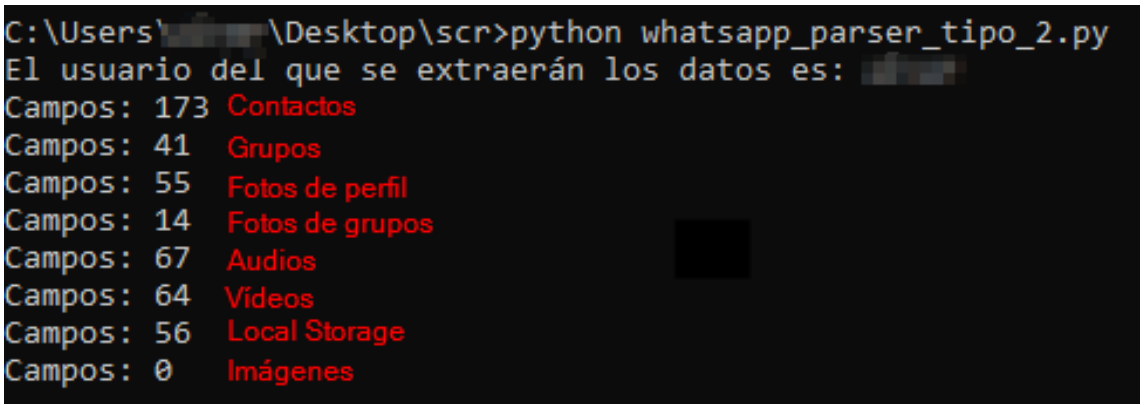


Figura 5.7: Ejecución del script

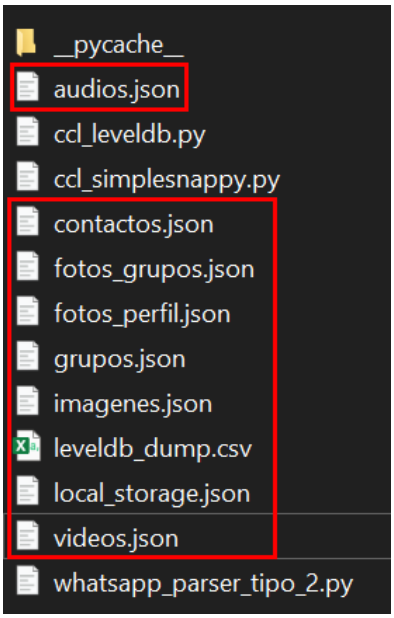


Figura 5.8: Ficheros que genera el script

```
{
    "w/Ju35l0lBuuQ+FY/pp8GF886OL87S96dpgRG8MktWU=": {
        "mimetype": "ogg",
        "url": "https://pps.whatsapp.net/v/t62.7117-24/21318727_1068957677303258_2489016678751841976_n.enc?ccb=11-4&oh=01_AVwfM9GbYNUk23479ZX1grkvXfp9WgrtG2Vi_L43hAYBUQ&oe=62913217"
    },
    "Nxyy+QGxq42JBnSRk+ytb6DzAznVu9stiZ/SQtWjzqI=": {
        "mimetype": "ogg",
        "url": "https://pps.whatsapp.net/v/t62.7117-24/30419117_400882081885118_2628622855802607190_n.enc?ccb=11-4&oh=01_AVxvC7Nu1VcEoUQ54pjNggn4nvXtXd06NhT4UgZxAZ45HQ&oe=6295C0A7"
    },
    "S5Doi3antHP9vM7ztwgL8iW8BLLfFN/13zfgBL3Bu5A=": {
        "mimetype": "ogg",
        "url": "https://pps.whatsapp.net/v/t62.7117-24/31376522_693387198534675_3776930437409613187_n.enc?ccb=11-4&oh=01_AVyPJ6FAJpHJI56NqeRxehR7kZAvmrYL9aJkXqkjWY3O7Q&oe=6290A19A"
    },
    "yGWSEf8J0c4d1CJ7P1ZL0Q+9pV5xkwqQZW3+oWR5UJw=": {
        "mimetype": "ogg",
        "url": "https://pps.whatsapp.net/v/t62.7117-24/41193490_1147905105992006_5364189217195529374_n.enc?ccb=11-4&oh=01_AVxJRppqx0GApJd7XsdvIk_oRuiRYsJ4k2CQfco9w_EMhg&oe=629201B3"
    },
    "y3PHtfcmq6vVQ2TdnvBKI7q7o5v5Q9d3YEZ1UIv/61Q=": {
        "mimetype": "ogg",
        "url": "https://pps.whatsapp.net/v/t62.7117-24/35298807_546284367033990_1174456624913039520_n.enc?ccb=11-4&oh=01_AVzkV-mSkcQ8RUAi6IXkaxlX-5Ta2AJ5NcUqUDGqhrCTCQ&oe=629F9EA3"
    },
    "Wny9Plnq47seHvkkfG+uH3YgsJmLWwmy3s+XTJ2dvKs=": {
        "mimetype": "ogg",
        "url": "https://pps.whatsapp.net/v/t62.7117-24/21315197_793229511655831_104106925491391325_n.enc?ccb=11-4&oh=01_AVwNiYhBlfjhJltD4JycS8gKocoIXJbLE8w2QzVfq94DKQ&oe=628FD265"
    },
    "EYaZ78ZfyUgXMJu9RyZxC38k16pinY4RkM39gnETJWE=": {
        "mimetype": "ogg",
        "url": "https://pps.whatsapp.net/v/t62.7117-24/34600628_145702897977400_1660245139476547267_n.enc?ccb=11-4&oh=01_AVzAlnpSP7GKOt2TG5CV-Z9V9icNWIxNpGlWmbEd7PYY3w&oe=629929E8"
    },
    "KgtJ6pVkZFPjorCsrxeUw9GfHNfbacNzYQ0e+jJIjkw=": {
        "mimetype": "ogg",
        "url": "https://pps.whatsapp.net/v/t62.7117-24/31856280_964596084211149_2287045157864519364_n.enc?ccb=11-4&oh=01_AVxthgzrfbMiehYUNtecf6TRGkPt6PjoW0nWvcdpX0zK_w&oe=629A5445"
    },
```


Figura 5.9: Fichero audios.json

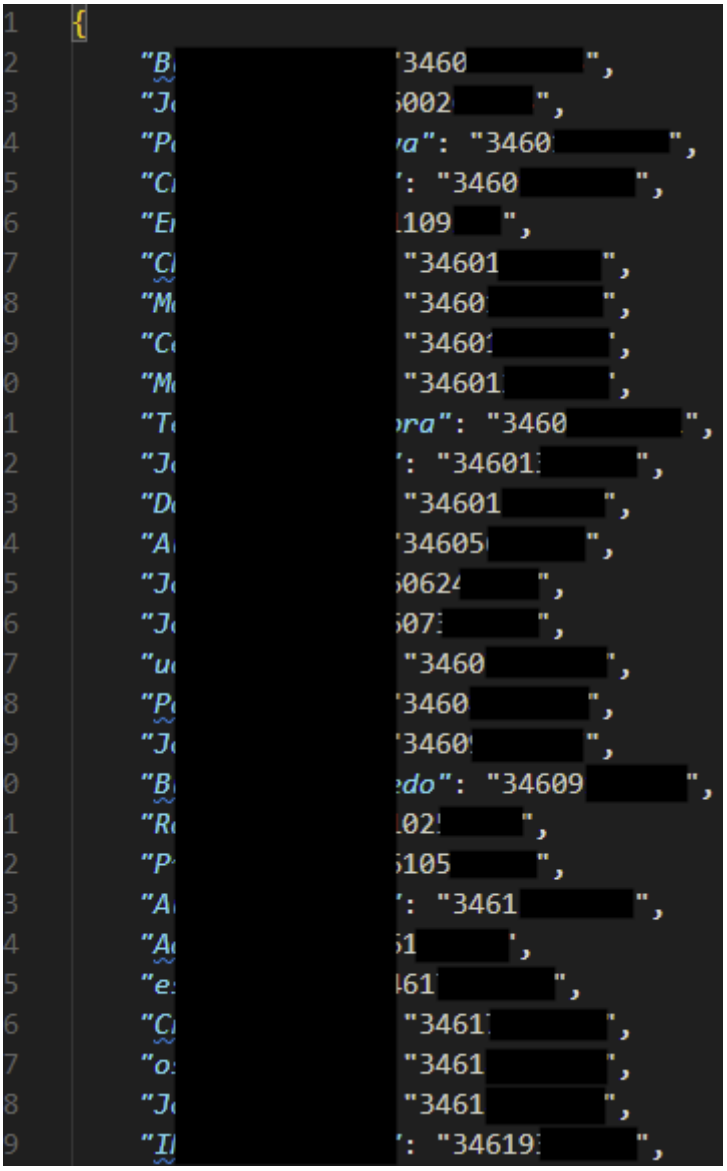


Figura 5.10: Fichero contactos.json

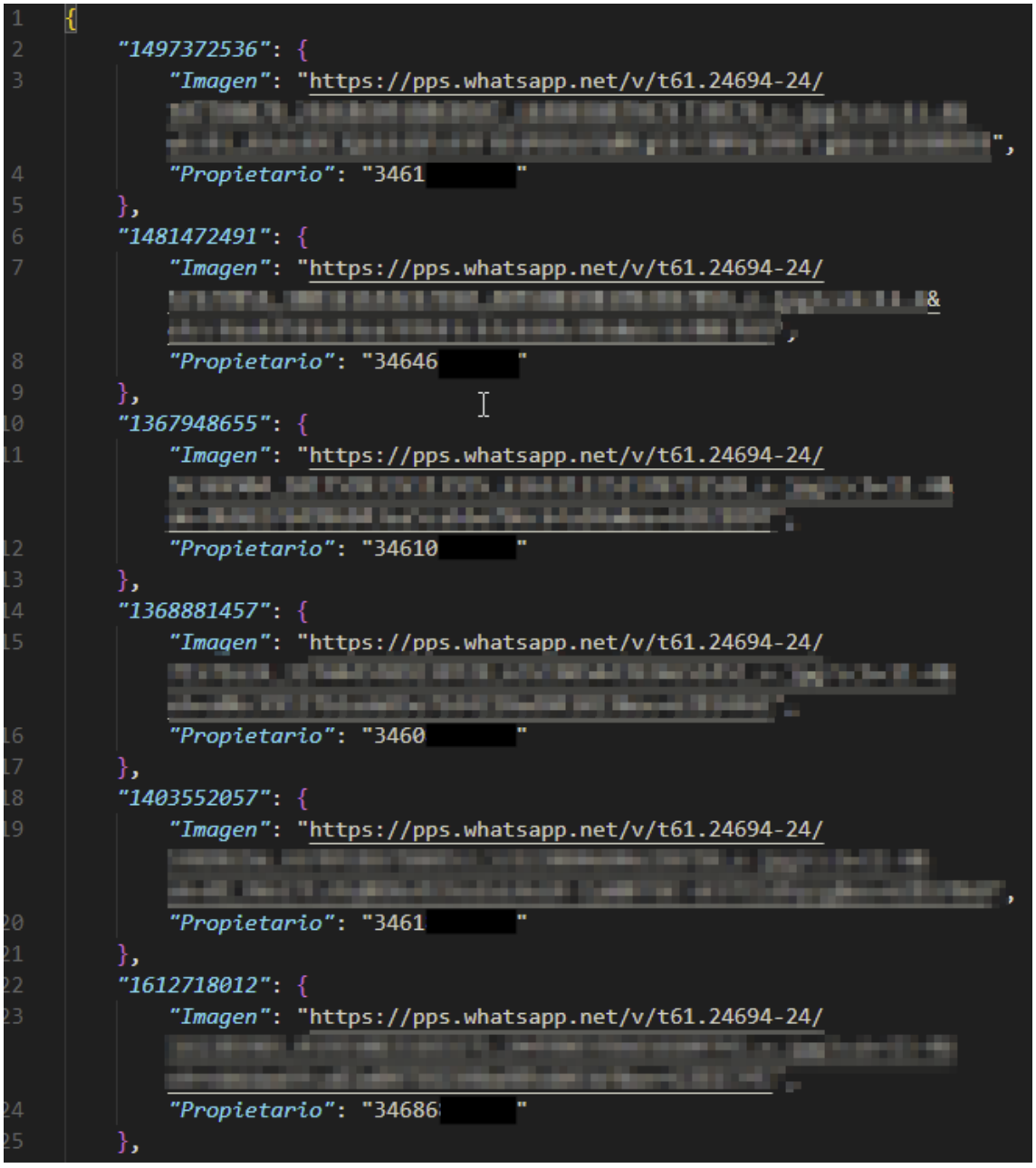


Figura 5.11: Fichero fotos_grupos.json

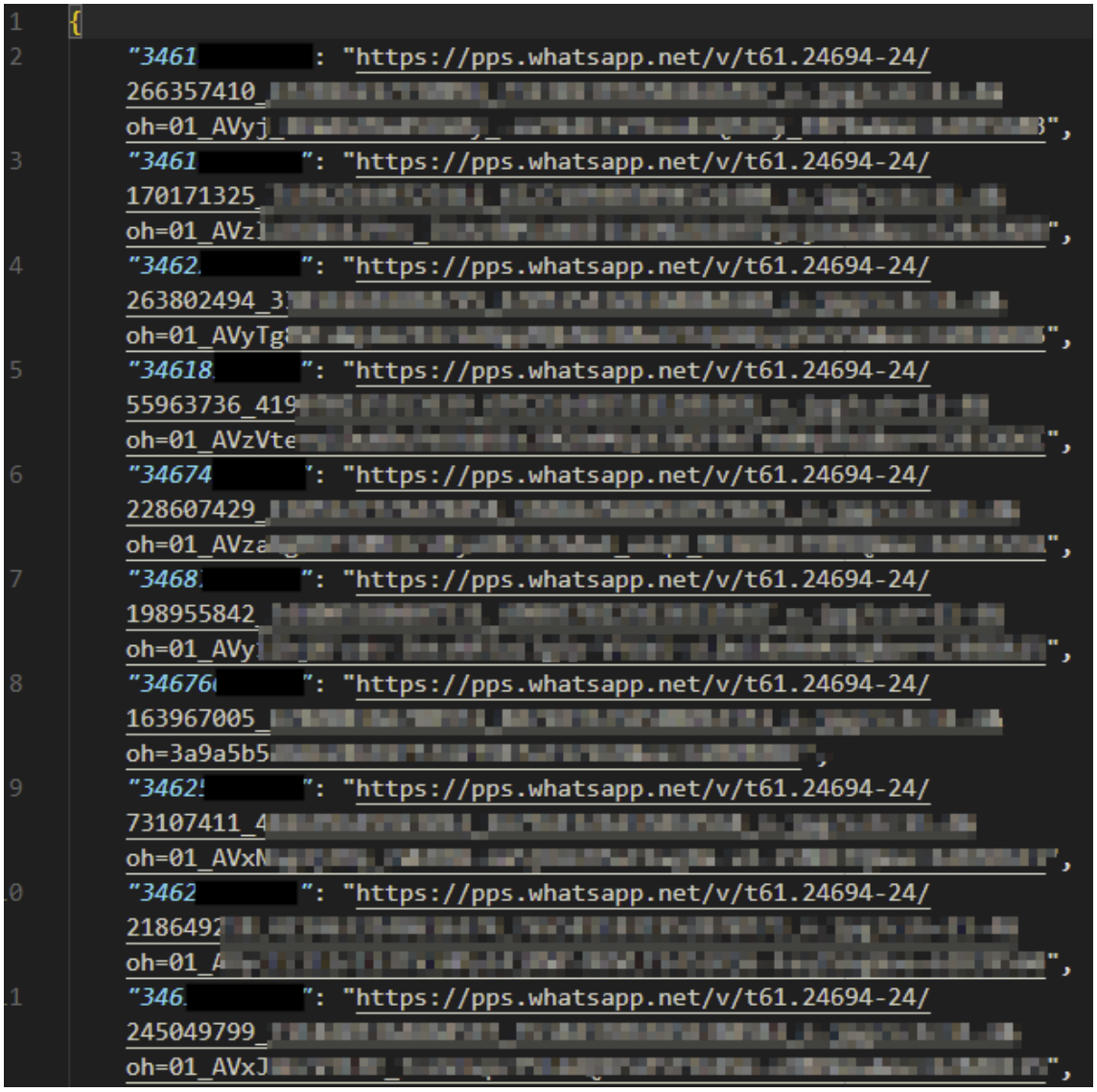


Figura 5.12: Fichero fotos_perfil.json

Figura 5.13: Fichero grupos.json

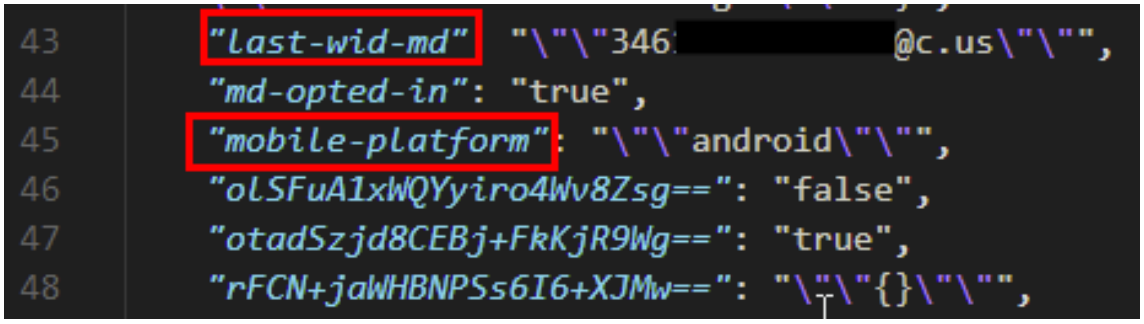


Figura 5.14: Fichero local_storage.json: Número de teléfono del dispositivo ('last-wid-md') y sistema operativo del terminal ('mobile-platform')

Figura 5.15: Fichero local_storage.json: Última hora de inicio de sesión ('whatsapp-mutex')

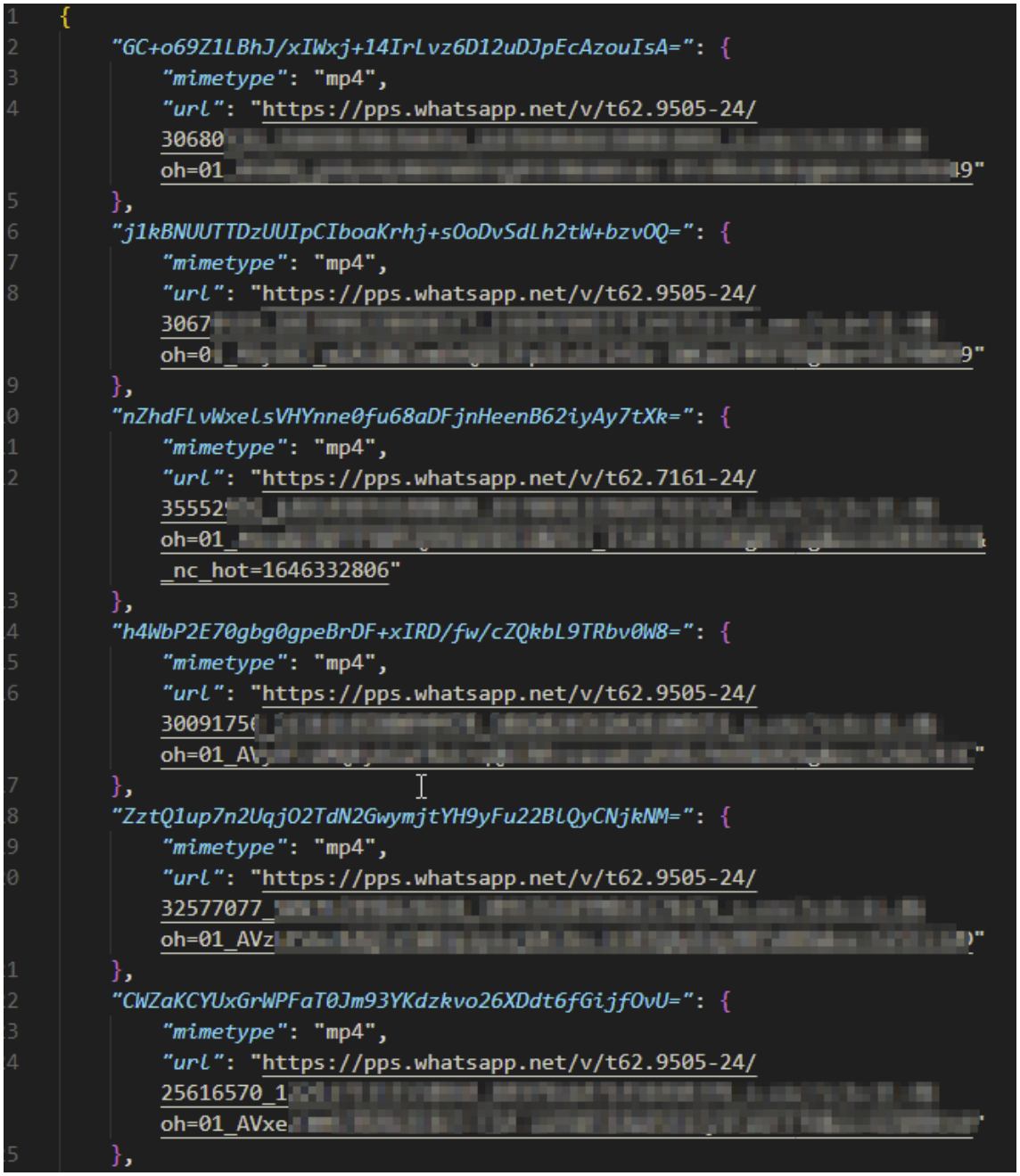


Figura 5.16: Fichero videos.json

Por tanto, con este script, se ha podido recuperar de la base de datos Leveldb:

- Audios
- Contactos
- Fotos de los grupos
- Fotos del perfil de los usuarios

- Grupos a los que pertenece
- Imágenes
- Vídeos
- ‘Local Storage’ de Chrome

5.3. Desencriptado de ficheros multimedia

En esta sección se proporciona un método para desencriptar los ficheros '.enc'. Estos ficheros se obtienen a través de las 'url' de los archivos generados por el script *whatsapp_parser_tipo_2.py*:

- audios.json
- videos.json
- imagenes.json

En esta sección se ha empleado el entorno de pruebas 3.

Para el desencriptado se ha utilizado, del repositorio de sigalor [15][25], la parte del *backend*. El código se encuentra en la información anexa con el nombre 'd3crypt.zip'.

A continuación se muestran 2 ejemplos de desencriptado:

- Audio (figura 5.17)
- Vídeo (figura 5.18)

backend > whats-enc.py > ...

```
1   from whatsapp import HKDF, AESUnpad;
2   import base64;
3   import urllib2;
4   from Crypto.Cipher import AES;
5
6   # Campos a rellenar
7   mediaurl = "https://pps.whatsapp.net/v/t62.7117-24/18980061_1643442439367
8   mediakey = "+jqtCehykdSPRaKkpraucn1+j1G21m1VQoqW72fT9jM="
9   salt = "WhatsApp Audio Keys"
10
11  nombre = 'audio.ogg'
12  # salt = "WhatsApp Image Keys"
13  # salt = "WhatsApp Video Keys"
14
15  ############################ No tocar nada
16  mediaKeyExpanded = HKDF(base64.b64decode(mediakey),112,salt)
17  iv = mediaKeyExpanded[:16]
18  cipherKey = mediaKeyExpanded[16:48]
19  macKey = mediaKeyExpanded[48:80]
20
21  mediaData = urllib2.urlopen(mediaurl).read()
22  file = mediaData[:-10]
23  mac = mediaData[-10:]
```

Figura 5.17: Desencriptado de un audio

backend > whats-enc.py > ...

```python
from whatsapp import HKDF, AESUnpad;
import base64;
import urllib2;
from Crypto.Cipher import AES;

# Campos a rellenar
mediaurl = "https://pps.whatsapp.net/v/t62.7161-24/12089503_1064057344540667_5
mediakey = "nb7y8RtpZZ35W1pKjrkLmr7IUIqNgorc7hm9/4iS21A="
salt = "WhatsApp Video Keys"

nombre = 'video1.mp4'
# salt = "WhatsApp Image Keys"
# salt = "WhatsApp Audio Keys"

############################## No tocar nada
mediaKeyExpanded = HKDF(base64.b64decode(mediakey),112,salt)
iv = mediaKeyExpanded[:16]
cipherKey = mediaKeyExpanded[16:48]
macKey = mediaKeyExpanded[48:80]

mediaData = urllib2.urlopen(mediaurl).read()
file = mediaData[:-10]
mac = mediaData[-10:]


decryptor = AES.new(cipherKey, AES.MODE_CBC, iv)
imgdata=AESUnpad(decryptor.decrypt(file))
```

Figura 5.18: Desencriptado de un vídeo

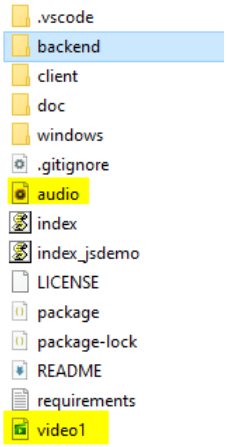


Figura 5.19: Ficheros desencriptados

Finalmente, como se puede ver en la figura 5.19 **se han podido desencriptar los ficheros multimedia** obtenidos con el script.

5.4. Conclusión

En este segundo análisis se ha identificado que todo lo que puede visualizarse a través de 'DevTools' en 'IndexedDB', se podrá encontrar en la base de datos 'LevelDB'.

Se han podido localizar mensajes cifrados en el extracto realizado de la base de datos.

A su vez, se ha proporcionado un script capaz de obtener de la base de datos los contactos, audios, fotos de perfil y de los grupos, imágenes, vídeos y el 'Local Storage' del navegador.

Finalmente, se han desencriptado ficheros multimedia, cuyos enlaces de descarga se encontraban en los ficheros generados por el script.

Capítulo 6

Conclusión

El objetivo de este estudio ha sido localizar y aportar información sobre el contenido de los artefactos que genera WhatsApp Web.

A lo largo de la investigación, se han generado dos análisis en entornos diferentes. Uno que se encontraba desactualizado y con pocos datos y otro más realista y actualizado.

En primer lugar, se ha trabajado con una herramienta que parecía no funcionar. Tras el análisis de su código fuente se ha comprendido el funcionamiento y se ha enfocado el estudio en el fichero del cual toma los datos. Se entendió entonces que lo que ocurría es que la herramienta se encontraba desactualizada. Tras esto se ha planteado una cuestión sobre si era posible obtener la base de datos completa, y no solo una parte, tal y como hacía esta herramienta. Así pues, se ha buscado la forma de extraer los datos y se ha desarrollado un script para automatizar la tarea. Tras todo esto, se ha verificado que la versión que se ha utilizando no estaba actualizada. Fue entonces cuando ha surgido el segundo análisis.

En la segunda parte de la investigación, se han identificado elementos en el navegador que estaban presentes en la base de datos y se han podido extraer. Además, se ha desarrollado otro script capaz de obtener más datos que el primero. Finalmente, el análisis ha terminado con el desencriptado de ficheros multimedia.

6.1. Resolución de preguntas de la investigación

- **¿Dónde se localizan los artefactos que genera WhatsApp Web?**

  Se pueden localizar en la siguiente ruta:

  C:\Users\windows\AppData\Local\Google\Chrome\User Data\Default\IndexedDB\https_web.whatsapp.com_0.indexeddb.leveldb

- **¿Qué tipo de artefactos son?**

  Se trata de una base de datos LevelDB.

- **¿Se pueden obtener mensajes?**

  En el primer análisis se han obtenido marcas de tiempo sobre el envío y recepción de mensajes, así como el identificador único del mensaje, el chat al que pertenece y quién lo envía.

  En el segundo análisis se han encontrado mensajes cifrados en la base de datos LevelDB.

- **En caso de que se guarden mensajes:**

  - **¿Cada mensaje queda identificado con el modelo del dispositivo?**
    En el primer estudio se puede averiguar la marca, el modelo, número de compilación o la versión de Android del dispositivo móvil. Queda registrado quién envía y quién recibe cada mensaje, así que la respuesta es sí. Podría quedar identificado el modelo del dispositivo con cada mensaje.

    En el segundo estudio no se puede establecer tal relación de forma directa. Requiere un mayor estudio.
  - **¿Se puede identificar cuándo un mensaje se envía desde el teléfono móvil, desde el cliente Web o a través de la aplicación de Escritorio?**
    Sí, se puede identificar ya que cuando un mensaje se envía desde el cliente Web, el identificador único de ese mensaje (campo '*key_id*') comienza por 3EB0.
  - **¿Se puede ver el contenido en texto plano de los mensajes?**
    En ninguno de los dos estudios se ha podido encontrar en texto plano el contenido de los mensajes.

- **¿Se pueden obtener archivos multimedia?**

  En el primer análisis, no se han podido obtener este tipo de archivos.

  No obstante, en el segundo se han podido obtener los ficheros multimedia cifrados. También se ha aportado un mecanismo de descifrado.

- **¿Se pueden obtener los contactos?**

  Sí. Es posible obtener contactos tanto en el primer como en el segundo estudio realizados.

  También se ha podido obtener el número de teléfono que está conectado, así como el sistema operativo del dispositivo móvil.

6.2. Trabajo futuro

Como líneas de trabajo futuras se proponen las siguientes:

- Análisis de la memoria RAM con herramientas como Windows Memory Extractor [21] para estudiar los artefactos que se podrían conseguir.
- Análisis de WhatsApp Desktop.
- Análisis de WhatsApp Business y las diferencias en sus bases de datos.
- Análisis de los ficheros en caché. Los cuales suponen un mayor espacio que el de la base de datos LevelDB.
- Mejora de los scripts proporcionados:
  - Mejora de las expresiones regulares utilizadas
  - Integración del desencriptado
- Análisis de otras redes sociales empleando esta metodología de análisis.

Bibliografía

[1] Statista. *Most popular global mobile messenger apps as of January 2022, based on number of monthly active users (in millions)*. URL: https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/.

[2] WhatsApp. *Seguridad en WhatsApp*. URL: https://www.whatsapp.com/security.

[3] Statista. *Market share held by the leading computer (desktop/tablet/console) operating systems worldwide from January 2012 to December 2021*. URL: https://www.statista.com/statistics/268237/global-market-share-held-by-operating-systems-since-2009/#:~:text=Microsoft's%20Windows%20is%20the%20most,OS%20market%20in%20December%202021.

[4] Statista. *Global market share held by leading internet browsers from January 2012 to December 2021*. URL: https://www.statista.com/statistics/268254/market-share-of-internet-browsers-worldwide-since-2009/.

[5] Xataka. *Chromium: qué es, en qué se diferencia de Chrome y cómo descargar*. URL: https://www.xataka.com/basics/chromium-que-que-se-diferencia-chrome-como-descargar.

[6] Catalin Cimpanu. *All the Chromium-based browsers*. URL: https://www.zdnet.com/pictures/all-the-chromium-based-browsers/.

[7] Sabarinath. *7 Best Chromium Based Browsers With Extra Features*. URL: https://techlog360.com/best-chromium-based-browsers/.

[8] Nicolas Villacis Vukadinovic. "Whatsapp forensics: Locating artifacts in web and desktop clients". Tesis doct. Purdue University Graduate School, 2019.

[9] Carlos Pintos Teigeiro. "Análisis del modelo de datos de la red social WhatsApp y sus aplicaciones al peritaje". Tesis doct. Universidad Europea con IBM, 2020.

[10] Igor Mikhailov. *WhatsApp in Plain Sight: Where and How You Can Collect Forensic Artifacts*. URL: https://blog.group-ib.com/whatsapp_forensic_artifacts.

[11] Buenaventura Salcedo Santos-Olmo. *Curso de Ciberseguridad Forense: Dispositivos Móviles, Enigma University*. 2022.

[12] Bery Actoriano e Imam Riadi. "Forensic investigation on Whatsapp web using framework integrated digital forensic investigation framework version 2". En: *International Journal of Cyber-Security and Digital Forensics (IJCSDF)* 7 (2018), págs. 410-419.

[13] Furkan Paligu y Cihan Varol. "Browser Forensic Investigations of WhatsApp Web Utilizing IndexedDB Persistent Storage". En: *Future Internet* 12.11 (2020), pág. 184.

[14] Nagendar Rao Koppolu. "A Deep-dive Analysis on WhatsApp Artifacts and their Relevance in Crime Investigation". En: *International Research Journal of Engineering and Technology* 08.06 (2021), pág. 22.

[15] sigalor. *WhatsApp Web reverse engineered*. URL: `https://github.com/sigalor/whatsapp-web-reveng`.

[16] @trascendentale. *Backing up WhatsApp data through the multi-device web client*. URL: `https://mazzo.li/posts/whatsapp-backup.html`.

[17] Google. *IndexedDB*. URL: `https://chromium.googlesource.com/chromium/src/+/master/content/browser/indexed_db/docs/README.md`.

[18] CCL SOLUTIONS GROUP. *IndexedDB on Chromium*. URL: `https://www.cclsolutionsgroup.com/post/indexeddb-on-chromium`.

[19] CCL SOLUTIONS GROUP. *Hang on! That's not SQLite! Chrome, Electron and LevelDB*. URL: `https://www.cclsolutionsgroup.com/post/hang-on-thats-not-sqlite-chrome-electron-and-leveldb`.

[20] furkanpaligu. *BrowSwEx*. URL: `https://github.com/furkanpaligu/BrowSwEx`.

[21] Pedro Fernández-Álvarez. *Recovering data from the memory of Telegram Desktop (and other IM applications)*. URL: `https://reversea.me/index.php/recovering-data-from-the-memory-of-telegram-desktop-and-other-im-applications/`.

[22] Daniel Avilla. *Avilla Forensics 3.0*. URL: `https://github.com/AvillaDaniel/AvillaForensics`.

[23] CCL SOLUTIONS GROUP. *ccl chrome indexeddb*. URL: `https://github.com/obsidianforensics/ccl_chrome_indexeddb`.

[24] andro4all. *Ya no necesitas tener el móvil encendido para poder usar WhatsApp en tu PC*. URL: `https://andro4all.com/whatsapp/ya-no-necesitas-tener-el-movil-encendido-ni-conectado-a-internet-para-usar-whatsapp-en-tu-pc`.

[25] Robert David Graham. *Whatsapp decryption of .enc files*. URL: `https://github.com/robertdavidgraham/whats-enc/tree/master/backend`.

Parte II

Anexos

Apéndice A

Extracción de las bases de datos de WhatsApp del dispositivo móvil

Para la extracción es necesario, principalmente:

- Un APK de WhatsApp de la versión 4.3.1 (whatsapp_431.apk)
- Android backup extractor (abe.jar) [1]

Con el dispositivo conectado, el modo de depuración activado y permitiendo las instalaciones desde USB sin verificar, se ejecutan los siguientes comandos:

```
adb shell pm list packages -3 | grep com.whatsapp
adb shell pm path com.whatsapp
adb shell am force-stop com.whatsapp
adb shell am kill com.whatsapp
adb pull -a /data/app/com.whatsapp-1/base.apk
adb shell pm uninstall -k com.whatsapp
adb install -r -d whatsapp_431.apk
adb reboot
adb install -r -d whatsapp_431.apk
adb shell monkey -p "com.whatsapp -c android.intent.category.LAUNCHER 1"
adb shell "input keyevent 3"
adb backup -f whatsapp.ab com.whatsapp
adb install -r -d base.apk
java -jar abe.jar unpack whatsapp.ab whatsapp.tar
```

Tras la ejecución de estos se obtiene el fichero '*whatsapp.tar*'. Al descomprimir este fichero, se encuentra la ruta de ficheros de la figura A.1.

[1] `https://github.com/nelenkov/android-backup-extractor`

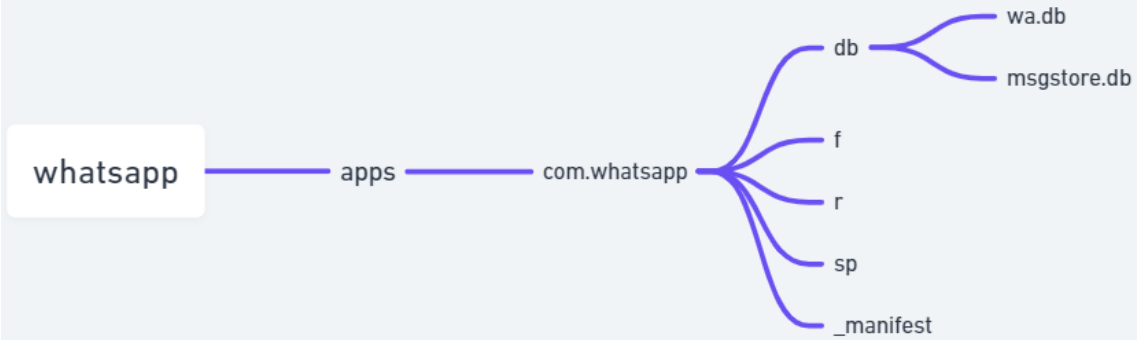


Figura A.1: Ficheros obtenidos tras descomprimir whatsapp.tar

```
                                    $ adb shell pm list packages -3 | grep com.whatsapp
package:com.whatsapp
                                    $ adb shell pm path com.whatsapp
package:/data/app/com.whatsapp-1/base.apk
                                    $ adb shell am force-stop com.whatsapp
                                    $ adb shell am kill com.whatsapp
                                    $ adb pull -a /data/app/com.whatsapp-1/base.apk
/data/app/com.whatsapp-1/base.apk: 1 file pul... skipped. 4.0 MB/s (39055777 bytes in 9.341s)
                                    $ adb shell pm uninstall -k com.whatsapp
Success
                                    $ adb install -r -d whatsapp_431.apk
Performing Push Install
whatsapp_431.apk: 1 file pushed, 0 skipped. 6.7 MB/s (18329558 bytes in 2.602s)
        pkg: /data/local/tmp/whatsapp_431.apk
Success
                                    $ adb shell monkey -p "com.whatsapp -c android.intent.cat
egory.LAUNCHER 1"
Events injected: 1
## Network stats: elapsed time=173ms (0ms mobile, 0ms wifi, 173ms not connected)
                                    $ adb shell "input keyevent 3"
                                    $ adb backup -f whatsapp.ab com.whatsapp
WARNING: adb backup is deprecated and may be removed in a future release
Now unlock your device and confirm the backup operation...
                                    $ adb install -r -d base.apk
Performing Push Install
base.apk: 1 file pushed, 0 skipped. 8.8 MB/s (39055777 bytes in 4.230s)
        pkg: /data/local/tmp/base.apk
Success
                                    $ java -jar abe.jar unpack whatsapp.ab whatsapp.tar
```

Figura A.2: Extracción de las BBDD de WhatsApp de un Dispositivo Móvil

En resumen, lo que se realiza a través de los comandos anteriores es:

1. Localizar la ubicación y obtener el APK original
2. Desinstalar la versión de WhatsApp sin borrar los datos (opción -k)
3. Instalar un APK WhatsApp que permite hacer backups (versión 4.3.1)
4. Lanzar la app
5. Realizar un backup de WhatsApp
6. Restaurar la versión original de WhatsApp
7. Descomprimir el backup.

Destacar que, para este proceso, **no es necesario root** en el dispositivo móvil.

No se abordará con más detalle este proceso debido a que queda fuera del alcance de este estudio. El método empleado se ha obtenido del ‘Curso de Ciberseguridad Forense: Dispositivos Móviles‘ [11].