Available online at www.HighTechJournal.org

HighTech and Innovation Journal

ISSN: 2723-9535

Vol. 3, No. 2, June, 2022

Forensic Analysis of WhatsApp SQLite Databases on the Unrooted Android Phones

Hasan Fayyad-Kazan [1]*, Sondos Kassem-Moussa [2], Hussin J. Hejase [3], Ale J. Hejase [4]

[1] *Department of Information Technology, Al Maaref University, Beirut, Lebanon.*

[2] *EDST, Lebanese University, Beirut, Lebanon.*

[3] *Faculty of Business Administration, Al Maaref University, Beirut, Lebanon.*

[4] *AKSOB, Lebanese American University, Beirut, Lebanon.*

Received 22 December 2021; Revised 07 February 2022; Accepted 17 February 2022; Available online 24 February 2022

Abstract

WhatsApp is the most popular instant messaging mobile application all over the world. Originally designed for simple and fast communication, however, its privacy features, such as end-to-end encryption, eased private and unobserved communication for criminals aiming to commit illegal acts. In this paper, a forensic analysis of the artefacts left by the encrypted WhatsApp SQLite databases on unrooted Android devices is presented. In order to provide a complete interpretation of the artefacts, a set of controlled experiments to generate these artefacts were performed. Once generated, their storage location and database structure on the device were identified. Since the data is stored in an encrypted SQLite database, its decryption is first discussed. Then, the methods of analyzing the artefacts are revealed, aiming to understand how they can be correlated to cover all the possible evidence. In the results obtained, it is shown how to reconstruct the list of contacts, the history of exchanged textual and non-textual messages, as well as the details of their contents. Furthermore, this paper shows how to determine the properties of both the broadcast and the group communications in which the user has been involved, as well as how to reconstruct the logs of the voice and video calls.

Keywords: Android; Instant Messaging; Mobile Forensics; SQLite Databases; WhatsApp Messenger.

1. Introduction

Over a decade ago, regular mobile phones offered the Short Message Service (SMS) as an alternative to the instant messaging (IM) that existed on the internet at that time. This service failed to offer the convenience of real-time texting, which is available in IM. Nevertheless, the potentially new-born smartphones in 2007 opened the doors for real-time communication capability in mobile phones through instant messaging applications such as WhatsApp, which is the most popular of these applications almost globally with 2 billion users, as shown by Statista (2021) in Figure 1 [1]. The huge number of WhatsApp users means remarkably big data getting transferred through the app. With that in mind, the way WhatsApp handles this data is a case to investigate. The messages exchanged on early versions of WhatsApp were kept in SQLite local databases on the devices. The database file was, in fact, not encrypted, which meant that WhatsApp chat records were vulnerable to intruders, putting users' data at risk. On the other hand, the more recent versions of the application have seriously reconsidered the database security and have encrypted the databases following the custom Advanced Encryption Standard (AES).

* Corresponding author: hafayyad@gmail.com

http://dx.doi.org/10.28991/HIJ-2022-03-02-06

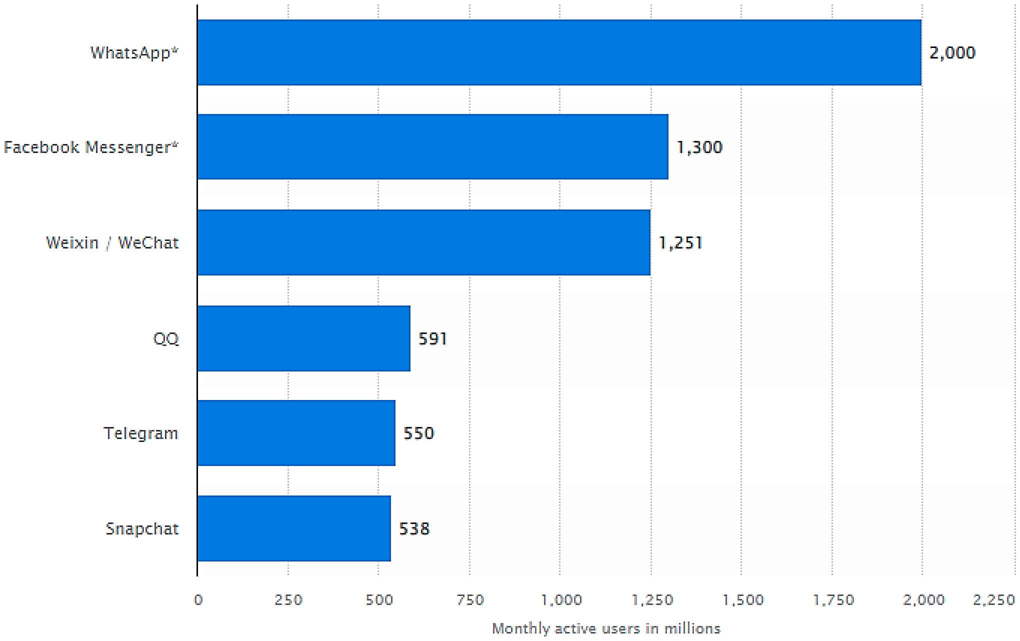


Figure 1. WhatsApp monthly active users compared to other IM apps

Technology itself allows for both good and bad behavior. Due to the privacy and, in most cases, encryption that come with the IM apps, they are becoming a common means for intruders (criminals, hackers, virtual thieves, etc.) to get involved in illegal actions such as drug dealing, hate speech, child pornography, terrorist acts and more. The importance and the need to retrieve electronic evidence from criminals' devices thus becomes a part of any forensic process [2]. Databases, such as WhatsApp's, are critical in this process, not only because average criminals lack the technical knowledge to access or modify them, but also because this is where the encryption challenge is presented.

In this paper, the authors address the forensic analysis of the artefacts left by the encrypted WhatsApp SQLite databases on unrooted Android devices. The study is limited to the Android OS because, according to Statista [3], the most sold smartphones to end users are those with the Android operating system. This work covered the analysis of artefacts such as the call logs and the status, as well as the WhatsApp desktop application. In addition, it shows how to decrypt the SQLite databases as one works on the unrooted phones. Furthermore, since it's possible that the SQLite database may be corrupted, the paper discusses how to deal with it.

This paper is divided into five sections, starting with the introduction. Section two presents related literature, followed by section three, exposing the methods and materials. Section four presents a discussion, and finally, the last section concludes the paper and offers recommendations.

2. Related Works

The forensic analysis of smartphones takes a lot of attention in the literature. Most of the papers and books on this subject focus on Android and iOS. Skulkin & Tindall [4] and Epifani & Stirparo [5] explained very deeply the guide to use while working on these two OSs, giving key strategies and techniques to extract and analyze forensic artefacts from mobile phones. In this study, the authors benefited from this work in order to extract and analyze the data generated by the WhatsApp messenger. Indeed, one of the most studied domains in mobile forensics is the applications installed on the device, mainly the IM apps. The importance and the popularity of the IM apps is the reason behind the increased number of works published on this topic. For example, Zhang et al. [6] focused on the forensic analysis of WeChat on Android phones, Anglano et al. [7] worked on the analysis of Chat-Secure, and Walnycky et al. discussed [8] the analysis of 20 popular IM messengers on Android phones, while Ovens & Morison [9] analyzed an IM application (Kik messenger) on iOS. Also, Anglano et al. [10] studied the telegram messenger, providing a general methodology for the analysis of android applications. The same thing was done by Zhang et al. [11] but on four popular IM apps rather than only Telegram. Rathi et al. [12] discussed in-depth the analysis of the encryption of various IM apps. Azfar et al. [13] proposed a taxonomy outlining the forensic importance of the evidence generated by the IM apps. All the aforementioned papers considered the analysis based primarily on the artefacts presented on the device and the encrypted databases generated by these apps.

Moreover, several studies focus on the analysis of the WhatsApp messenger on Android phones. Thakur [14] and Mahajan et al. [15] focused on the analysis of just a part of the artefacts left by WhatsApp (just the chat database). Whereas, Anglano [16] discussed the analysis of WhatsApp, focusing on the contacts and chat databases on rooted phones without aiming to study their encryption. In the current study, the authors explained the decryption of the encrypted databases while working on unrooted Android phones. In addition, they covered new features that became available later on, as well as the new update of the values stored under the existing and new fields.

3. Methods and Materials

A set of experiments were performed in order to accomplish this study. Each experiment deals with different interaction scenarios between users. The forensic data generated by WhatsApp is saved to the internal memory of the device, and some of it is located in inaccessible areas for the normal user, unless when using advanced commercial forensic tools for the purpose of making them accessible. As well, this data is stored in encrypted SQLite databases, which also need suitable commercial tools to be extracted [17, 18]. Unfortunately, these tools are expensive, so the work was carried out using open-source tools and a powerful programming language such as Python to achieve the goals. In addition, unrooted Android devices were used for two reasons: first, the failure of the rooting process during the investigation will require re-installing the OS, which leads to overwriting the data preserved on the device, and second, the rooting process is becoming more difficult with the new versions of Android (7.0 and higher).

3.1. Required tools used

1. Android Smartphones:
   - Sony Xperia Z2 - Android 6.0.1
   - Sony Xperia L - Android 4.2.2
   - Huawei y7 prime - Android 8.0.0
2. WhatsApp version:
   - Version 2.19.53
   - Web WhatsApp PC application
3. Forensics workstation (PC):
   - Toshiba Satellite C850-B907
4. WhatsApp encryption key extractor:
   - Python script written for this purpose.
5. WhatsApp extract:
   - Open-source tools to decode the encrypted databases
6. SQLite DB viewer:
   - SQLite DB browser to parse the data saved in the SQLite databases.

3.2. Experimental Setup

From the Google Play Store, WhatsApp was installed on the three devices (Android Smartphones mentioned in the previous paragraph). The application is stored on the internal memory in the directory "whatsApp.com". When a person uses the application, every message sent and received is stored in an encrypted SQLite database named msgstore.db, and the contacts involved are stored in another encrypted SQLite database named wa.db. This encryption should be decrypted in order to extract data from these SQLite databases; otherwise, it is not possible to open and read their contents. The strength of this encryption makes it unbreakable using brute force techniques since it uses 256-bit key AES encryption. The only way to decrypt it is to use the private key, which is stored uniquely on each device and cannot be accessed without root privilege. In this work, a method to get this key without a root is used. For this purpose, the USB debugging mode was enabled from the settings menu on the mobile devices. This is done by tapping seven times on "build number" in "about phone". In the new option named "developer option", "developer mode" is enabled as well.

3.3. Sets of experiments

1. Experiments concerning contacts:

The goal from these experiments is to determine the list of the user’s contacts as well as the operation done on it by the user. These experiments are listed in Table 1.

Table 1. User contacts experiments. User1 and User2 are the WhatsApp users involved in the experiments

Operation	Steps
Add contacts	User 1 adds user 2
Remove contacts	User 1 deletes user 2
Block contacts	1- User 1 blocks user 2
	2- User 1 unblocks user 2

2. Experiments concerning the private chat communication between the user and contacts:

The goal is to reconstruct the history of messages exchanged as well as the contents of the textual and non-textual messages between the user and each contact. See Table 2.

Table 2. Experiments concerning all the types of messages exchanged privately

Operation	Steps
Textual messages exchange	1- User 1 and user 2 exchange messages
	2- User 1 and user 2 delete messages
Textual messages forward	1- User 1 forwards to user 2 from user 3
	2- User 2 forwards to user 3 from user 1
Non-textual messages exchange	1- User 1 sends a picture to user 2
	2- User 1 sends a video to user 2
	3- User 1 sends an audio to user 2
	4- User 1 sends a contact to user 2
	5- User 1 sends a geolocation to user 2

3. Experiments concerning the messages state (Table 3):

The goal is to determine if the message has reached the server and if this message has been delivered to the recipient.

Table 3. Message state experiments

Operation	Steps
Sending message, receiver offline	1- User 1 sends a message to offline user 2
	2- User 2 replies when online
Sending message, sender offline	1- User 1 is offline
	2- User 1 sends a message
	3- User 1 goes online

4. Experiments concerning the broadcast and group messages:

The goal is to determine the users involved in a broadcast message and to reconstruct the chronology of a group chat as well as the events that happened within. See Table 4.

Table 4. Broadcast and group message experiments

Operation	Steps
Broadcast messages	1- User 1 sends a broadcast message to more than two saved contacts
	2- User 1 sends a broadcast to unsaved contacts
Group messages	1- User 1 creates a group and adds user 2 and user 3
	2- User 1 adds user 4
	3- User 1 removes user 2, then user 3 and user 4

5. Experiments concerning voice and video calls (Table 5)

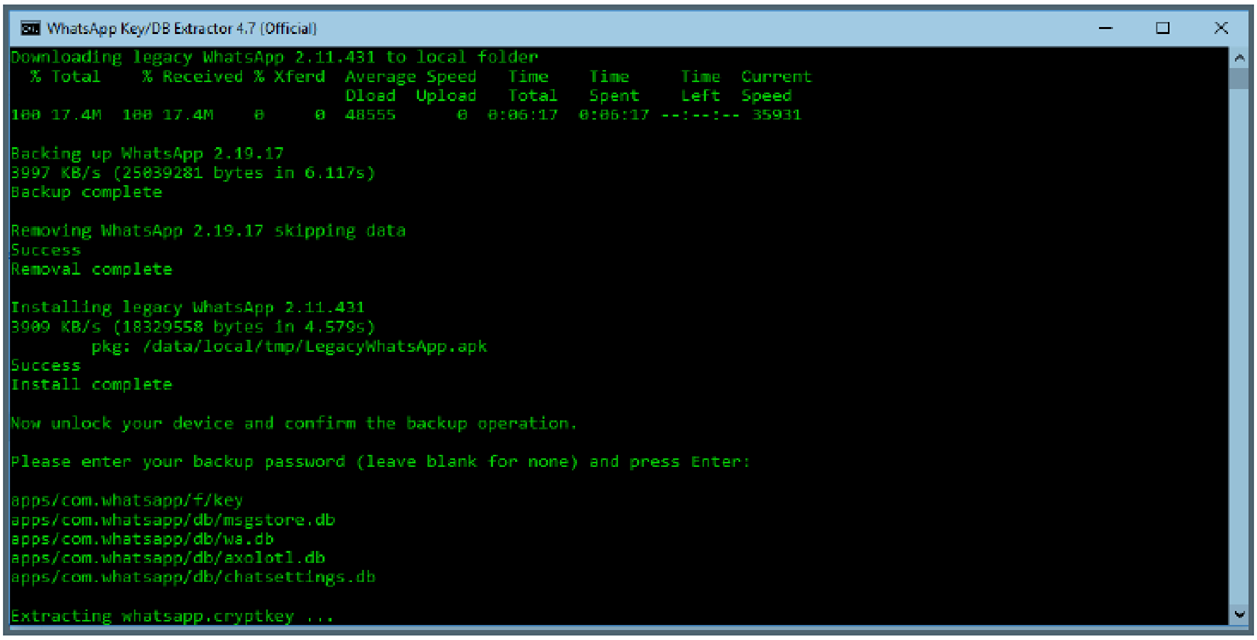
The goal is to reconstruct the chronology of incoming and outgoing calls of the user.

Table 5. WhatsApp voice and video calls experiments

Operation	Steps
Performed voice/ video call	1- User 1 calls user 2
	2- User 2 answers
	3- User 1 hangs up
	4- Repeat 1-3 as user 2 is the sender
Missed voice/ video call	1- User 1 calls user 2
	2- User 2 does not answer
Refused voice/ video call	1- User 1 calls user 2
	2- User 2 terminates the call without answering

3.4. SQLite Databases Decryption

A python script DB key extractor was used in order to extract the cipher key, which is stored in the internal memory of the device. This is an open-source project, so some functions in the script were re-written in order to suit the work. One of the most important variations is that the script didn't need an internet connection. This is very important because it allows you to disconnect the phone from all networks, thus preventing any control that might happen to the phone remotely. The principle of this method is simple, consisting of downgrading WhatsApp to versions prior to 2.18 (Figure 2) where the databases' encryption was not supported, and then extracting the latest unencrypted WhatsApp messages database, msgstore.db, and contacts database, wa.db, as well as the cipher key that can be used to decrypt the encrypted database (Figure 3). This will create a folder on the PC named "extracted" that contains these DBs and the key (Figure 4).



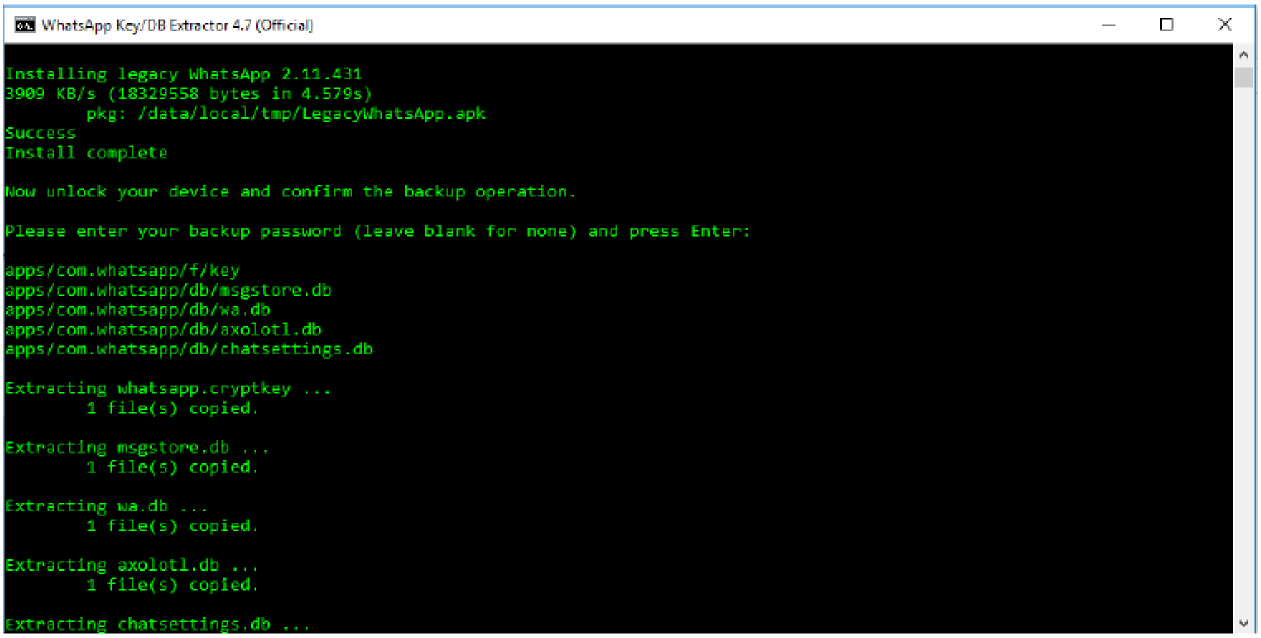
```
Downloading legacy WhatsApp 2.11.431 to local folder
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 17.4M  100 17.4M    0     0  48555      0  0:06:17  0:06:17 --:--:-- 35931

Backing up WhatsApp 2.19.17
3997 KB/s (25039281 bytes in 6.117s)
Backup complete

Removing WhatsApp 2.19.17 skipping data
Success
Removal complete

Installing legacy WhatsApp 2.11.431
3909 KB/s (18329558 bytes in 4.579s)
        pkg: /data/local/tmp/LegacyWhatsApp.apk
Success
Install complete

Now unlock your device and confirm the backup operation.

Please enter your backup password (leave blank for none) and press Enter:

apps/com.whatsapp/f/key
apps/com.whatsapp/db/msgstore.db
apps/com.whatsapp/db/wa.db
apps/com.whatsapp/db/axolotl.db
apps/com.whatsapp/db/chatsettings.db

Extracting whatsapp.cryptkey ...
```

WhattsAppKey/DBExtractor 4.7 (Official)

```
Downloading legacy WhatsApp 2.11.431 to locate folder
     % Total       % Received  % xferd  Average Speed    Time    Time     Time    Current
                                        Dload   Uoload   Total   Spent    Left    Speed
100  17.4M   100  17.4M         0       48555   0        0:06:17 0:06:17  --:--:--  35931

Backing Up WhatsApp 2.19.17  skipping data
3997 KB/s (25039281 bytes in 6.117 s)
Backup complete

Removing WhatsApp 2.19.17 skipping data
Success
Removal complete

Installing legacy WhatsApp 2.11.431
3909 KB/s (18320558 bytes in 4.579s)
```

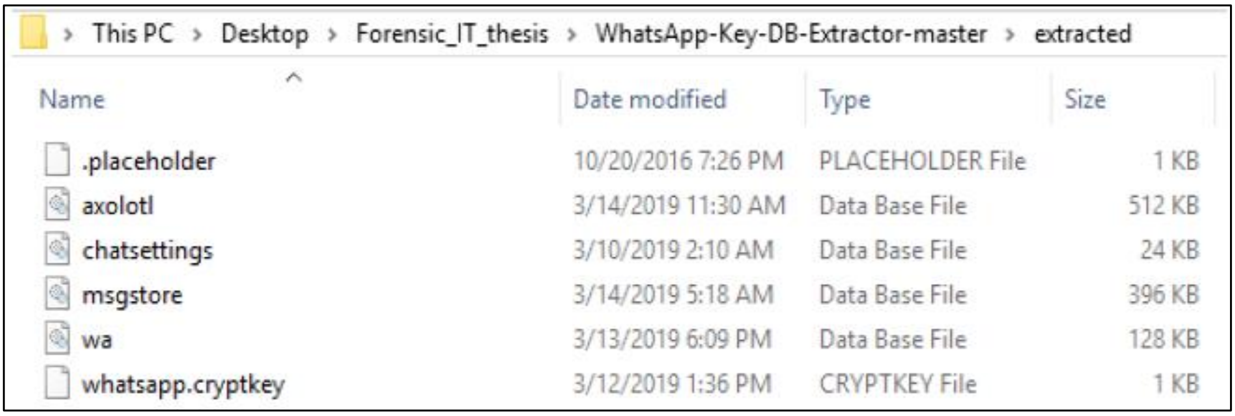
```
        pkg: data/local/tmp/legacyWhattsApp.apk
Success
Install complete

Now unlock your device and confirm the backup operation.

please enter your backup password (leave blank for none) and press enter:

apps/com.whatsapp/apps/f/key
apps/com.whattsapp/db/msgstore/db
apps/com.whattsapp/db/wa.db
apps/com.whattsapp/db/axolot1.db
apps/com.whattsapp/db/chatsettings.db

extracting whattsapp-cryptkey …
```

Figure 2. WhatsApp key/DB extractor - Downgrade and Backup WhatsApp



```
Installing legacy WhatsApp 2.11.431
3909 KB/s (18329558 bytes in 4.579s)
        pkg: /data/local/tmp/LegacyWhatsApp.apk
Success
Install complete

Now unlock your device and confirm the backup operation.

Please enter your backup password (leave blank for none) and press Enter:

apps/com.whatsapp/f/key
apps/com.whatsapp/db/msgstore.db
apps/com.whatsapp/db/wa.db
apps/com.whatsapp/db/axolotl.db
apps/com.whatsapp/db/chatsettings.db

Extracting whatsapp.cryptkey ...
        1 file(s) copied.

Extracting msgstore.db ...
        1 file(s) copied.

Extracting wa.db ...
        1 file(s) copied.

Extracting axolotl.db ...
        1 file(s) copied.

Extracting chatsettings.db ...
```

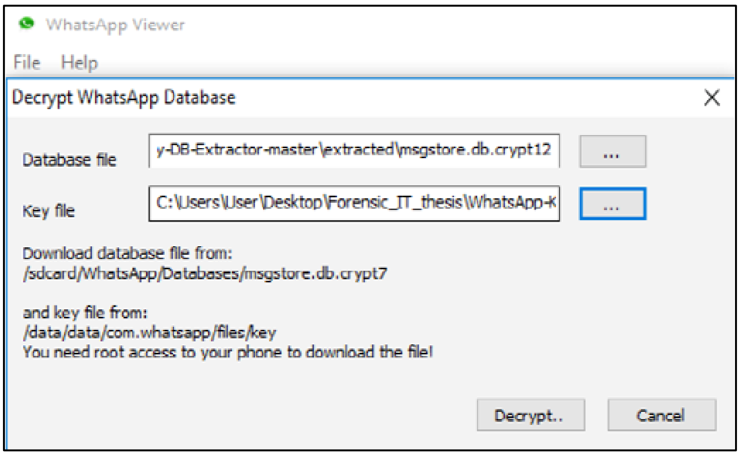
Figure 3. Extraction of msgstore.db, wa.db, and the crypt key


This PC › Desktop › Forensic_IT_thesis › WhatsApp-Key-DB-Extractor-master › extracted

Name	Date modified	Type	Size
.placeholder	10/20/2016 7:26 PM	PLACEHOLDER File	1 KB
axolotl	3/14/2019 11:30 AM	Data Base File	512 KB
chatsettings	3/10/2019 2:10 AM	Data Base File	24 KB
msgstore	3/14/2019 5:18 AM	Data Base File	396 KB
wa	3/13/2019 6:09 PM	Data Base File	128 KB
whatsapp.cryptkey	3/12/2019 1:36 PM	CRYPTKEY File	1 KB

Figure 4. DBs and the key extracted from the phone

Once the key is extracted, it can be used to decrypt msgstore.db.crypt12 or any crypt12 WhatsApp SQLite DB created on this device. This is done by an open-source tool: WhatsApp viewer (Figures 5 and 6). It is a very simple tool that associates the database with its key in order to decrypt it.



Decrypt WhatsApp Database

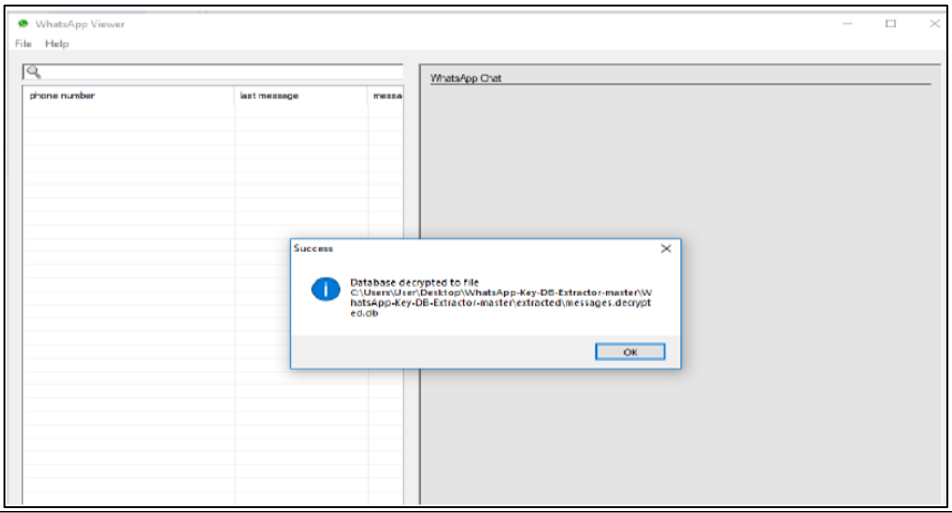
Database file: y-DB-Extractor-master\extracted\msgstore.db.crypt12

Key file: C:\Users\User\Desktop\Forensic_IT_thesis\WhatsApp-K

Download database file from:
/sdcard/WhatsApp/Databases/msgstore.db.crypt7

and key file from:
/data/data/com.whatsapp/files/key
You need root access to your phone to download the file!

Figure 5. WhatsApp viewer tool



Success

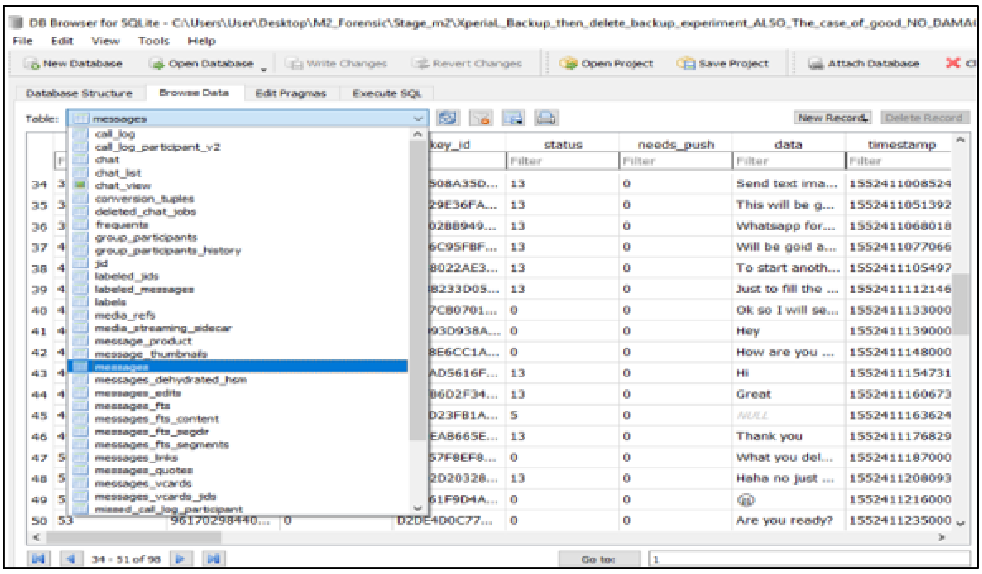
Database decrypted to file
C:\Users\User\Desktop\WhatsApp-Key-DB-Extractor-master\WhatsApp-Key-DB-Extractor-master\extracted\messages.decrypted.db

Database decrypted to file

C:\Users\Desktop\WhattsApp.Key-DB-Extractor-master\WhattsApp-Key-DB-Extractor-master\Extracted\messages.decrypted.db

Figure 6. The database is successfully decrypted

Now, the decrypted databases can be manually parsed using SQLite viewer such as DB Browser for SQLite (Figure 7).



DB Browser for SQLite - C:\Users\User\Desktop\M2_Forensic\Stage_m2\Xperial_Backup_then_delete_backup_experiment_ALSO_The_case_of_good_NO_DAMA

Table: messages

call_log
call_log_participant_v2
chat
chat_list
chat_view
conversion_tuples
deleted_chat_jobs
frequents
group_participants
group_participants_history
jid
labeled_jids
labeled_messages
labels
media_refs
media_streaming_sidecar
message_product
message_thumbnails
messages
messages_dehydrated_hsm
messages_edits
messages_fts
messages_fts_content
messages_fts_segdir
messages_fts_segments
messages_links
messages_quotes
messages_vcards
messages_vcards_jids
missed_call_log_participant

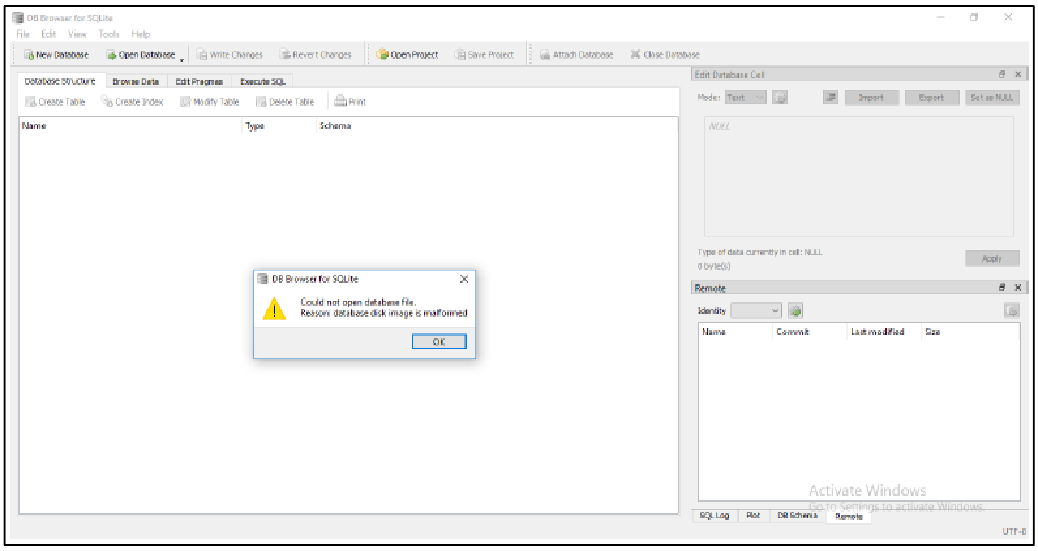
				key_id	status	needs_push	data	timestamp
	F				Filter	Filter	Filter	Filter
34	3			508A35D...	13	0	Send text ima...	1552411008524
35	3			29E36FA...	13	0	This will be g...	1552411051392
36	3			02B8949...	13	0	Whatsapp for...	1552411068018
37	4			6C95F8F...	13	0	Will be goid a...	1552411077066
38	4			8022AE3...	13	0	To start anoth...	1552411105497
39	4			B233D05...	13	0	Just to fill the ...	1552411112146
40	4			7C80701...	0	0	Ok so I will se...	1552411133000
41	4			93D938A...	0	0	Hey	1552411139000
42	4			8E6CC1A...	0	0	How are you ...	1552411148000
43	4			AD5616F...	13	0	Hi	1552411154731
44	4			B6D2F34...	13	0	Great	1552411160673
45	4			D23FB1A...	5	0	NULL	1552411163624
46	4			EAB665E...	13	0	Thank you	1552411176829
47	5			57F8EF8...	0	0	What you del...	1552411187000
48	5			2D20328...	13	0	Haha no just ...	1552411208093
49	5			61F9D4A...	0	0	☺	1552411216000
50	53	96170298440...	0	D2DE4D0C77...	0	0	Are you ready?	1552411235000

34 - 51 of 98

Go to: 1

Figure 7. The decrypted database msgstore opened in DB Browser for SQLite Corrupted Databases

SQLite is basically a highly reliable, embedded, and self-contained SQL database engine. However, due to a certain error, this database can be corrupted [19]. Frequently, the corruption results in an error reading: "*Disk Image Is Malformed*". In this case, the decryption of this damaged database would be successful but the database would not be opened on any SQLite database viewer, showing the warning in Figure 8.



DB Browser for SQLite

Could not open database file.
Reason: database disk image is malformed

Type of data currently in cell: NULL
0 byte(s)

Figure 8. Damaged database: Disk image is malformed

To recover the database, by using the SQLite command-line tool, the following command sequence is run:

```
sqlite3.exe msgstore.db
.mode insert
.output msgstore_dump.sql
.dump
.exit
```

Now, an SQL file (msgstore.db) with dumped database tables will be obtained. It is imported to a new SQLite database. In this paper, the database was heavily damaged, so it was necessary to examine the file manually. The SQL file was opened in a text editor (sublime text) and the tables of interest were saved in separate SQL files then used DB Browser for SQLite to create a database. Figure 9 shows the recovered table messages from the damaged database, compared to all tables as shown in Figure 7. For each table, this process should be respected in order to open the damaged tables in the database.

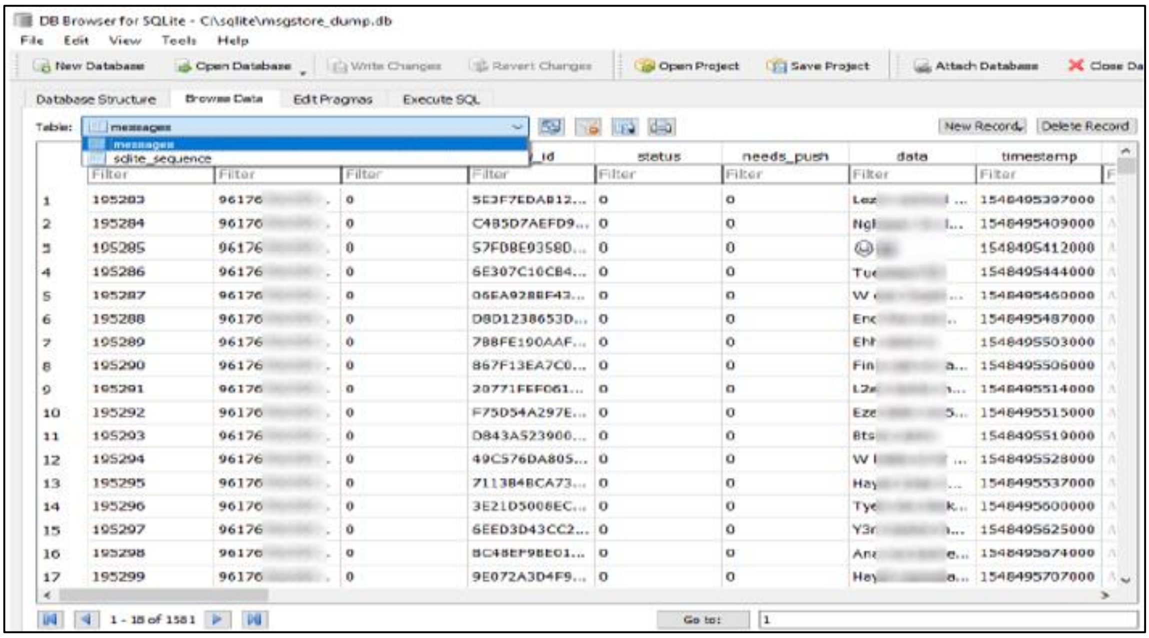


Table: messages
messages
sqlite_sequence

				_id	status	needs_push	data	timestamp
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	195283	96176[redacted].	0	5E3F7EDA812...	0	0	Lez[redacted]l ...	1548495397000
2	195284	96176[redacted].	0	C485D7AEFD9...	0	0	Ngl[redacted]l...	1548495409000
3	195285	96176[redacted].	0	57FD8E9358D...	0	0	☺[redacted]	1548495412000
4	195286	96176[redacted].	0	6E307C10CB4...	0	0	Tue[redacted]	1548495444000
5	195287	96176[redacted].	0	06EA9288F43...	0	0	W [redacted]...	1548495460000
6	195288	96176[redacted].	0	D8D1238653D...	0	0	Enc[redacted]..	1548495487000
7	195289	96176[redacted].	0	788FE190AAF...	0	0	Ehh[redacted]	1548495503000
8	195290	96176[redacted].	0	867F13EA7C0...	0	0	Fin[redacted]a...	1548495506000
9	195291	96176[redacted].	0	20771FEF061...	0	0	L2e[redacted]n...	1548495514000
10	195292	96176[redacted].	0	F75D54A297E...	0	0	Eze[redacted]5...	1548495515000
11	195293	96176[redacted].	0	D843A523900...	0	0	8ts[redacted]	1548495519000
12	195294	96176[redacted].	0	49C576DA805...	0	0	W l[redacted] ...	1548495528000
13	195295	96176[redacted].	0	711384BCA73...	0	0	Hay[redacted]...	1548495537000
14	195296	96176[redacted].	0	3E21D5008EC...	0	0	Tye[redacted]k...	1548495600000
15	195297	96176[redacted].	0	6EED3D43CC2...	0	0	Y3r[redacted]n...	1548495625000
16	195298	96176[redacted].	0	8C48EF98E01...	0	0	Anz[redacted]e...	1548495674000
17	195299	96176[redacted].	0	9E072A3D4F9...	0	0	Hey[redacted]a...	1548495707000

1 - 18 of 1581

Go to: 1

Figure 9. Recovered messages table from the damaged "msgstore" database

4. Results and Discussion

4.1. Analysis of Contacts Database "wa.db"

The analysis of the contacts list is very important to know with whom the user was interacting. The experiments listed in Table 1 were carried out.

1. The structure of the contacts database wa.db

The first step of the analysis of the user contacts is to study the structure of the database “wa.db”. This database contains different tables. The valuable information concerning each contact are stored as a record mainly in one table, namely “wa_contacts”. This information is stored under several fields (columns) based on the origin of the data (set by WhatsApp system or stored by the user in the phonebook). The structure of the table is described in Tables 6 and 7. WhatsApp updated its tables by adding a new table that was not there before. We find that this new table can be useful as it contains valuable information about the blocked contacts. This table is called “wa_block_list”.

Table 6. Structure of the contacts wa.db - Information set by WhatsApp

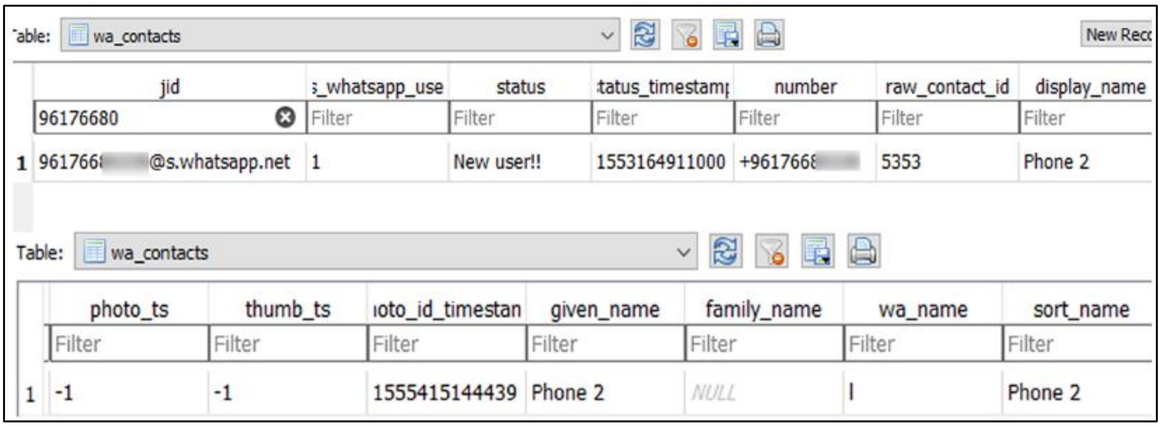
Contact information from WhatsApp system	
Field name	**Information presented**
_id	The number of record set by SQLite
Jid	WhatsApp ID of the contact containing his number
is_whatsapp_user	If the contact is an active WhatsApp user
unseen_msg_count	The number of the unread messages by the user
photo_id_timestamp	Unix epoch time when the profile picture was set
wa_name	WhatsApp name of the user set by him/herself

Table 7. Structure of the contacts wa.db - Information set by Phonebook

Contact information from phonebook	
Field name	**Information presented**
Number	The phone number of the contact
raw_contact_id	Record number
display_name	The contact name set by the user
given_name	The name of the user
family_name	The family name of the user

2. Reconstruction of the contacts list

To reconstruct the contacts list, we have to analyze the different values stored in the fields of the Tables 6 and 7. The Figures 10 and 11 show the database wa.db opened in DB browser for SQLite. We just show two records for demonstration. In the first record (Figure 10), each contact is associated with a WhatsApp ID which is stored under the "jid" field, with the structure "number@.whatsApp.net", where the number refers to the phone number of the contact (here 76680***). The contact is also associated with a Boolean value stored under the field "is_whatsApp_use" indicating whether the contact is an active WhatsApp user or not. The user is an active WhatsApp member if "is_whatsApp_user = 1". In addition, the field "given_name" stores the name given by the user to the contact, (here Phone 2), while the field "wa_name" stores the name of the contact set by the contact itself (here l). Furthermore, the "About" and its set time (previously known as status) is stored in the field "status" and "status_timestamp", respectively. The avatar picture can link the user to his real identity if the picture displays his face or location. The avatar picture of a contact is stored in media/pictures folder but we did not find the thumbnail of the profile picture stored in the database. The timestamp stored in the field "thumb_ts" indicates when the contact has set its current avatar, respectively. In this figure, the value stored under this field is "-1", which means that no profile picture is set for this contact. As this table stores the information about individual contacts, it stores also the information about the group that the user has joined. In this case, the WhatsApp ID is structured under a different string such as the string 96170298***-1555701066@g.us (Figure 11). The phone number is that of the group creator and the UNIX epoch time is the group creation time. The name of the group, here “Forensic IT test”, is stored under the "display_name". Note that the members of the group are not stored under any fields in this table.



able: wa_contacts

	jid	s_whatsapp_use	status	tatus_timestam	number	raw_contact_id	display_name
	96176680	Filter	Filter	Filter	Filter	Filter	Filter
1	961766[redacted]@s.whatsapp.net	1	New user!!	1553164911000	+961766[redacted]	5353	Phone 2

Table: wa_contacts

	photo_ts	thumb_ts	ıoto_id_timestan	given_name	family_name	wa_name	sort_name
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	-1	-1	1555415144439	Phone 2	NULL	I	Phone 2

Figure 10. wa_contacts table - individual record

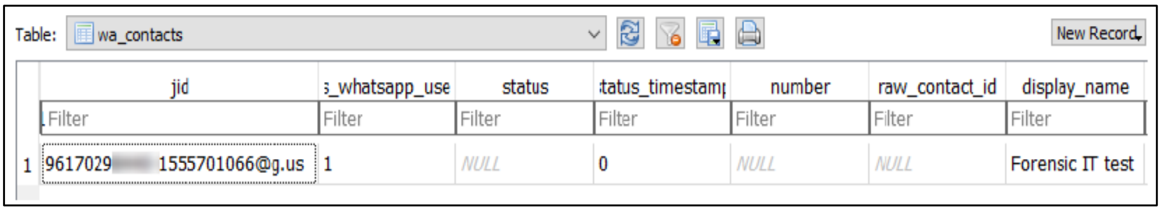


Table: wa_contacts

	jid	s_whatsapp_use	status	tatus_timestam	number	raw_contact_id	display_name
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	9617029[redacted]1555701066@g.us	1	NULL	0	NULL	NULL	Forensic IT test

Figure 11. wa_contacts table - group record

3. Added and deleted contacts

In addition to the explicit insertion or deletion of a contact (where these operations are carried out by the user), WhatsApp Messenger is able to synchronize the phonebook of the device with the contact list. In particular, it automatically adds to the contact list any WhatsApp user whose phone number is stored in the phonebook of the device. Furthermore, it automatically removes from the contact list any WhatsApp user whose phone number is removed from the phonebook. Being able to tell which contacts have been added deliberately by the user may be important in some investigative scenarios. The results of our analysis, however, show that WhatsApp Messenger does not store in any database information allowing one to distinguish between the explicitly-added and automatically-added contacts. Nevertheless, Anglano [10] found that explicitly-added contacts can be identified by analyzing the database corresponding to the phonebook of the device, which is implemented as an SQLite database named contacts2.db. Unfortunately, this is not our interest as we are trying just to collect information from the artefacts left in the WhatsApp directory.

4. Blocked contacts

WhatsApp allows user to block a contact, which prevents any type of communication between the user and the contact as well as getting any update of the profile picture and status. When the user blocks a contact, WhatsApp adds a record to the table “wa_block_list” of the contact database “wa.db”. This table (depicted in Figure 12) contains one field called "jid" that stores the ID (phone number) of the blocked contacts.

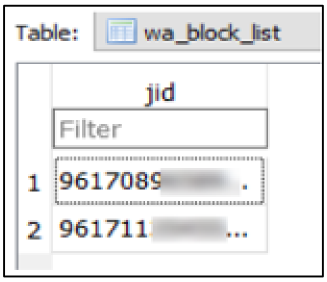


Table: wa_block_list

	jid
	Filter
1	9617089[redacted].
2	961711[redacted]...

Figure 12. “wa_block_list” table

Therefore, the other information of the blocked users may be deduced by selecting those records in table "wa_contacts" using the following SQL query: SELECT * FROM wa_contacts WHERE jid IN (SELECT jid FROM wa_block_list). The results of our analysis show also that when a contact is unblocked, the corresponding record in table "wa_block_list" is deleted. Furthermore, it is not possible to tell whether a currently unblocked contact has been blocked in the past, or how many times a currently blocked contact has been blocked and unblocked in the past. As a final consideration, we note that no information is stored on the side of the contact that gets blocked, so it is not possible to tell whether the user of the device under analysis has been blocked or not by anyone of their contacts.

4.2. Analysis of Chat Database "msgstore.db"

This database has a very evidentiary value. Its analysis reveals evidence about the content of the messages, the time the messages have been sent or received, and what type of communication the user has been involved in.

1. The structure of "msgstore":

The chat database contains different tables, the two most important tables that have evidentiary value concerning the messages exchanged are:

- messages: where every message and its details are stored as a record. The data is stored in multiple fields, which can be classified into two categories, the first is the characteristics of the messages (listed in Table 8) and the second is the content of the messages (listed in Table 9).
- chat_list: where the messages are classified based on the contact involved. Fields are presented in Table 10.

In the following, we discuss how to correlate the values stored in these fields to the actions taken by the user.

Table 8. Structure of the chat database msgstore.db - message characteristics

Message Characteristics	
Field name	**Information stored**
_id	Record number set by SQLite
key_remote_jid	WhatsApp ID of the contact
key_id	Message identifier
key_from_me	Message sender
status	The status of message (delivered or not)
timestamp	Time of message sending (Unix epoch format from the user device clock)
received_timestamp	Time of message receiving
receipt_server_timestamp	Time of message when delivered to the server
reicept_device_timestamp	Time of delivery to the contact
needs_push	Broadcast message
recipient_count	Number of recipients in a broadcast message
remote_ressource	Group message

Table 9. Structure of the chat database msgstore.db - message content

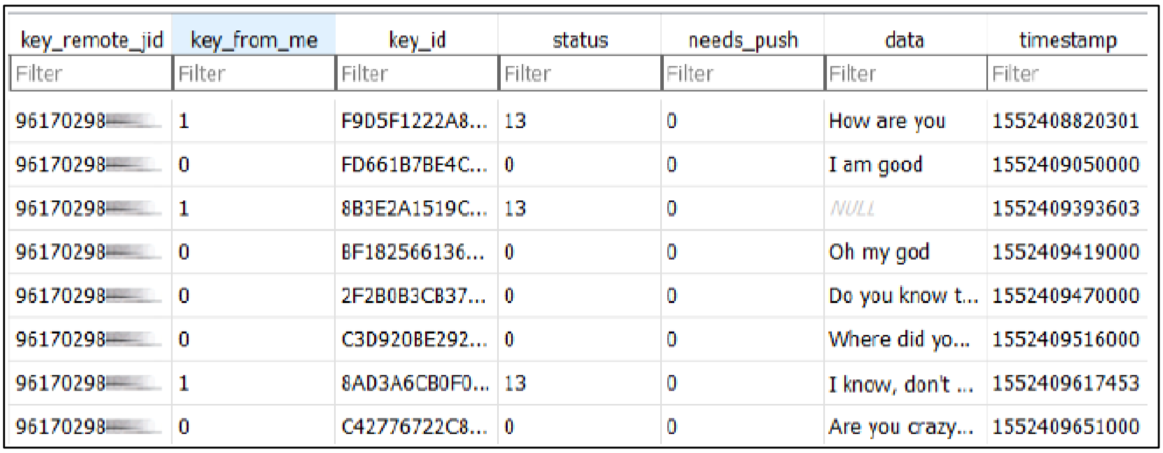
Message Characteristics	
Field name	**Information stored**
media_wa_type	Message type (text, media, …)
data	Message content when text
media_mime_type	Exact type of the media message
raw_data	Thumbnail of the media message
media_hash	Hash of the media message
media_url	URL of the media message
media_size	Size of the media message
media_name	Name of the media file
media_duration	Time in seconds of a media file (video, audio)
latitude	Latitude of the message (location)
longitude	Longitude of the message (location)

Table 10. Structure of the chat database chat_list.db

Field name	Information stored
_id	Record number set by SQLite
key_remote_jid	WhatsApp IF of the contact
message_id_table	Record number in the messages table of the last message of the conversation

2. Determination of the chat history:

To determine how, when, and with whom the conversation had started, we should decode the fields of the table "messages" presented in Figure 13.



key_remote_jid	key_from_me	key_id	status	needs_push	data	timestamp
Filter	Filter	Filter	Filter	Filter	Filter	Filter
96170298[redacted]	1	F9D5F1222A8...	13	0	How are you	1552408820301
96170298[redacted]	0	FD661B7BE4C...	0	0	I am good	1552409050000
96170298[redacted]	1	8B3E2A1519C...	13	0	*NULL*	1552409393603
96170298[redacted]	0	BF182566136...	0	0	Oh my god	1552409419000
96170298[redacted]	0	2F2B0B3CB37...	0	0	Do you know t...	1552409470000
96170298[redacted]	0	C3D920BE292...	0	0	Where did yo...	1552409516000
96170298[redacted]	1	8AD3A6CB0F0...	13	0	I know, don't ...	1552409617453
96170298[redacted]	0	C42776722C8...	0	0	Are you crazy...	1552409651000

Figure 13. Reconstruction of the chat history

The records presented in Figure 13 show that the user had a conversation with a contact who has the number 96170298*** which is clear from the field "Key_remote_jid". This field indicates the phone number of the user involved in the conversation. The field "key_from_me" indicates the message direction. If "Key_from_me" = 0, then the user receives the message (incoming text) and if "Key_from_me" = 1, the user sends the message (outgoing text).

In this case, by analyzing the first record, the user started the conversation (Key_from_me = 1) by sending a text message "How are you" stored in the "data" field, which contains the content of the textual messages. The field "timestamp" indicates the time the message was sent and "received_timestamp" indicates the time the message was received (by the user). Here, the message "How are you" was sent by the device owner (Key_from_me = 1) at Tuesday 12th of march, 2019 4:40:20:30 PM ("timestamp") and the contact replied at the same day at 4:44:10 PM ("received_timestamp") with “I am good” as shown in the second record. Note that the time in these fields is stored as a UNIX epoch time.

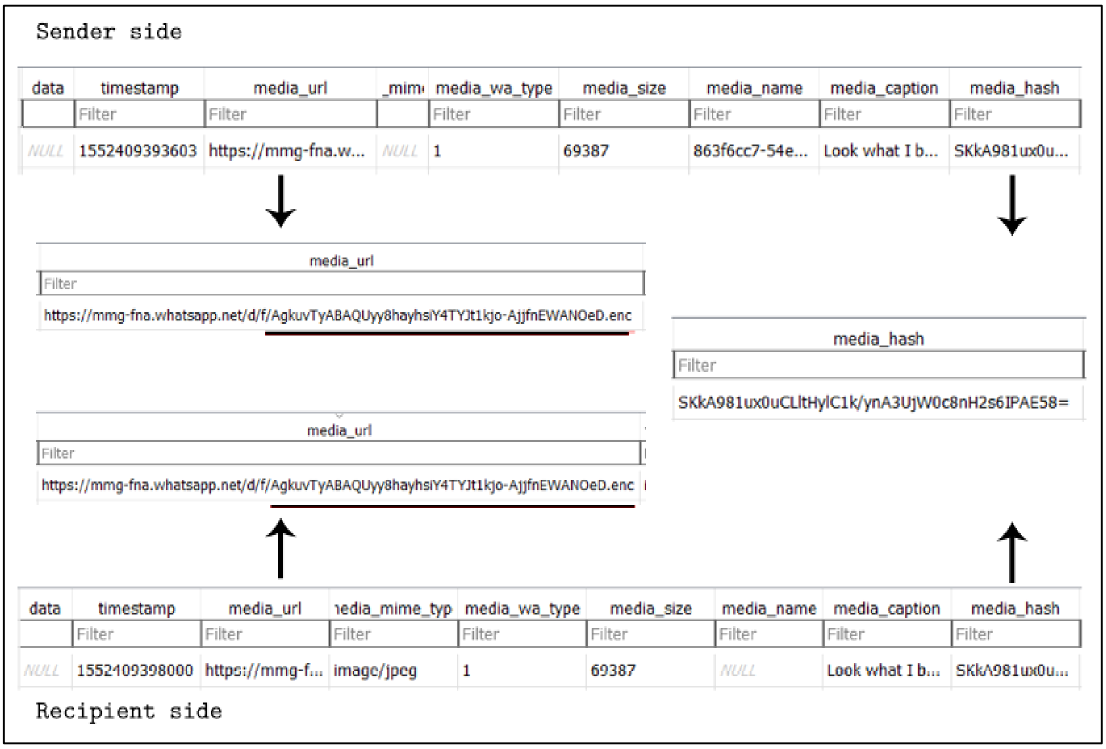
As shown, each message (record) is associated with a unique identifier under the field "key_id". This field is used usually to correlate the information stored about a message in different tables.

3. Analysis of messages content:

WhatsApp is used to exchange all types of information, such as text, images, videos, audios, contacts, and locations. The type of data is determined by looking at the field "media_wa_type". If "media_wa_type = 0", the messages exchanged are textual and then stored in the "data" field. Otherwise, the messages could be a multimedia file, a contact card or a location. To determine exactly what the type of the non-textual message is, we should look at the other fields that depend on the type of the messages exchanged.

- Multimedia files

When the messages exchanged are media files (images, videos, audios), WhatsApp copies these files into this folder in the internal memory WhatsApp/Media (if the user sends the file, the location is WhatsApp/Media/Sent). Anglano [16] explained this process. WhatsApp uploads the file to the WhatsApp server, which sends back the URL of the corresponding location. Finally, the sender sends to the recipient a message containing this URL and, upon receiving this message, the recipient sends an acknowledgment back to the sender. Following this, a record is stored in the messages under different fields. First, the type of the file is determined by the field "wa_media_type" if the message is not a text. For images "wa_media_type = 1", for videos "wa_media_type = 3" and for audio "wa_media_type = 2". The field "wa_mime_type" indicates exactly what the type of the media file is. For example, if the file is an image, the value stored is JPEG or JPG (for video is MP4 and for audio is aac). The name of the file is stored in "media_name" column and its size in bytes in "media_size". The URL, which corresponds to its location in the server (temporarily storage), is stored in "media_url" and the hash of the file is stored in "media_hash" field. These fields are shown in Figure 14. These fields are the same on the recipient side except the "media_url". In this field, the URL is different but the name given by the server to the file is the same. We can note that the "media_name" is empty on this side, the comparison between the sender and the recipient hash and URL helps to identify if it is the same file on the two sides.



Sender side

data	timestamp	media_url	_mim	media_wa_type	media_size	media_name	media_caption	media_hash
Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
NULL	1552409393603	https://mmg-fna.w...	NULL	1	69387	863f6cc7-54e...	Look what I b...	SKkA981ux0u...

media_url
Filter
https://mmg-fna.whatsapp.net/d/f/AgkuvTyABAQUyy8hayhsiY4TYJt1kjo-AjjfnEWANOeD.enc

media_hash
Filter
SKkA981ux0uCLltHylC1k/ynA3UjW0c8nH2s6IPAE58=

media_url
Filter
https://mmg-fna.whatsapp.net/d/f/AgkuvTyABAQUyy8hayhsiY4TYJt1kjo-AjjfnEWANOeD.enc

data	timestamp	media_url	nedia_mime_typ	media_wa_type	media_size	media_name	media_caption	media_hash
Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
NULL	1552409398000	https://mmg-f...	image/jpeg	1	69387	NULL	Look what I b...	SKkA981ux0u...

Recipient side

Figure 14. Multimedia message content: the sender and the recipient

To decode the thumbnail of an exchanged image, one must analyze another table, the "messages_thumbnail". The following command must be run:

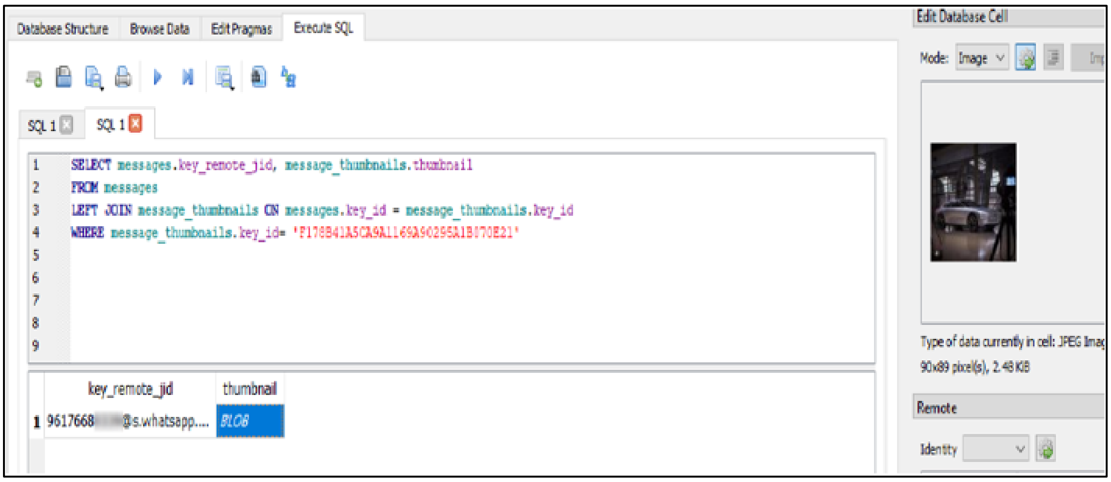
SELECT messages.key_remote_jid, message_thumbnails.thumbnail

FROM messages

LEFT JOIN message_thumbnails ON messages.key_id = message_thumbnails.key_id

WHERE message_thumbnails.key_id= ’3A7D65066A639163C948’

This is shown in Figure 15.



```
SELECT messages.key_remote_jid, message_thumbnails.thumbnail
FROM messages
LEFT JOIN message_thumbnails ON messages.key_id = message_thumbnails.key_id
WHERE message_thumbnails.key_id= 'F178B41A5CA9A1169A90295A1B870E21'
```

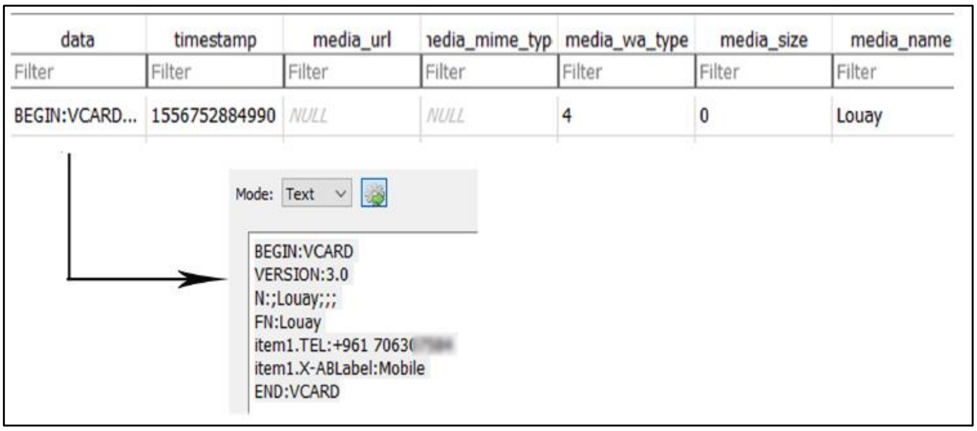
	key_remote_jid	thumbnail
1	9617668[redacted]@s.whatsapp....	BLOB

Type of data currently in cell: JPEG Imag
90x89 pixel(s), 2.48 KB

Figure 15. The thumbnail of an image message

- Contact cards

WhatsApp enables the user to exchange contacts from the phonebook of the sender. In this case, the "media_wa_type" is "4". The messages are sent in VCARDS format and are stored in the data field. The number of the contact exchanged is stored in this field, here 7063****. The name of this contact (as it’s saved in the phonebook) is stored in "media_name" field. The other fields have the same meaning and value as other type of data. In the recipient side, all the fields are the same except the "from_me" is "0". Figure 16 shows a contact card from the sender’s side.



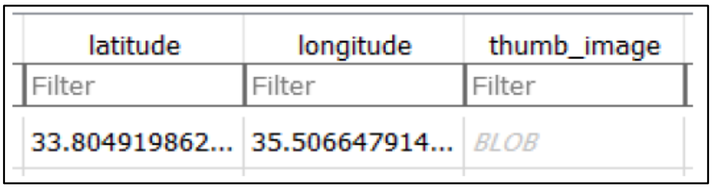
data	timestamp	media_url	ıedia_mime_typ	media_wa_type	media_size	media_name
Filter	Filter	Filter	Filter	Filter	Filter	Filter
BEGIN:VCARD...	1556752884990	NULL	NULL	4	0	Louay

```
BEGIN:VCARD
VERSION:3.0
N:;Louay;;;
FN:Louay
item1.TEL:+961 7063[redacted]
item1.X-ABLabel:Mobile
END:VCARD
```

Figure 16. Contact card

- Geolocation

WhatsApp provides the users with the ability to send their actual location or any other location on the map. The type of data in this case is marked by "5" in the field "media_wa_type". This information is stored in "latitude" and "longitude" columns. A thumbnail is stored in the other table “message_thumbnails”. An example of such record is shown in Figures 17 and 18.



latitude	longitude	thumb_image
Filter	Filter	Filter
33.804919862...	35.506647914...	BLOB

Figure 17. Geolocation latitude and longitude

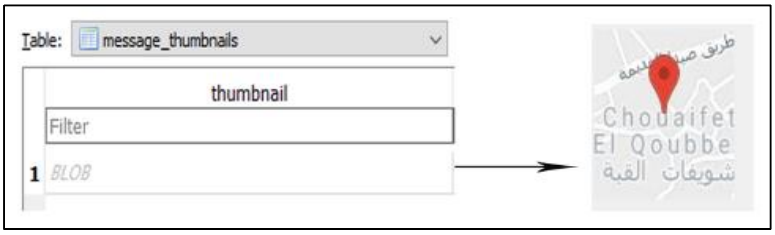


Table: message_thumbnails

	thumbnail
	Filter
1	BLOB

Figure 18. Thumbnail of the location from the table message_thumbnails

4. Message state

Messages are not exchanged directly among communicating users, but they are first sent to the central server, that forwards them to the respective recipients if they are online. Otherwise, it stores them in the local server until they can be delivered. When a message is stored in the sender database, it has not necessarily been delivered to the contact. In fact, there are three possible states.

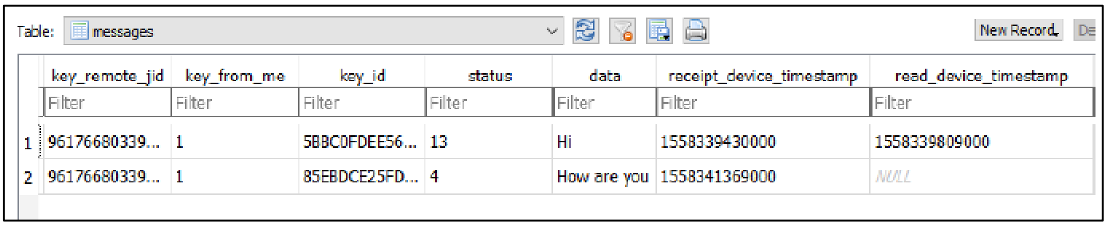
i. The message is sent from the user but it is not transmitted to the server (the clock sign).

ii. The message is sent by the user to the server but still not delivered to its recipient (one gray tick).

iii. The message is delivered to its recipient (two gray tick mark) and is read.

The analysis of the message state is very important during investigation to know if the message was delivered or not to its recipients. To reveal this information, several fields in the "messages" table of the sender database must be analyzed. The first field is "status", which will indicate whether the message has reached the server. If the message is sent but still not transmitted to the server, a record is stored in the database. In this case, the status = 0 given that "key_from_me = 1" (status is always zero when key_from_me = 0).

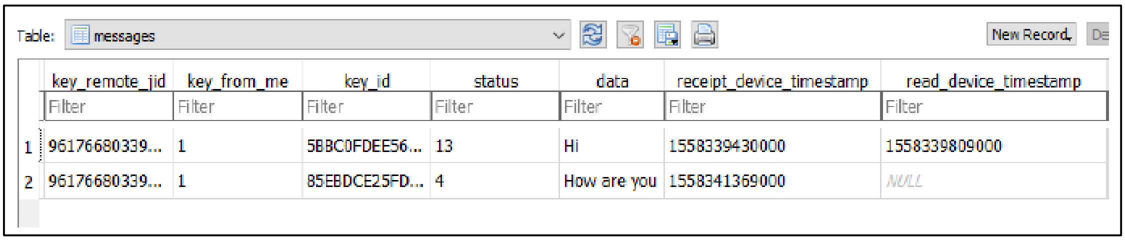
The field “timestamp" indicates the time the message was sent by the user (Figure 19). If the message is transmitted to the server but it is not delivered to the recipients, the "status = 5" and the time when the message reaches the server is stored in "receipt_ server_timestamp" (Figure 19).



	key_remote_jid	key_from_me	key_id	status	data	receipt_device_timestamp	read_device_timestamp
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	96176680339...	1	5BBC0FDEE56...	13	Hi	1558339430000	1558339809000
2	96176680339...	1	85EBDCE25FD...	4	How are you	1558341369000	NULL

Figure 19. Message state: message is transmitted to server(1) and message is not transmitted to server(2)

When the message is delivered to its destination, the status = 4 and the time the recipient receives the message is stored in the field "receipt_device_timestamp". When the message is read by that recipient, the status = 13 and the time the recipient reads the message is stored in "read_device_timestamp". In the case of an audio message, the time the audio is heard by the contact is stored in "played_device_timestamp" (Figure 20).



	key_remote_jid	key_from_me	key_id	status	data	receipt_device_timestamp	read_device_timestamp
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	96176680339...	1	5BBC0FDEE56...	13	Hi	1558339430000	1558339809000
2	96176680339...	1	85EBDCE25FD...	4	How are you	1558341369000	NULL

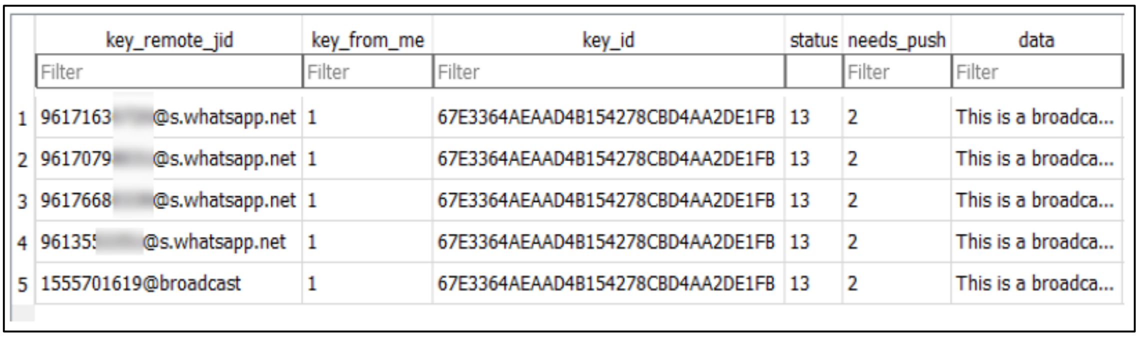
Figure 20. Message state: delivered and read messages

By analyzing the aforementioned fields together, the status of the messages can be revealed.

4.3. Multiple Message Destinations

1. Broadcast Messages

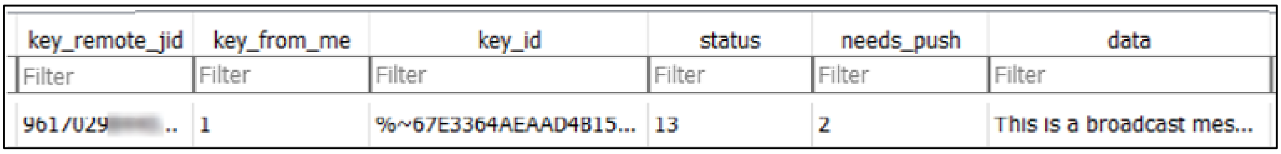
WhatsApp enables users to send the same message to multiple contacts at the same time privately and the contacts' reply is shown just for the sender. When the user sends a broadcast, a record is generated in the table messages for each recipient. All these records (messages) have the same identifier in the field "key_id", which helps to identify the nature of the message as a broadcast (Figure 21). The phone number of the recipients and their IDs are stored in "key_remote_jid". The user WhatsApp ID is marked with the word broadcast. The field "recipient_count" identifies the number of the recipients involved in this broadcast message. On the recipient side, the received broadcast message is saved in just one record in messages table. This record is distinguished from other records by the presence of the %_ sign in the field "key_id". This is shown in Figure 22. In the case of a broadcast message sent for a non-saved contact, it will not be delivered to the recipient. Our experiment shows that if the recipient replies to the message, the record generated on both sides would not be distinguished from any other private message, and this is because the reply is sent just to the original sender as a regular private message between the user and the contact.



	key_remote_jid	key_from_me	key_id	status	needs_push	data
	Filter	Filter	Filter		Filter	Filter
1	9617163[redacted]@s.whatsapp.net	1	67E3364AEAAD4B154278CBD4AA2DE1FB	13	2	This is a broadca...
2	9617079[redacted]@s.whatsapp.net	1	67E3364AEAAD4B154278CBD4AA2DE1FB	13	2	This is a broadca...
3	9617668[redacted]@s.whatsapp.net	1	67E3364AEAAD4B154278CBD4AA2DE1FB	13	2	This is a broadca...
4	961355[redacted]@s.whatsapp.net	1	67E3364AEAAD4B154278CBD4AA2DE1FB	13	2	This is a broadca...
5	1555701619@broadcast	1	67E3364AEAAD4B154278CBD4AA2DE1FB	13	2	This is a broadca...

Note: Real phone numbers blurred for privacy issue

Figure 21. Broadcast records



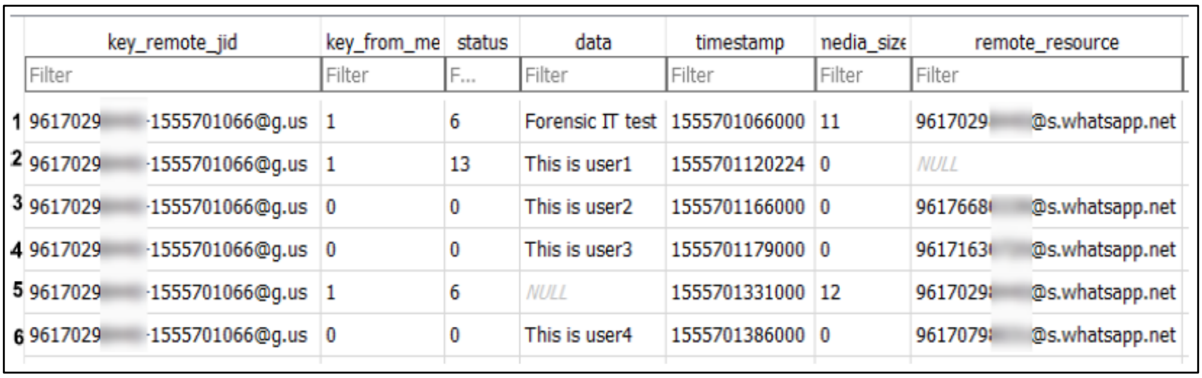
key_remote_jid	key_from_me	key_id	status	needs_push	data
Filter	Filter	Filter	Filter	Filter	Filter
961/029[redacted]..	1	%~67E3364AEAAD4B15...	13	2	This is a broadcast mes...

Figure 22. Broadcast message - recipient side

2. Group Chat

WhatsApp allows another type of chat communication, where messages are sent within a group of members and every message is shown to all of them. As the other types, a message sent within a group is stored as a record in the messages. The analysis of this case requires the study of different fields to investigate the events that happened within the group. To do this analysis, we created a group of four members (including the group creator). The textual messages sent contain the name of each user namely user 1, user 2, user 3 and user 4. The first field that should be analyzed is 'key_remote_jid' because this would give the information about the creator of the group, the creation time and the group ID. As shown in Figure 23, this field contains, all the records, the creator's phone number and the creation time of the group as in the following string 9617029****-1555701066@g.us. The time format is the UNIX epoch which means that the group was created at Friday, April 19, 2019 10:11:06 PM by the user who has the number 9617029****. The name of this group is stored in the field data, here Forensic IT test, where the field media_size = 11 in record number 1 referring to the action of creation of the group. The record number 2 corresponds to the message sent by the creator. On the other hand, when a member of the group sends a message, his/her number is stored in the field "remote_resource" and not in the field "key_remote_id" because this last field is always related to the admin. This is the case in record number 3 and number 4, where user 2 and user 3 have the numbers 9617668**** and 9617163****, respectively. The record number 5 is the action of adding a user by the admin.

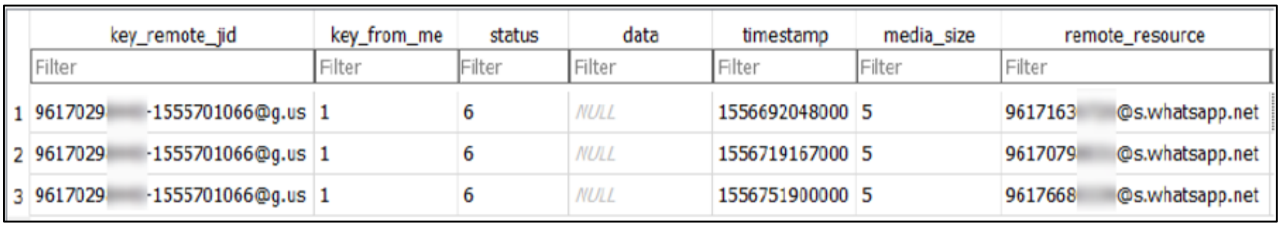
To identify this action, we look at the field media_size which in this case is "12". The record number 6 is analyzed in the same way for records number 4 and number 5. Here, user 4 has the number 9617079****. Note that there is no such a record like record number 5 (media_size = 12) for both user 2 and user 3, meaning that these two members were added in the same action of the group creation. These records lack some information that are important for the investigation process. For example, no record tracks the identity of the group members that receive a specific message at any point in time. Although this information is not stored explicitly in the database, it can be deduced by examining the fields that store when a member is added and when a member leaves. This field is media_size which is the same field that stores the creation of group. We observe from the previous points that when media_size equals "11", it represents the creation of the group while when it is "12", it indicates adding a member. This field also stores a value when a member leaves/is removed. To clarify, we did an experiment in which the users mentioned above left the group in order to reconstruct the chronology of the group composition. What we can conclude from Figure 23 is that user 1 created the group, named Forensic IT test, on Friday, April 19, 2019 at 10:11:06 PM (field timestamp) and he added in the same time of creation user 2 (9617668****) and user 3 (9617163****), then added the user 4 (9617079****) on the same day at 10:15:31 PM. Note that this applies to all the users including the group creator.



	key_remote_jid	key_from_me	status	data	timestamp	nedia_size	remote_resource
	Filter	Filter	F...	Filter	Filter	Filter	Filter
1	9617029[redacted]-1555701066@g.us	1	6	Forensic IT test	1555701066000	11	9617029[redacted]@s.whatsapp.net
2	9617029[redacted]-1555701066@g.us	1	13	This is user1	1555701120224	0	NULL
3	9617029[redacted]-1555701066@g.us	0	0	This is user2	1555701166000	0	9617668[redacted]@s.whatsapp.net
4	9617029[redacted]-1555701066@g.us	0	0	This is user3	1555701179000	0	9617163[redacted]@s.whatsapp.net
5	9617029[redacted]-1555701066@g.us	1	6	NULL	1555701331000	12	9617029[redacted]@s.whatsapp.net
6	9617029[redacted]-1555701066@g.us	0	0	This is user4	1555701386000	0	9617079[redacted]@s.whatsapp.net

Figure 23. Groups chat records in chat database

After that, user 3 left the group. This action, in record number 1 in Figure 24, is stored in the database under the field media_size with value "5" indicating leaving the group. The identity of the user is reported in the field remote_resource. This user left on Wednesday, May 1, 2019 9:27:28 AM (field timestamp). Then, by the same analysis, we know that user 4 and user 2 left the group on Wednesday, May 1, 2019 4:59:27 PM and Thursday, May 2, 2019 2:05:00 AM, respectively.



	key_remote_jid	key_from_me	status	data	timestamp	media_size	remote_resource
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	9617029[redacted]-1555701066@g.us	1	6	NULL	1556692048000	5	9617163[redacted]@s.whatsapp.net
2	9617029[redacted]-1555701066@g.us	1	6	NULL	1556719167000	5	9617079[redacted]@s.whatsapp.net
3	9617029[redacted]-1555701066@g.us	1	6	NULL	1556751900000	5	9617668[redacted]@s.whatsapp.net

Figure 24. Group's records created when a member leaves the group

So, this will help in the reconstruction of the composition of the group and thus to identify whether a user was in the group during the conversation (Figure 25).

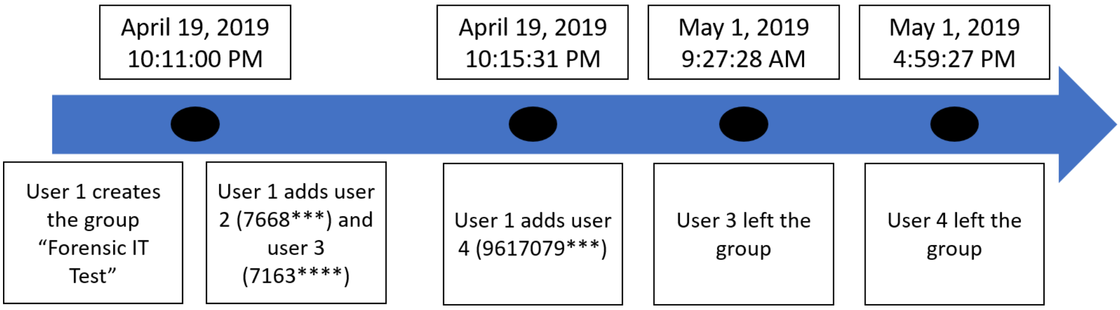


Figure 25. Timeline shows the chronology of the group composition

4.4. Voice and Call Logs

WhatsApp allows users to place voice and videos calls through the mobile broadband network or the Wi-Fi. In fact, when a call is performed, whatever were its results, a record will be stored in the main database msgstore.db but normally not in the messages table. Such records are saved in the table call_logs. As shown in Figure 27, WhatsApp stores (in the table) the following information about the calls: the call identifier in the field call_id, its type video_call, its direction (outgoing or incoming call) in from_me, its time in timestamps, its duration in duration, its size in bytes_transferred, its result in call_result (successful, missed or refused).

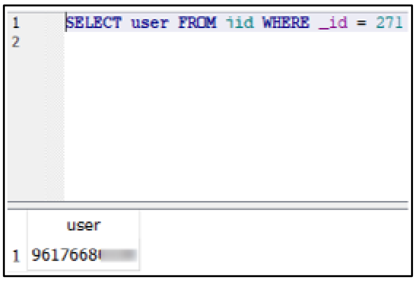
The identifier of the contact involved in the call is stored in the field 'jid_row_id', which can be identified by correlating this field to the 'fields _id' and user in the other table, jid. To be able to decode the number of the user involved easily, we use the following SQL query (Figure 26).

SELECT user FROM jid WHERE _id = *user_id_number*

An example is shown in Figure 27. Based on the experiments of Table 11, we have distinct records corresponding to the three possibilities presented in that table. By looking at the field from_me, its value is equal to 1 in the two records 929 and 930 which means that these two records correspond to outgoing calls from the user to the contact who have the jid_row_id = 271. Using the SQL query above, this contact has the number 9617668**** (Figure 26). The first call is a voice call because video_call = 0 and the second record is a video call since video_call = 1. The field call_result = 5 means that the two calls were successful, the voice call was established at 21st April 2019 1:45:30 AM (timestamps) and lasted for 49 seconds (duration). The second was done at 21st April 2019 1:17:34 AM and lasted 80 seconds. The next two records (_id 931 and 932) have the field from_me = 0, and call_result = 5, which means that these two calls were successful incoming calls. The next four records from _id 933 to 936 all have the field call_result = 4 and the duration = 0, which means that these four records correspond to missed calls. The analysis of the other fields shows the same value as above. The next record also shows that the call did not happen (duration = 0 sec), but the field call_result shows the value of 2, which means that the call was terminated by the user if the call is incoming (from_me = 0), or by the contact if the call is outgoing (from_me = 1). Table 11 resumes the chronology of the voice call logs presented in Figure 27.

Table 11. Reconstruction of the call history

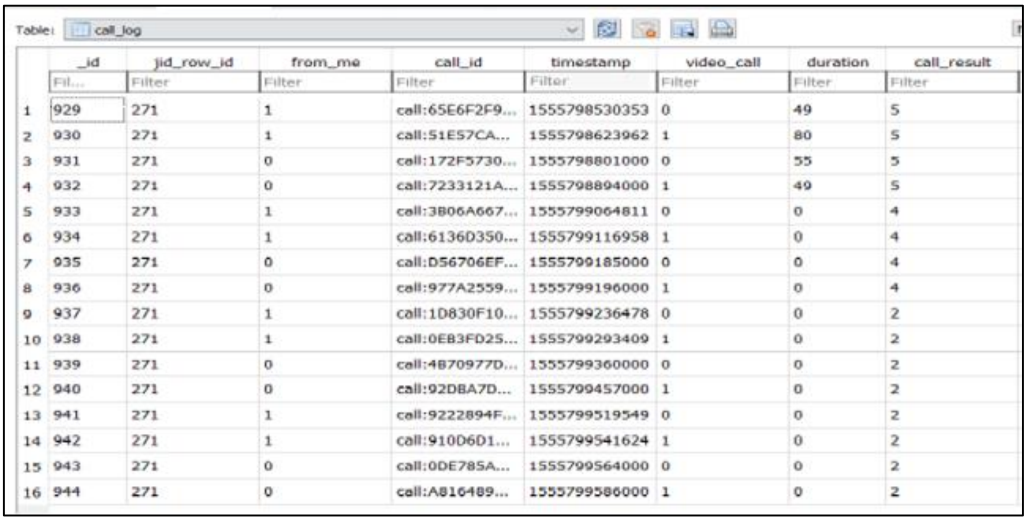
id	Video call	The caller	The contact number	Time of call	Duration (sec)	Status
929	0	The user	9617029****	1:17:30 AM	49	Success
931	0	The contact	9617668****	1:45:12 AM	55	Success
933	0	The user	9617029****	1:50:02 AM	0	Missed call
939	0	The contact	9617668****	1:51:45 AM	0	refused



```
SELECT user FROM jid WHERE _id = 271
```

```
  user
1 9617668[redacted]
```

Figure 26. User phone number identification



	_id	jid_row_id	from_me	call_id	timestamp	video_call	duration	call_result
	Fil...	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	929	271	1	call:65E6F2F9...	1555798530353	0	49	5
2	930	271	1	call:51E57CA...	1555798623962	1	80	5
3	931	271	0	call:172F5730...	1555798801000	0	55	5
4	932	271	0	call:7233121A...	1555798894000	1	49	5
5	933	271	1	call:3B06A667...	1555799064811	0	0	4
6	934	271	1	call:6136D350...	1555799116958	1	0	4
7	935	271	0	call:D56706EF...	1555799185000	0	0	4
8	936	271	0	call:977A2559...	1555799196000	1	0	4
9	937	271	1	call:1D830F10...	1555799236478	0	0	2
10	938	271	1	call:0EB3FD25...	1555799293409	1	0	2
11	939	271	0	call:4B70977D...	1555799360000	0	0	2
12	940	271	0	call:92DBA7D...	1555799457000	1	0	2
13	941	271	1	call:9222894F...	1555799519549	0	0	2
14	942	271	1	call:910D6D1...	1555799541624	1	0	2
15	943	271	0	call:0DE785A...	1555799564000	0	0	2
16	944	271	0	call:A816489...	1555799586000	1	0	2

Figure 27. call_log table

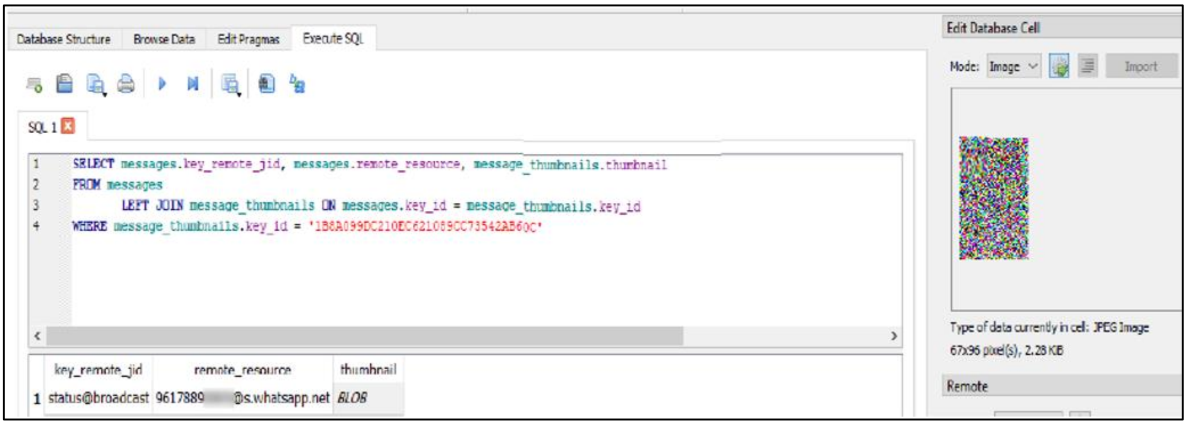
4.5. Status Analysis

WhatsApp has a feature which allows the users to publish a text or a media file that will disappear after 24 hours. This temporary data can contain evidence, as criminals might use it instead of texting in order to be sure that their communication would be deleted and then no evidence would be left behind. When the user posts a status, a record is generated in the table messages. This record is stored as status@broadcast under the field key_remote_jid. The phone number of the user sharing the status is stored in remote_resource. WhatsApp allows the users to share a status in two ways. The status can be a textual data, in this case this text is stored in the field data in the messages table, or a media file such as an image or a video, and in this case a thumbnail is stored in the table message_ thumbnail. To relate a status to its thumbnail in the other table, we can use this SQL query using the LEFT JOIN operation:

SELECT messages.key_remote_jid, messages.remote_resource, message_thumbnails.thumbnail FROM messages

LEFT JOIN message_thumbnails ON messages.key_id = message_thumbnails.key_id WHERE message_thumbnails.key_id= key_id of the status concerned.

Figure 28 illustrates the aforementioned query.



```
SELECT messages.key_remote_jid, messages.remote_resource, message_thumbnails.thumbnail
FROM messages
      LEFT JOIN message_thumbnails ON messages.key_id = message_thumbnails.key_id
WHERE message_thumbnails.key_id = '1B8A099DC210EC621089CC73542AB60C'
```

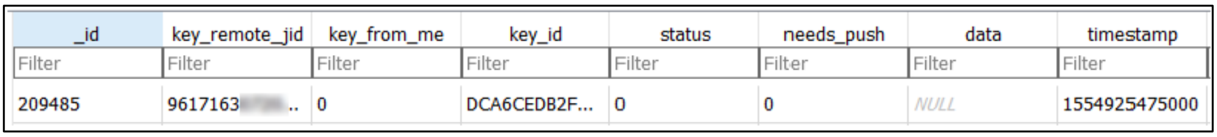
	key_remote_jid	remote_resource	thumbnail
1	status@broadcast	9617889[redacted]@s.whatsapp.net	BLOB

Type of data currently in cell: JPEG Image
67x96 pixel(s), 2.28 KB

Figure 28. Identification of the status picture using SQL query

4.6. Deleted Data

The analysis of deleted data is based on the structure of the SQLite database. If the WhatsApp messages are deleted, it is not possible to view them in the data table. We can see the value NULL as shown in Figure 29. However, according to the analysis of the SQLite database storage mechanism, we know that the actual messages may still exist in the database but only their page header information is erased. That is, the database will delete the first page header of deleted data and mark it as a free page, but its data area is not deleted. So, we can recover the deleted messages as long as the deleted data area has not been covered by other data. Specifically, it is possible to locate and extract deleted data according to the logical structure of the database file page. In fact, all the data in SQLite is stored on the page, and each page has its own corresponding file structure. There are different pages in the SQLite database. The data is stored in the B-tree pages. SQLite pages within a B-tree are classified as either internal or leaf pages. Internal pages contain pointers to other pages. Leaf pages contain the data. In general, each table has a root page that points to several leaf pages. This is illustrated in Figure 30.



_id	key_remote_jid	key_from_me	key_id	status	needs_push	data	timestamp
Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
209485	9617163[redacted]..	0	DCA6CEDB2F...	0	0	NULL	1554925475000

Figure 29. The value NULL indicates that a message was deleted

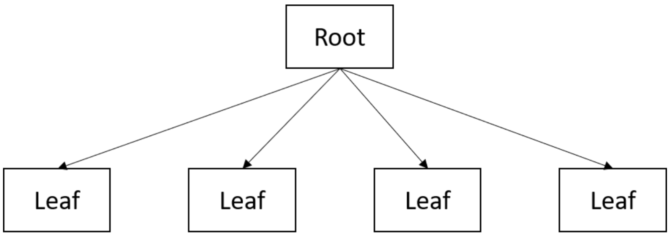
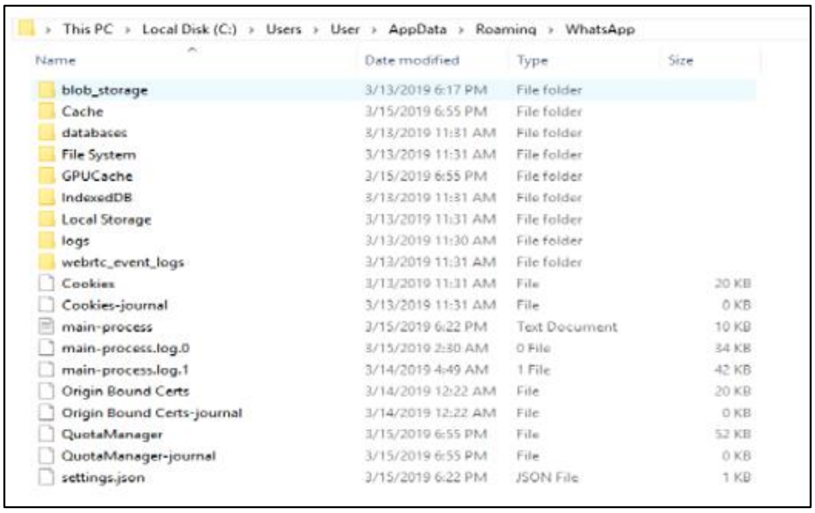


Figure 30. The structure of the B-tree pages where all data is stored

All data chat records are stored in the leaf pages, including the deleted messages in spaces called "unallocated space" (unallocated space is the space after the header information and before the first cell starts). Therefore, in order to find the deleted data, it is necessary to search the root page and use the navigation pointer to determine the leaf page. Finally, the deleted message can be extracted from the corresponding data area. Because of the expensive cost of the commercial forensic tools, we tried to write a Python script following this logic. We were able to reach these spaces but not generate data from them. More work is needed in this regard, specifically the analysis of the WAL journal of WhatsApp SQLite databases. Note that if the tables are vacuumed, there is no possibility to recover deleted data even with the best commercial tools. Fortunately, WhatsApp doesn't use an auto-vacuum index in their SQLite engine, so recovering is always possible.

4.7. Web WhatsApp and WhatsApp Windows Application

Like other popular Instant messaging apps such as Telegram and Viber, WhatsApp has both mobile and desktop applications. Also, WhatsApp can be accessed through a website "web.whatsApp.com". The analysis of this website shows that it does not leave any forensic artefacts on the suspect's computer. On the other hand, WhatsApp desktop application can be a good source of evidence that helps the investigation. The files created on the computer while using this application can be found in the following location in Windows 8 or 10: /Users/<User profile>/AppData/Roaming/WhatsApp (Figure 31).



This PC › Local Disk (C:) › Users › User › AppData › Roaming › WhatsApp

Name	Date modified	Type	Size
blob_storage	3/13/2019 6:17 PM	File folder	
Cache	3/15/2019 6:55 PM	File folder	
databases	3/13/2019 11:31 AM	File folder	
File System	3/13/2019 11:31 AM	File folder	
GPUCache	3/15/2019 6:55 PM	File folder	
IndexedDB	3/13/2019 11:31 AM	File folder	
Local Storage	3/13/2019 11:31 AM	File folder	
logs	3/13/2019 11:30 AM	File folder	
webrtc_event_logs	3/13/2019 11:31 AM	File folder	
Cookies	3/13/2019 11:31 AM	File	20 KB
Cookies-journal	3/13/2019 11:31 AM	File	0 KB
main-process	3/15/2019 6:22 PM	Text Document	10 KB
main-process.log.0	3/15/2019 2:30 AM	0 File	34 KB
main-process.log.1	3/14/2019 4:49 AM	1 File	42 KB
Origin Bound Certs	3/14/2019 12:22 AM	File	20 KB
Origin Bound Certs-journal	3/14/2019 12:22 AM	File	0 KB
QuotaManager	3/15/2019 6:55 PM	File	52 KB
QuotaManager-journal	3/15/2019 6:55 PM	File	0 KB
settings.json	3/15/2019 6:22 PM	JSON File	1 KB

Figure 31. Whatsapp desktop version files on the computer

The subdirectory "databases" contains SQLite file – Databases.db. But this file does not contain contacts or chats. Other subfolders contain temporary files of the WhatsApp desktop Application. Further investigation on these files has to be done.

5. Conclusion

This paper investigated the forensic artefacts of WhatsApp Messenger SQLite databases on Android phones. The methodology used was based on the performance of designed experiments on unrooted Android phones as well as a method to decrypt the encrypted databases by using free tools and Python scripts written for this study. Findings helped identify the artefacts left by the WhatsApp SQLite databases on Android phones, and the researchers have shown their evidentiary value. This research has demonstrated that it is possible to reconstruct the history of WhatsApp by analysing these artefacts. It was focused on how to analyze the data stored in the contact database in order to reconstruct the list of contacts of the user as well as the operations of adding and blocking contacts. Similarly, the researchers have discussed how to interpret the data stored in the chat database, aiming to reconstruct the status, the chronology, and the content of the textual and non-textual messages exchanged, the content of the feature called "status," and the communications that happen within groups, as well as the reconstruction of the chronology of the voice and video calls. Moreover, the paper has shown the importance of linking the information stored in the different tables of the database by using SQL queries in order to assume the coverage of all information that can be missed if each table is studied in isolation.

This research is considered innovative in such a way that forensic investigations in Lebanon or abroad may benefit from the methodology and results. No such work has been reported before, a fact that makes this research a new addition to the reservoir of knowledge needed to deter intruders and criminals who are active in practicing their malice on other people's phones. As this study was done completely using free tools as well as Python scripts, the researchers have not been able to recover the deleted data from these databases. They tried to write their own program using Python, and were able to access the spaces where the deleted data was stored in SQLite databases, but the main problem was how to generate this data in a readable format. This study focused on the SQLite databases on the Android phones, which could be considered a limitation. Nevertheless, such a limitation becomes a motivation for future work that should be extended to other operating systems such as iOS and Windows phones, where the storage and the artefacts generated might be different. As well, it is important to study the network of this application side by side with the file system analysis to provide the full picture to the analysts. Furthermore, the analysis of the WAL journal of the WhatsApp databases could be useful as a future work to recover deleted data from these databases.

6. List of Abbreviations

App:	Application	IT:	Information Technology
SQL:	Structured Query Language	IM:	Instant Messaging
SMS:	Short Message Service	OS:	Operating System
iOS:	iPhone Operating System	DB:	Database
AES:	Advanced Encryption Standard	SD:	Storage Device

7. Declarations

7.1. Author Contributions

Conceptualization, H.F.K. and S.K.M.; methodology, H.F.K and S.M.K.; software, S.M.K and A.H.; validation, H.F.K., S.M.K. and A.H.; investigation, H.F.K and S.M.K.; resources, H.F.K and S.M.K.; data curation, A.H.; writing—original draft preparation, S.M.K. and H.H.; writing—review and editing, H.H.; visualization, H.H and A.H.; supervision, H.F.K.; project administration, H.F.K.; All authors have read and agreed to the published version of the manuscript.

7.2. Data Availability Statement

The data presented in this study are available in article.

7.3. Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

7.4. Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

8. References

[1] Statista. (2021). Most popular global mobile messenger apps. Available online: https://www.statista.com/statistics/258749/ most-popular-global-mobile-messenger-apps/ (accessed on May 2021).

[2] Seigfried-Spellar, K. C., & Leshney, S. C. (2016). The intersection between social media, crime, and digital forensics: #WhoDunIt? Digital Forensics, 59–67. doi:10.1016/b978-0-12-804526-8.00004-6.

[3] Statista, (2021). Mobile operating systems' market share worldwide from January 2012 to January 2021. Available online: https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/#:~:text=Android maintained its position as,of the global market share (accessed on May 2021)..

[4] Skulkin, O., Tindall, D., & Tamma, R. (2018). Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques. Packt Publishing Ltd, Birmingham, United Kingdom.

[5] Epifani, M., & Stirparo, P. (2016). Learning iOS forensics. Packt Publishing Ltd, Birmingham, United Kingdom.

[6] Zhang, L., Yu, F., & Ji, Q. (2016). The Forensic Analysis of WeChat Message. 2016 Sixth International Conference on Instrumentation & Measurement, Computer, Communication and Control (IMCCC). doi:10.1109/imccc.2016.24.

[7] Anglano, C., Canonico, M., & Guazzone, M. (2016). Forensic analysis of the ChatSecure instant messaging application on android smartphones. Digital Investigation, 19, 44–59. doi:10.1016/j.diin.2016.10.001.

[8] Walnycky, D., Baggili, I., Marrington, A., Moore, J., & Breitinger, F. (2015). Network and device forensic analysis of Android social-messaging applications. Digital Investigation, 14, S77–S84. doi:10.1016/j.diin.2015.05.009.

[9] Ovens, K. M., & Morison, G. (2016). Forensic analysis of Kik messenger on iOS devices. Digital Investigation, 17, 40–52. doi:10.1016/j.diin.2016.04.001.

[10] Anglano, C., Canonico, M., & Guazzone, M. (2017). Forensic analysis of Telegram Messenger on Android smartphones. Digital Investigation, 23, 31–49. doi:10.1016/j.diin.2017.09.002.

[11] Zhang, H., Chen, L., & Liu, Q. (2018). Digital Forensic Analysis of Instant Messaging Applications on Android Smartphones. 2018 International Conference on Computing, Networking and Communications (ICNC). doi:10.1109/iccnc.2018.8390330.

[12] Rathi, K., Karabiyik, U., Aderibigbe, T., & Chi, H. (2018). Forensic analysis of encrypted instant messaging applications on Android. 2018 6th International Symposium on Digital Forensic and Security (ISDFS), 1-6. doi:10.1109/isdfs.2018.8355344.

[13] Azfar, A., Choo, K. K. R., & Liu, L. (2016). An android communication app forensic taxonomy. Journal of forensic sciences, 61(5), 1337-1350. doi: 10.1111/1556-4029.13164.

[14] Thakur, Neha S., (2013)."Forensic Analysis of WhatsApp on Android Smartphones". University of New Orleans Theses and Dissertations, Louisiana, United States.

[15] Mahajan, A., S. Dahiya, M., & P. Sanghvi, H. (2013). Forensic Analysis of Instant Messenger Applications on Android Devices. International Journal of Computer Applications, 68(8), 38–44. doi:10.5120/11602-6965.

[16] Anglano, C. (2014). Forensic analysis of WhatsApp Messenger on Android smartphones. Digital Investigation, 11(3), 201–213. doi:10.1016/j.diin.2014.04.003.

[17] Belkasoft (2018). Belkasoft Evidence Center v.8.6. Available online: https://belkasoft.com/whats_new_in_version_8_6. (accessed on December 2021).

[18] Oxygen Forensics (2021). Mobile forensic solutions: software and hardware. Available online: https://www.oxygen-forensic.com/en/ (accessed on May 2021).

[19] Sqlite Viewer. (2021). "SQLite Database Disk Image Is Malformed." Available online: https://sqliteviewer.com/blog/database-disk-image-malformed/ (accessed on August 2021).