TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technology
Department of Software Science

Kärte Pärend 192571IVCM

Forensic Traces of Messaging Applications on Android and iOS Mobile Phones

ITC70LT Master's Thesis

Technical Supervisor
Priit Lahesoo
MSc
Academic Supervisor
Sten Mäses
PhD

Tallinn 2021

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Kärte Pärend 192571IVCM

ANDROID JA IOS TELEFONIDE SUHTLUSRAKENDUSTE DIGITAALNE EKSPERTIIS

ITC70LT Magistritöö

Tehniline juhendaja
Priit Lahesoo
MSc
Akadeemiline juhendaja
Sten Mäses
PhD

Tallinn 2021

Author's declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author:	Kärte Pärend	.... (signature)
Date:	May 28, 2021	

Annotatsioon

Käesoleva töö eesmärgiks on uurida, kuhu ja milliseid jälgi jätavad suhtlusrakendused Android ja iOS mobiilides. Selleks kasutati Android ja iOS telefone, paigaldati kümme populaarset suhtlusrakendust ja saadeti telefonide vahel sõnumeid. Järgmisena valmistati telefonidest koopiad nelja erineva digitaalse ekspertiisi tööriistaga. Lõpuks analüüsiti tööriistade poolt genereeritud aruandeid ja koondati andmed tabelitesse, mida kasutati leitud jälgede kirjeldamisel.

Töö tegeleb uurimisprobleemiga, milleks on suhtlusrakenduste poolt nutitelefonidesse jäetavad jäljed. Antud töö ei kirjelda, mis juhtub failidega nende kustutamisel ja rakenduse kasutamise ajal. Töö on kvalitatiivne empiiriline uuring, milles uurimisstrateegiaks on kirjeldav juhtumiuuring.

Andmete kogumisel (dokumendid ning ekspertandmed ja vestlused praktikute ning akadeemiliste uurijatega) ja tõlgendamisel kasutatakse fenomenograafilist ning kriitilise diskursuseanalüüsi lähenemist.

Leiti, et telefonide digitaalse ekspertiisi koopiatest on võimalik leida sõnumeid, manuseid, jagatud asukohta ja muud infot. Osad suhtlusrakendused hoiavad enda andmebaase telefonides. Aruannetest ei olnud võimalik leida andmebaase kõikide rakenduste jaoks. Mitme rakenduse puhul leiti saadetud manuste ja muude rakendusega seotud meediafailide linke.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 53 leheküljel, 4 peatükki, 9 joonist, 35 tabelit.

Abstract

This work deals with a research problem about the traces that messaging applications leave on smartphones. The thesis aims to find out where and what artifacts different messaging applications leave on Android and iOS phones. For that, an Android and an iOS phone were used, ten popular messaging applications were installed on them, and messages were sent between phones. After that, forensic images were made with four different forensic tools used by law enforcement. Finally, the forensic reports were analysed, and data was gathered into tables used to describe the artifacts.

The following messaging applications were chosen for the analysis: Discord, Facebook Messenger, Kik Messenger, Signal, Skype, Slack, Telegram, Viber, WhatsApp, Wickr Me. The applications that left the most artifacts were Facebook Messenger, Kik, Telegram, Viber, and WhatsApp.

It was found that it is possible to find messages, attachments, shared location, and other types of data. Some messaging applications store some of their databases on the phone. It was not possible to find databases for all the apps from the forensic reports. Several applications had links to sent attachments or other media files related to the application. The results of this thesis can be useful for forensic examiners and people interested in smartphones.

The thesis is written in English and contains 53 pages of text, 4 chapters, 9 figures, 35 tables.

List of abbreviations and terms

ADB	Android Debug Bridge
APFS	Apple File System
API	Application Programming Interface
APK	Android Application Pack
ART	Android Runtime
ASCII	American Standard Code for Information Interchange [1]
AVD	Android Virtual Device
Cellebrite UFED	Cellebrite Universal Forensic Extraction Device [2]
CSV	Comma Separated Values [3]
DAT	File extension for a generic data file
DFU	Device Firmware Update
exFAT	Extended File Allocation Table [4]
EXT	Extended File System [4]
FBE	File-based Encryption [4]
FDE	Full-Disk Encryption [4]
F2FS	Flash Friendly File System [4]
GIF	Graphics Interchange Format
GUI	Graphical User Interface
Hex dump	Computer data in hexadecimal format
HFS Plus	Hierarchical File System Plus
HTML	Hypertext Markup Language
IMEI	International Mobile Equipment Identity
IPC	Inter-process Communication
JSON	Java Script Object Notation
JTAG	Joint Test Action Group
MSAB	Micro Systemation AB
MMS	Multimedia Messaging Service
NAND	Logical NOT AND
NIST	National Institute of Standards and Technology
NVM	Nonvolatile Memory
OHA	Open Handset Alliance

PDF	Portable Document Format
Plist	Property List
PTP	Picture Transfer Protocol
RAM	Random Access Memory
RFS	Robust File System
RJ45 cable	Registered jack cable [5]
SD Card	Secure Digital Card
SELinux	Security-Enhanced Linux
SIM	Subscriber Identity Module [6]
Sky ECC	Subscription-based end-to-end encrypted messaging application
SMS	Short Message Service
SQLite	Database engine
USB	Universal Serial Bus
TAR	TAR Compressed file archive [7]
TEE	Trusted Execution Environment [4]
UID	Unique User Identifier
UNIX	Uniplexed Information and Computing System
Wi-Fi	Wireless Fidelity
XLS	Microsoft Excel Spreadsheet [8]
XLSX	Microsoft Excel Spreadsheet
XML	Extensible Markup Language [9]
YAFFS2	Yet Another Flash File System 2 [4]

Table of Contents

List of Figures

List of Tables

1. Introduction

With the growing number of smartphones and smartphone users in the world, mobile forensics is becoming more important. The data stored on mobile devices can help to solve criminal cases. Crucial data sources are applications. The main purpose of this study is to give an overview of artifacts that messaging applications leave on smartphones. Different messaging apps store their files in different places on the phone. The purpose is to find out where. It also depends on the operating system where the files are stored.

The topic is important because apps change constantly, and there are not many research papers that include the most popular messaging applications. Also, smartphones can say a lot about peoples' lives, and when solving a crime, messaging application data on smartphones is a valuable source of evidence. There have been cases where mobile forensics has helped to solve the case. For example, the "Cyanide coffee" case where a woman was murdered with cyanide in her coffee and WhatsApp artifacts were used to solve the case [10]. Another example is a case where the police found the murderer, but after investigating his smartphone, they found that he had not acted alone. Mobile forensics has even helped to uncover a nationwide car theft operation [11].

Our research question is the following: where and what artifacts messaging applications leave on Android and iOS phones.

This thesis aims to find out where communication apps store their data on smartphones and describe what the found artifacts contain. This thesis does not discuss what happens when files are deleted in the app and what happens to files during the app's usage.

The apps used in this research are the following: WhatsApp, Facebook Messenger, Telegram, Viber, Signal, Discord, Wickr Me, Skype, Kik Messenger, Slack. These applications were chosen because they are popular on app stores (Google Play Store[1], App Store[2]) and also used in Europe.

This topic is important because it will be possible to get an overview of where instant messaging apps leave their traces on smartphones. These findings can help people working

[1]To be found at `https://play.google.com/store/apps/category/COMMUNICATION`
[2]To be found at `https://www.apple.com/app-store/`

in the forensics field and people interested in mobile forensics or Android and iOS phones.

For this study, help from the Estonian police was received. One of the supervisors is from the Estonian police. The devices used for this work were from their forensics laboratory, and the author made copies of the phones in their laboratory under supervision.

There are some confidentiality concerns related to this work because of the author's confidentiality agreement. The author can not share the forensic images made in the laboratory. Despite that, this document is accessible to all because publicly available information from the Internet was used mostly. There are also some open-source mobile forensic tools on the Internet that might give similar results. An account was created for each application on each phone to add data to applications, and messages were exchanged between the phones to create more traces.

The research gap that this research solves is that there are not enough up-to-date papers written on this topic. There also is not any documentation about where messaging applications leave their traces. This research contributes to the academic field by filling this gap. The research gap is discussed in chapter 3 about related work.

2. Background

The number of mobile phone users grows every year. According to statistics platform Statista[1] there were 3.6 billion smartphone users in the world in 2020, and by the year 2023 the number might increase to 4.3 billion [12]. Many activities that before were done using computers are now done on mobile devices (smartphones, tablets, etc.). For example, it is possible to make mobile payments. This means that in the forensics field, the importance of mobile forensics is increasing. Mobile devices might contain even more information about a person's everyday life than computers because they are portable, and most communication is done using messaging applications on smartphones. Furthermore, people use phones to take photos, track their health, and do other activities.

This chapter presents an overview of mobile forensics and introduces mobile operating systems and messaging applications used in the practical part. Section 2.1 is about mobile forensics, the following sections 2.2 and 2.3 are about mobile operating systems, and the last section 2.4 of this chapter is about messaging applications.

2.1 Overview of Mobile Forensics

Digital forensics is a branch of forensics sciences. This thesis uses the term digital forensics as "the retrieval, analysis, and use of digital evidence in a civil or criminal investigation" as defined by Hayes [13]. Digital forensics is not limited to computers only. "Any medium that can store digital files is a potential source of evidence for a computer forensics investigator" [13]. The devices and digital evidence used in investigations have to be handled in a forensically sound manner. "Forensically sound means that, during the acquisition of digital evidence and throughout the investigative process, the evidence must remain in its original state [13]." Computer forensics (digital forensics) is not the same as computer security. In computer security, the focus is on protecting computer systems and data, but computer forensics is focused on investigating digital evidence [13].

Mobile forensics is a branch of digital forensics. The field of mobile forensics is growing exponentially because the capabilities of mobile devices continue to expand [13]. Mobile

[1]To be found at
`https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/`

forensics includes the investigation of cellphones, tablets, personal media players, GPS devices, smartwatches and other mobile devices [13].

According to [4] the mobile forensic process can be divided into three main steps: seizure, acquisition and examination/analysis. During the seizure, the examiner has to handle the devices correctly. For example, if the phone is switched off, the phone should be put into a Faraday bag[2] to isolate it from the network and prevent changes to the data if the phone should turn on. If the phone is switched on, it should be disconnected from the network and put into the Faraday bag. In the acquisition step, the examiner might have to use multiple methods to extract data from the phone. In the examination/analysis stage, relevant data is separated from the rest of the data. In this stage, usually, the memory dump from the previous stage is imported to an examination tool that returns a report [4].

The digital evidence gathered and presented in the court has to meet requirements. The evidence has to be admissible, authentic, complete, reliable and believable. Admissibility refers to that the evidence "must be preserved and gathered in such a way that it could be used in court or elsewhere" [4]. For example, the evidence has to be gathered with legal tools. The evidence has to be authentic. It has to be relevant to the case and help to prove something. The evidence used in the court has to be complete because if something is missing, it might lead to a different judgment. "Presenting incomplete evidence is more dangerous than not providing any evidence at all." [4] Reliability means that it should be possible to reproduce the evidence using the same techniques. An exception might be the chip-off method that can be destructible. The evidence presented must be believable, and the forensic examiner must be capable of explaining what methods were used and how the integrity of the evidence was preserved. The evidence presented by the examiner has to be clear, understandable and believable to the jury [4].

There are also some good forensic practices that the examiner has to follow to ensure that the evidence meets the requirements. The examiner has to secure the evidence using the "right equipment and techniques to isolate the phone from all networks". Also, it is necessary to collect all the accessories (cables, power adapters, etc.) related to the device at the scene. To preserve the evidence, a copy of the disk has to be made, and the hashes of the original and the copy have to be compared. All further examination has to be done on the copy. Information about the requirements can be found in the code of criminal procedure. For instance, the Estonian Code of Criminal Procedure has paragraphs about the requirements that the evidence presented in the court has to meet [14].

[2]Faraday bag is a bag made from materials that block external static electrical fields. It is used to isolate devices and prevent wiping and tracking [4].

During the investigation, it is also necessary to document the evidence and changes. The notes have to be detailed enough so that another examiner can reproduce the result. The results have to be reported, and the report has to be a detailed summary of the steps taken and actions performed, the results, and interference during the investigation [4]. People working in the forensics field have to also follow the CIA model in their work. CIA triad, also known as confidentiality, integrity and availability is a model designed to guide information security policies in an organization [15]. Sometimes the model is also referred to as the AIC triad to avoid confusion with the Central Intelligence Agency [15].

2.1.1 Challenges in Mobile Forensics Field

The biggest challenge in mobile forensics is that the data is volatile and that data "can be accessed, stored, and synchronized across multiple devices [4]" and transformed or deleted remotely. Some messaging apps allow having the same account on multiple devices and synchronize everything immediately. For example, Telegram [16], Discord, Slack. This might also be a problem.

Forensic analyst has to prevent data alteration on the device and know which approach to use based on the device model because there is a wide range of different operating systems and device models [17]. Also, with newer phones the Faraday bag might not work so well. The examiner has to be knowledgeable of anti-forensic techniques, hardware differences, mobile malware and constant software updates because the criminal might have used anti-forensic techniques to hide data, or there might be malware on the device [4]. Anti-forensic techniques include hiding data, data obfuscation, data forgery and secure wiping [18]. Criminals can use legitimate apps to hide their activity [18]. For example, the cloud-based app Telegram has a Secret Chat function that uses end-to-end encryption and where the messages are only stored on the phone, not on the Cloud [18]. In addition, there might be legal issues when investigating a case. For example, if the crime took place in multiple countries, it is necessary to tackle the multi-jurisdictional issues [4]. Accessing locked or encrypted devices is also challenging [18].

During the extraction, technical problems might occur. For example, boot loop on Android phone. This means that the phone does not start and shows the start-up screen [4]. This problem might occur even when using the extraction tools. Fortunately, there are ways to fix it. If it is impossible to fix, it is necessary to flash the phone [4]. That means that everything on the phone is overwritten. This is the worst case because it is impossible to extract evidence from the phone.

Forensic examiners have to be aware of the inherent security features that the devices have.

Because privacy is becoming more important, device manufacturers are implementing robust security controls that make gaining data from devices harder. For example, if the device is password protected, the examiner has to find a way to bypass it. Also, full-disk encryption mechanisms prevent accessing the information on the device [17].

2.1.2 Tools

When examining phones, it is also necessary to use correct tools. For a forensic examiner, it is necessary to be familiar with multiple tools because one tool might not support all the devices or not have all the necessary functions [18]. There are many open-source and commercial tools for acquiring data from phones. For example, one Android forensics tool is a command-line tool ADB (Android Debug Bridge) [4]. It enables to extract all files from the device through a USB (Universal Bus Cable) cable [19]. Some commercial tools are Cellebrite UFED, MSAB XRY Logical, MOBILedit Forensic Express, Magnet ACQUIRE.

According to a research report [20], there are no court-approved mobile forensic tools or standard forensic tool certifications applicable to all legal systems in the world. However, in many countries, the government has institutions that can test the tools [20]. For example, in the United States, there is the National Institute of Standards and Technology (NIST) that regularly tests and provides assessments of the tools [20].

When choosing a tool, there are several things to consider: accessing data, decoding data, data integrity, training the users. The tool has to decode data well to make the data easily readable for the forensic examiner. Data integrity is important because, in many courts, it is necessary to prove the origin and reliability of the evidence. This is called the “Chain of Custody” or “Chain of Evidence”. For example, the tool can provide reliable evidence when it stores the evidence in a different file format. If the evidence is stored in open file format, someone can accidentally drop some other files into the folder. The training of forensic examiners is very important because the examiner has to be ready to explain how the tools work and show that he is qualified to work with the tools [21].

There have also been some research papers comparing different mobile forensic tools. These papers might help to choose a tool. For example, a paper [22] by Saleem *et al.* published in 2012 compared Cellebrite UFED and MSAB XRY. Unfortunately, the tools and mobile devices are constantly changing, and papers written multiple years ago are no longer useful. This kind of work should be done regularly, for example, every two or three years, if forensic examiners find that it might be useful for them.

2.1.3 Mobile Forensic Tool Leveling System

Mobile forensics tools can be classified by the method that is used. The tool classification system developed by Sam Brothers can be seen in Figure 1. The techniques that require less effort are lower in the pyramid, and the methods that require more effort are higher in the pyramid. Each technique has advantages and disadvantages. To prevent modifying or losing the evidence, the examiner has to be aware of these issues before using a tool and applying a technique [4].

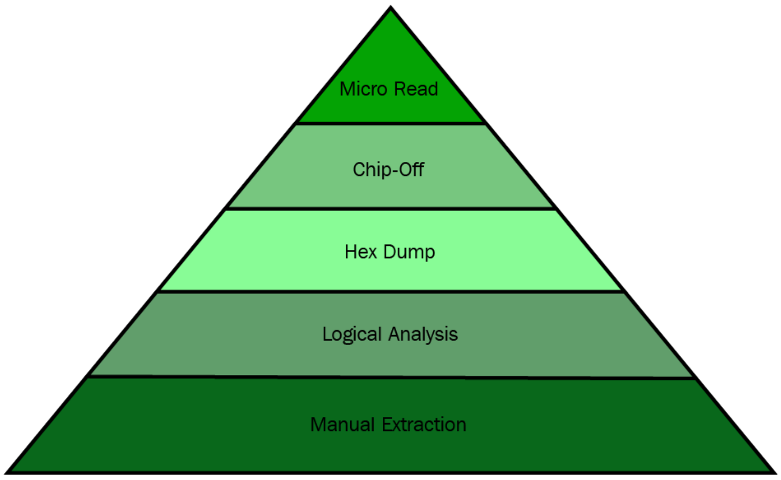


Figure 1. *"Cellular phone tool leveling pyramid [4]".*

Manual Extraction

Manual extraction is the easiest method for the examiner. The examiner has to look through all the content and take photos. However, there can also be some problems. For example, if the user interface is in a language that the examiner does not understand. If the screen is broken, the examiner has to use other methods. Manual extraction is time consuming and has a greater risk for human error [4]. The examiner might modify, overwrite or delete data accidentally. Also, the examiner might miss some important data if he is not familiar with the phone's operating system. It is not possible to access deleted data [4].

Logical Extraction

For logical extraction, the examiner has to use tools to extract data. There are many commercial and open-source tools available. Usually, these tools can also produce a human-readable report. These reports might be quite long because the tool has examined all the applications on the device [4].

For logical extraction, the examiner has to connect the phones to the computer using wired (USB cable, RJ45 cable, also called registered jack cable [5]) or wireless (Wi-Fi or wireless fidelity [23] in other words, Bluetooth, infrared) communication. The computer sends commands to the phone, the phone answers, and a report is made. The extraction process is fast, and it does not require much training. The disadvantages are that the process might write data to the phone, or some data might be modified, and deleted data is not usually accessible with this method [4].

Hex Dumping

One physical extraction technique is hex dumping. Hex dump is computer data in hexadecimal format [24]. To create a hex dump, the examiner connects the phone to a forensic workstation that instructs the phone to dump memory to a computer. The result is in binary format and has to be translated to a readable format. One advantage of this method is that the process is inexpensive. It is also possible to get more data from the phone and it is possible to recover deleted files from unallocated space [4].

Chip-Off

Chip-off data acquisition refers to data acquisition from the memory chip. Usually, the chip is removed from the device and placed to a chip reader or another phone that extracts the data. The disadvantages are that it is a more technically challenging technique because different chip types are used in phones. It is expensive, and the examiner has to have some hardware-level knowledge. The result is in raw format and needs parsing, decoding, and interpreting. It is recommended to use other extraction techniques before chip-off because the chip-off technique can be destructive. The advantage of chip-off is that it preserves the state of the memory exactly as it was, and it is possible to use this method when the phone is damaged, but the chip is intact [4].

Often the Joint Test Action Group (JTAG) [4] method is used to read the chip [4]. JTAG is a method for verifying and testing circuit boards. This method forces the chip to transfer the raw data that is stored there. This method is often used with devices that are inaccessible with standard tools. Chip-off method works even when the phone is screen-locked [4].

Micro Read

In the micro-read method, the examiner views and interprets the data on the memory chip manually. An electron microscope is used to analyze the chip, and then the information is translated to 0s and 1s and then to ASCII characters. The disadvantages are that the process is time-consuming, costly, and requires extensive knowledge. It is rarely used. It is only used for high-profile cases when all the other methods have been used before. There are also no commercial tools for micro-read currently [4].

2.1.4 Data Extraction Methods

There exist three data acquisition methods for mobile phones: manual, logical, and physical acquisition. Some of the methods might overlap with some levels of the tool classification pyramid. The manual acquisition involves looking through the content on the phone's memory and taking photos of the content. Logical acquisition is the extraction of files and directories on the file system. Physical acquisition is a bit-by-bit copy of the physical storage. Level one of the pyramid is manual extraction, level two is logical extraction, and levels three to five are physical extraction methods [4].

Logical and physical acquisition methods have differences. Logical data extraction is easier, less time-consuming, does not return all the information. Physical extraction is difficult, time-consuming, gives almost all the information, including deleted data, and it might be possible to access unallocated space [4].

Physical acquisition is the most thorough option because it is possible to acquire all the data, including deleted data, and access unallocated space. However, in practice, the logical acquisition is used the most. The manual acquisition is the last option because it involves using the user interface to look the phone through and there is a greater risk for human error [4]. Sometimes when physical acquisition is used, it is good to make a logical image also because it can help to find data from the physical image.

Data from the phone can be extracted from several places: SIM card, external storage card, and phone memory. The SIM card identifies a user on a cellular network. It also contains IMSI (International Mobile Subscriber Identity) which is an internationally unique number for identifying the user on the cellular network [13]. The first three digits of the IMSI are MCC (Mobile Country Code), the following two to three digits are MNC (Mobile Network Code), and the final part is MSIN (Mobile Subscriber Identity Number), which identifies the subscriber on the network [13]. The SIM card also has ICCID (Integrated Circuit Card ID), which is a 19- to 20-digit serial number and physically located on the card [13]. The primary functions of a SIM card are to identify the subscriber to a cellular network and store data [13]. External storage usually refers to an SD (Secure Digital) card. Nowadays, phone internal memories are quite large (64 GB, 128 GB, 256 GB, etc.), so that an SD card is always not used in the phone.

Mobile forensic tools mainly extract data from memory. Some examples of what can be extracted are the following: phone address book, call history, SMS (Short Message Service) and MMS (Multimedia Messaging Service) messages, e-mails, web browser history, photos, videos, music, documents, calendar, network communications, maps, social networking data, deleted data [4]. Besides data extraction, it is also possible to receive communication-related data from the phone service provider.

To learn more about an application or access an application that is locked, it is also possible to reverse engineer the application. Reverse engineering is “the process of retrieving source code from an executable” [4]. With reverse engineering, it is possible “to understand the functionality of the app, the data storage, the security mechanisms in place, and more” [4]. For reverse engineering an Android application, it is necessary to extract the application APK file at first. It is possible to extract the APK file both on rooted and unrooted phones. On Android phones preinstalled apps have their APK file in /system/app folder and third-party apps in /data/app folder [4]. Reverse engineering is out of the scope of this work.

2.1.5 Types of Evidence

The range of evidence extracted from mobile phones is different from what can be extracted from laptops and desktop computers [13]. The primary difference is that mobile phones have SMS, MMS, and RCS messages. SMS is a text message service. These messages can be found from the memory of the device or a SIM card. By checking the status flag, it is possible to determine whether a message has been read, deleted, has not been read, sent, or unsent [13]. MMS (Multimedia Messaging Service) is a messaging service that enables to send multimedia content. With forensics tools, it is possible to extract multimedia content out of the user’s messages. RCS (Rich Communication Services) is a messaging standard that aims to broaden the capabilities of SMS. It does not support end-to-end encryption [13]. Besides SMS, MMS, RCS messages, it is possible to extract different data related to applications.

The physical evidence package usually includes a mobile phone with the phone model name, model number, serial number, IMEI number(s), a SIM card. The SIM card has the following information: the carrier, SIM serial number (ICCID), IMSI.

The logical evidence package includes the content of the phone, SIM card(s), and applications on the phone. The phone’s content is the phonebook, user accounts on the phone, call history (canceled, dialed, unanswered, answered calls), SMS and MMS (sent and received) messages. The content of the applications is, for instance, the content of Slack, Telegram, Viber, WhatsApp, Wickr Me, SureSpot, e-Takso, Swedbank, etc.

Besides mobile forensic tools, data acquisition methods, and types of evidence a forensic examiner needs to know about different mobile operating systems. For example, how the file system looks like. In the following two sections, an overview of two well-known mobile operating systems is given. These operating systems are Android and iOS.

2.2 Android Operating System

Android is an open-source operating system that is based on the Linux kernel [13]. Google acquired it in 2005, and it is maintained by the Open Handset Alliance (OHA) [13]. OHA is a group of telecom companies, mobile phone manufacturers, semiconductor and software companies [13]. The first Android version 1.0 was officially launched in 2008 [4]. In 2014 Android 5.0, named Lollipop, was released. In this version, the user interface was redesigned, and the design language Material Design was used. Dalvik virtual machine was replaced with Android Runtime (ART) [4], and Android became available on TVs and in cars for the first time. Android operating system can be found on smartphones, tablets, home appliances and other devices [13] and it is the leading OS for smartphones [4].

Android architecture can be seen in Figure 2. Each layer performs its own operations and provides services to the layer that is on top of it. There are five layers. The lowest layer is called the Linux kernel layer because Android is built on top of the Linux kernel. This layer is an abstraction layer between the software and hardware. It contains drivers that translate hardware instructions into software instructions. The layer on top of the Linux kernel layer is the hardware abstraction layer which has several library modules for different hardware components. The next layer consists of Android libraries, which are written in C or C++. This layer also has Android Runtime, responsible for running applications on Android devices [4].

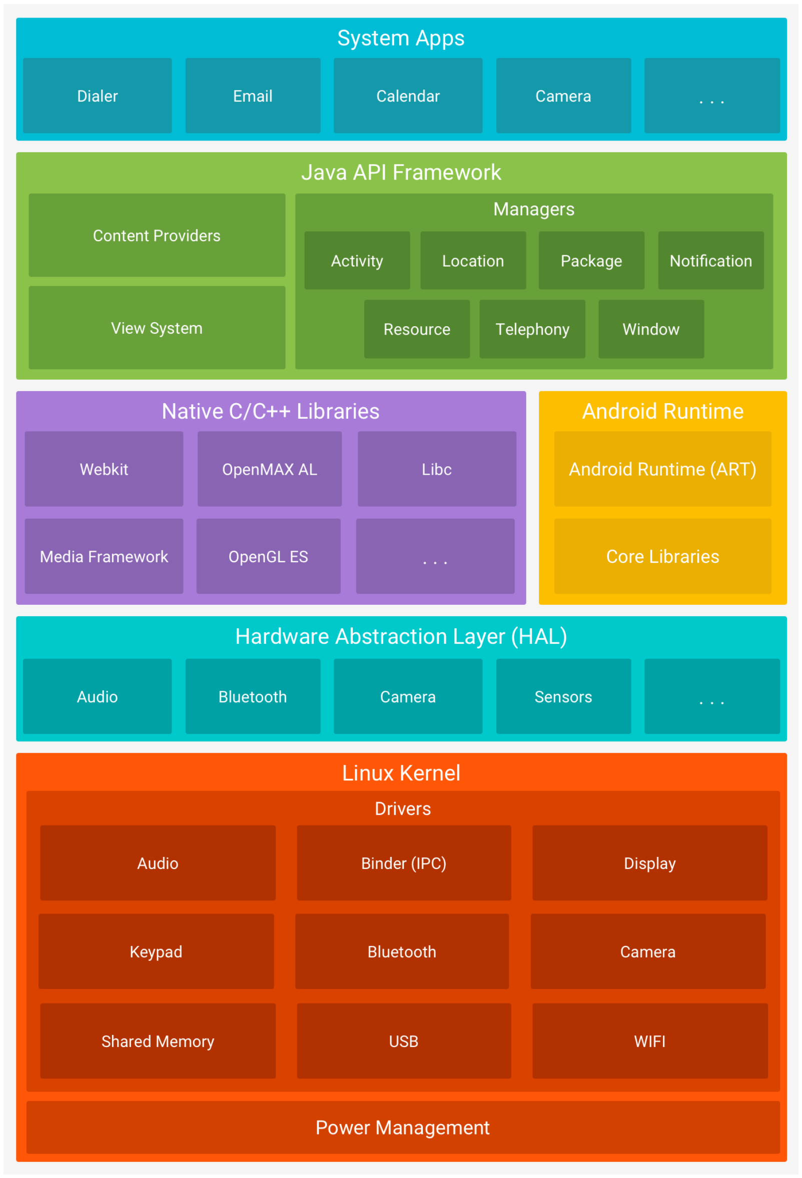


Figure 2. *Android architecture [25].*

The next layer is the Java API (Application Programming Interface) [26] framework layer responsible for the basic functioning of a phone. Through this layer, the applications installed on the phone can communicate with the phone. The topmost layer is the system apps layer. In this layer, the user can directly interact with the device. The applications that users use can be divided into preinstalled and user-installed applications. Everything that the user sees on the phone, for example, contacts, cameras, mail, is an application [4].

2.2.1 Security Features

Android has several security features. Some of them came from the Linux kernel. For example, a permissions model, isolation of running applications, and secure inter-process communication (IPC) [4]. The permission model is very useful because the user can choose which permissions to give to applications and the user knows what permissions the application has. The application prompts the user the first time the permission is needed when the app is in use [13]. Permissions are stored in AndroidManifest.xml file [17]. On Android, permissions are divided into four categories: normal, dangerous, signature, signature/system. Normal permissions are the default. “These are low risk permissions and do not pose a risk to other applications, system, or user [17].” These permissions are granted during installation automatically without asking for approval. For dangerous permissions, user approval is necessary because these permissions can cause harm to the system and other apps. Signature permissions “are automatically granted to a requesting app if that app is signed by the same certificate as the one that declared/created the permission” [17]. Signature permissions exist to allow apps that are related to data sharing. Signature/System permissions are permissions “that the system grants only to applications in the Android system image or that are signed with the same certificate as the application that declared the permission” [17].

Isolation of running processes (application sandboxing) means that every application is given a unique user identifier (UID), which prevents one application from accessing the data of other applications. Secure IPC makes sending information between activities inside one application or between applications more secure [4].

A security feature that did not come with the Linux kernel is application signing. It means that all installed applications have to be digitally signed with a certificate. Developers can upload their applications to Google Play Store only if they have signed the application. The developer of the app holds the private key of the certificate. When using the private signing key the developer can provide future updates to the application [4], [17].

Other security features in Android are Security-Enhanced Linux (SELinux) [4], file-based encryption (FBE), Android Keystore, TEE (Trusted Execution Environment) [4], and verified boot [4]. SELinux is used to enforce mandatory access control (MAC) which ensures that applications (applications running as root or superuser included) work in isolated environments [17]. This also prevents malware from accessing the OS and corrupting the device [17]. SELinux uses the principle of *default denial* which means that if something is not explicitly allowed, it is denied [17]. SELinux has two global modes: permissive mode and enforcing mode. In permissive mode, the permission denials are logged but not enforced. In enforcing mode the permission denials are logged and enforced [17].

2.2.2 File System

For analyzing the data extracted from an Android phone, it is necessary to understand Android's file system. This understanding helps the examiner to narrow down the search. Knowledge about Unix-like systems is very useful because Android uses Linux kernel. UNIX means Uniplexed Information and Computing System. Linux file hierarchy is a single tree with a root directory (marked as /) at the top. "The Android file hierarchy is a customized version of this existing Linux hierarchy [4]." This hierarchy might have some differences based on the underlying Linux version.

Android has several partitions. "Partitions are logical storage units made inside the device's persistent storage memory [17]." Partitioning allows to logically divide the available space into sections [17]. Some examples of different partitions on Android are the /boot partition, /system partition and /data partition.

The /boot partition contains files and information that are necessary to boot the phone. It has the kernel and RAM disk. The /system partition contains system-related files. For example, the Android framework, libraries, system binaries, and pre-installed applications [17]. This partition is needed so that the device could boot into normal mode and it should be never deleted [17], [4]. The /recovery partition is for backup purposes, and it allows the device to boot into the recovery console [17].

The /data partition contains the data of each application. Data storage is the place where most of the forensic evidence will reside [17]. The /cache folder contains frequently accessed data, some of the logs, and update packages downloaded using Wi-Fi [17], [4]. This folder is important for the forensic examiner because data that might not exist in the /data folder anymore might still be in the /cache folder. The /misc folder has information about miscellaneous settings. For instance, the state of the device, hardware settings and USB settings. The /sdcard folder contains all the files and folders that are on the SD card [4].

Android has mount points like Linux. Filesystems in Android can be divided into flash memory filesystems, media-based filesystems, and pseudo filesystems. Each filesystem has a different file retrieval speed, security, size, etc. "Flash memory is a type of constantly powered nonvolatile memory (NVM) that retains data in the absence of a power supply [4]." The flash memory units are called blocks, and flash memory can be reprogrammed or erased in blocks. Common flash memory filesystems are: exFAT, F2FS, YAFFS2, RFS (Robust File System). Media-based filesystems are EXT, EXT2, EXT3, and EXT4. The advantage of EXT3 is journaling which means that in case of an unexpected shutdown there is no need to verify the filesystem. Pseudo filesystems are a logical grouping of files. They are not real files. Some important pseudo filesystems are root filesystem (rootfs) for booting the device and sysfs filesystem, which contains information about the device's configuration. The data in pseudo filesystems is mostly related to configuration. For forensic examiners, the most important filesystems are those that store user's data. Nowadays, forensic tools are capable of showing the filesystems in a graphical user interface (GUI) screen [4].

2.2.3 Application Data Storage

Apps can be categorized as system and user-installed apps. However, they can also be categorized as apps that come along with Android, apps installed by the manufacturer, apps installed by the wireless carrier, apps installed by the user [17]. All these apps store different types of data on the smartphone. Application data can be stored on the device internally or externally. In the case of external storage, data is stored on the SD card, and it can be stored in any location on the card. External storage can also be non-removable storage that comes with the phone. In the case of internal storage, the data is stored in predefined folders. The internal data of all apps is saved in /data/data in a subdirectory named after the application package (APK) [17]. For example, if the package name of Discord is com.discord, then the data is stored in /data/data/com.discord folder.

The data that belongs to applications can be stored in the following locations: shared preferences, internal storage, external storage, SQLite database, network. Application preferences files are mostly in DAT or XML format [4]. DAT is a file extension for generic data files [27]. Shared preferences files are in XML format. XML files contain key-value pairs of primitive data types. Primitive data types are Boolean, float, int, long, and string [17]. "Strings are stored in the Unicode Transformation Format (UTF) format [17]." Shared preferences files are usually stored in an application's /data/data/shared_pref path [17]. Preference files are used to keep track of user and application preferences [28].

The folders that are created for most applications are: /cache, /databases, /files, /lib, /shared_prefs. "Folders other than these are custom folders created by the app developer [17]." Table 1 has descriptions for these main folders.

Table 1. Application sub directory descriptions [17].

Sub directory	**Description**
/cache	Files cached by app
/databases	SQLite and journal files
/files	Developer saved files

Continues...

Table 1 – *Continues...*

Sub directory	**Description**
/lib	Custom library files required by an app
/shared_prefs	XML files of shared preferences

The /cache directory contains files cached by the app. The database files are stored in /databases. On Android phones, application files are mostly stored in SQLite databases. SQLite database is a relational database and it is usually the preferred storage for mobile app related data [13]. A freeware tool that can be used to read these database files is SQLite Database Browser. The /lib folder contains custom library files required by the app. The content of /shared_prefs was discussed above.

2.2.4 Extracting Data from Android Devices

Extracting data from initial Android versions (until Android 5.0 [29]) was much easier because Android did not have full-disk encryption (FDE) and file-based encryption (FBE) mechanisms which enable to store the data in an encrypted format within the device. Android versions 5.0 until 9.0 have full-disk encryption and Android versions 7.0 and later have file-based encryption [29]. Also, applications did not have so many security features on different operating systems [4]. On Android phones, the examiner might also encounter a boot loop problem. Another problem might be phones that are modified to make them hard to track. Android is an open-source operating system, and anyone can modify the source code and distribute it [30].

Secure messaging applications might also cause problems to forensic examiners—for example, Sky ECC, a subscription-based end-to-end encrypted messaging application. The company Sky Global who sold smartphones with the application was shut down by judicial institutions in March 2021 because there was evidence that criminals used the application and that the company's management might have allowed it. The smartphones "had their cameras, microphones and GPS capabilities removed to make them harder to track" [31]. Sky ECC messages were automatically deleted after 30 seconds, and if the user entered the password "panic", all the content of the device was erased [32]. Sky ECC had approximately 170 000 users [33]. Even after the company was shut down, the usage of the phones might continue, or there might appear new companies providing similar secure communication services. Another example was the company Phantom secure that provided encryption services and devices to criminal organizations [34].

Sometimes it might be necessary to bypass the lock screen on an Android phone. The examiner should never try to guess the code because most devices have “a setting that will wipe the device after several failed attempts” [17]. There are other methods for bypassing the lock screen, but it all depends on the OS version, device settings, and the capabilities of the examiner. There is no universal solution that will work every time on every device. Some commercial forensic tools have the capabilities to bypass the lock. For example, Cellebrite and Oxygen [17].

Forensic examiners have to know about different ways how it is possible to modify the device. Sometimes examiners might encounter rooted Android phones or have to root the phone by themselves for forensic examination. Rooting means gaining the root privileges (highest privileges) on the phone. The root user can change/delete any file, change privileges of other users, change system applications and settings, etc. [35]. There are two types of root accounts: permanent root and temporary root. Most public tools result in the permanent root, which means that root privileges will persist even after rebooting the device. In case of temporary root, the changes are lost once the device is rebooted. For forensic cases, temporary roots should be preferred. A phone should be rooted only when it is necessary. Rooting has several disadvantages. Rooting makes the phone more vulnerable. For instance, a malicious app might access the entire operating system and the data on the phone. Another disadvantage is that there is a possibility to brick the device if rooting is not done properly. Bricking is a word that is used for phones that can not be turned on in any way. Rooting may also void the warranty [17].

Before rooting a phone, the forensic examiner must understand boot loader, recovery, and fast boot modes in Android [17]. “An Android phone can be seen as a device having three main partitions: boot loader, Android ROM, and recovery [17].” The boot loader is in the first partition. It is the first program that runs when the device is turned on. The boot loader’s job is to “take care of low-level hardware initialization, and boot into other partitions” [17]. The boot loader loads the Android ROM partition, also referred to as the Android partition. Android ROM contains files that are needed to run the device. The recovery partition (stock recovery) is used to delete all user data and files or update the system. When the user does a factory reset on the phone, the recovery partition boots up and erases all the data. When the user installs an official update, the phone boots into recovery mode and installs the latest updates that are written on the Android ROM partition. A recovery environment created by a third party is called custom recovery. It can be used to replace the default recovery, also called as stock recovery. Custom recovery has more options than stock recovery [17].

Fastboot mode is also one of the modes that a forensic examiner should know about.

Fastboot is a protocol. It can be used to re-flash partitions on the device. Fastboot is a tool that is included in the Android SDK and is an alternative to the recovery mode, for instance, for installations, updates and in some cases unlocking the boot loader [17].

“Boot loaders may be locked or unlocked [17].” Locked boot loaders do not allow the user to modify the device’s firmware. Factory data reset is performed on the phone when unlocking a locked boot loader. Gaining root access on a device with an unlocked boot loader is easier [17].

Before using mobile forensic tools on an Android phone, it is necessary to put the phone into developer mode and turn USB debugging on from the settings [17]. “Enabling USB debugging allows a computer to communicate with an Android device via a USB cable [13].”

2.3 iOS Operating System

Even though Android is the most popular mobile operating system for smartphones in the world, iOS is the most popular mobile operating system in the United States and Japan [36]. iOS is also the leading OS for tablets [4]. The first iPhone using iOS was released in 2007 [4].

iOS architecture has four layers that can be seen in Figure 3. The core OS is the lowest layer and deals with low-level functionalities. The Core Services layer provides system services that are necessary for the applications. The media layer provides the graphics, audio, and video frameworks. Cocoa Touch layer contains frameworks that are necessary to develop user-interface for iOS applications [4].

iOS phones have HFS Plus (Hierarchical File System Plus) [37] and APFS (Apple File System) [38] filesystems. HFS filesystem was developed in 1996 to store large datasets. To overcome some limitations of the HFS filesystem, they made an HFS Plus filesystem designed to support larger file sizes. HFS Plus uses journaling which means logging every transaction to the disk. Logging helps to prevent disk corruption. The APFS filesystem was introduced in 2016 as the replacement for HFS Plus. With the release of iOS 10.3, it became the default iOS filesystem [4]. The filesystem is divided into “two logical disk partitions: the system (root or firmware) partition and the user data partition” [4]. “The system partition contains the OS and all of the preloaded applications used with the iPhone. The system partition is mounted as read-only unless an OS upgrade is in progress or the device is jailbroken [4].” The system partition takes a small portion of storage space. “The user data partition contains all the user-created data, ranging from music and contacts to

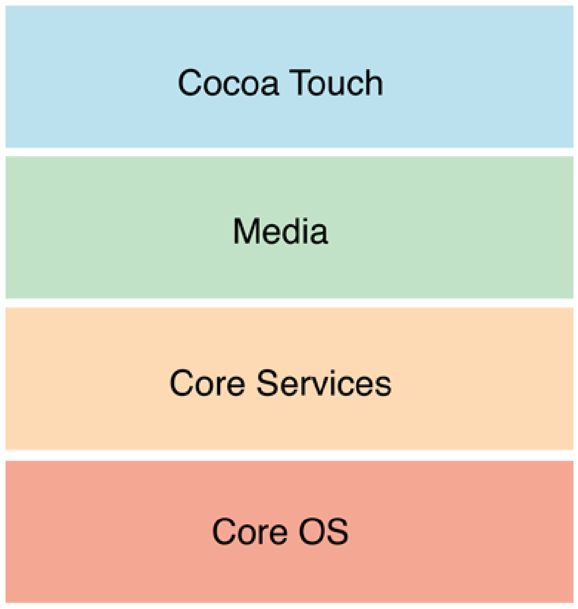


Figure 3. *iOS architecture [4].*

third-party application data. The user data partition occupies most of the NAND (logical NOT AND) memory and is mounted to the /private/var directory on the device [4].” When extracting data the user data partition should be saved as TAR file [4]. TAR is a compressed file archive [7].

iOS also has security features like Android. Some examples of these are passcodes, Touch ID, Face ID, code signing, sandboxing, encryption, privilege separation. Starting with the iPhone 4 the entire filesystem is encrypted with a filesystem key [4].

On Apple phones, application data is mostly stored in SQLite and Plist files. Property list file “is a structured data format used to store, organize, and access various types of data on an iOS device as well as a macOS device” [4]. Sometimes JSON (JavaScript Object Notation) [39] files are also used [4].

2.3.1 Extracting Data from iOS Devices

Like Android device has USB debugging mode, iOS devices also have different operating modes. It might be necessary to put the device into one of the modes depending on the extraction method or forensic tool. The different modes are normal mode, recovery mode, and Device Firmware Update (DFU) mode [4].

If the iPhone has password protection, then it is necessary to bypass it to extract data. Unlocking an iPhone that runs a newer iOS version (iOS 8 or newer) or unlocking a newer iPhone (starting from iPhone 6) is difficult and might not work in all cases. One possibility to unlock the device is to use lockdown files [4]. This “only works if the device was unlocked with a passcode at least once after the last reboot” [4]. There are also more advanced techniques. For example, “fingerprint molds to trick Touch ID, masks to trick Face ID, and NAND mirroring to bypass passcode entry limits” [4].

With the logical acquisition, it is possible to capture everything that an iTunes backup contains. This way, it is not possible to get any deleted files, and it is necessary to use other methods to recover SQLite data and deleted artifacts [4]. With the logical acquisition, it is possible to test if the iOS device is locked. Most tools will fail if the device is locked.

For filesystem acquisition, iPhone has to be jailbroken to remove some restrictions. Jailbreaking is like rooting an Android phone. Jailbreaking voids the user’s warranty and might brick the device. The filesystem image is stored in the TAR archive and can be opened with an archiver. With filesystem acquisition, it is possible to get the majority of data from the iOS device [4].

2.4 Messaging Applications

When analyzing application data, it is also useful to know more about the applications—for example, their features and how they store data. In the beginning of 2021 there were approximately 240 applications available in the free communication app category on the Google Play store. Some of these apps are extensions of apps. According to Google Play store the most popular messaging applications that are also used in Europe are WhatsApp[3], Facebook Messenger[4], Telegram[5], Viber[6], Signal[7], Discord[8], Wickr Me[9], Skype[10], Kik Messenger[11], Slack[12]. The table 3 in the appendix has comparison of different messaging applications. Most applications can fully function using Wi-Fi network. They do not require a data plan for use [4].

Discord is a messaging application that is meant for groups and communities. It was initially made for people playing computer games. It has over 100 million monthly active users [40]. Some interesting features of Discord are bots, streaming, and screen sharing

[3]To be found at `https://www.whatsapp.com/`
[4]To be found at `https://www.messenger.com/`
[5]To be found at `https://telegram.org/`
[6]To be found at `https://www.viber.com/en/`
[7]To be found at `https://www.signal.org/`
[8]To be found at `https://discord.com/`
[9]To be found at `https://wickr.com/me/`
[10]To be found at `https://www.skype.com/en/`
[11]To be found at `https://www.kik.com/`
[12]To be found at `https://slack.com/intl/en-ee/`

[41]. Discord does not use end-to-end encryption. It uses only standard encryption [42] and stores its messages in Discord's servers as can be seen from Table 3.

Facebook Messenger was made by Facebook. Besides Messenger application, they offer messaging services to businesses. Messenger does not use end-to-end encryption, but it has a secret conversation mode that has to be turned on in the settings to use end-to-end encryption. Secret conversations do not have all the features that are available in the standard conversation mode. Secret conversations do not support the following: group messages, GIFs (Graphics Interchange Format) [43], voice or video calls, payments. Furthermore, there is no possibility to report a secret conversation [44]. For end-to-end encryption Messenger uses Signal Protocol [45]. Facebook stores all the messages of a user in their server and a backup of the messages on the user's phone [46].

Kik was founded in 2009 by a group of University of Waterloo students [47]. Kik does not use end-to-end encryption. Kik messages are temporarily stored in their servers until the messages are delivered to the user's device [48].

Signal is a messaging application that uses end-to-end encryption [49]. Signal uses Signal Protocol, and Signal messages are stored locally on the user's device [50].

Skype is a peer-to-peer (P2P) communication app [13]. It uses Wi-Fi connection to facilitate free video, voice, and instant messaging (IM) [13]. To allow file transferring to other Skype contacts and fee-based voice calls to landline phones and cellular phones it uses VoIP [13]. Skype has approximately 300 million active monthly users worldwide [13]. Skype uses encryption, but end-to-end encryption has to be turned on [51]. Voice messages are encrypted when they are delivered to the device, but when the user opens them, they are stored on the user's device as an unencrypted file [52].

"Instant messages (IM), between the Skype and chat service in the Cloud, are encrypted using TLS (transport-level security) [13]." For message encryption between two Skype users, the app is using AES (Advanced Encryption Standard). Voice messages are also encrypted when they are sent [13]. When "the voice message is downloaded and listened to, it is stored on the client's computer in an unencrypted way" [13]. The calls are also encrypted [13].

Slack is a platform for team chats. "Over 750,000 companies use Slack to get work done [53]." Data is stored in Slack's virtual private cloud (VPC), and "all data transmitted between Slack clients and the Slack service is done so using strong encryption protocols" (cloud security principles) [53].

According to the website of Telegram, Telegram has 500 million monthly active users and is one of the ten most downloaded apps in the world. Telegram uses end-to-end encrypted voice and video calls, as well as voice chats in groups. One-to-one chats are not end-to-end encrypted. The messages are stored securely in a Telegram cloud. Like Messenger, Telegram also has secret conversation mode where all messages in secret chats use end-to-end encryption. The difference from Messenger's secret conversation is that in Telegram when messages are deleted on one side of the conversation, they have to be deleted on the other side also. Secret chats are device-specific, and when the user logs out from the device, all the secret chats are lost [54].

Viber is a calling and messaging app. It uses end-to-end encryption. Viber has self-destructing messages feature [55]. Viber uses end-to-end encryption, and it stores messages on the server only temporarily until the messages are delivered [56].

According to the website of WhatsApp, 2 billion people in over 180 countries use it. "WhatsApp is free and offers simple, secure, reliable messaging and calling, available on phones all over the world [57]." WhatsApp was founded by Jan Koum and Brian Acton in 2009, and in 2014 it joined with Facebook but remained as a separate app. It started as an alternative to SMS, but now it supports different media types. WhatsApp messages are end-to-end encrypted. For end-to-end- encryption, they use Signal Protocol. End-to-end encryption ensures that only people who communicate to each other can read or listen what is sent. Messages sent with WhatsApp are not stored in WhatsApp servers after they are delivered [58]. They are stored on the user's device [57], [59].

Wickr Me uses 256-bit authenticated end-to-end encryption. It uses Wickr Secure Messaging Protocol [60]. Wickr Me does not store messages on their servers [61].

Some of these applications store their users' data on the cloud to access their data from different devices. Apps that store their users' data (messages, information) in cloud are WhatsApp, Telegram, Viber, Skype [54], [62].

Many applications are advertised as secure applications, but there might be vulnerabilities. It might still be possible to find more information from the app even though it is said to be secure. For example, app journals, write-ahead logs, or shared memory files might include unencrypted data that should not be there, or users might have taken screenshots of their chats. It might also happen that files that were supposed to be encrypted are not. Secure messaging applications are, for example, Telegram, Wickr Me, and Signal. What can be found about messaging applications from the smartphone depends on the model, the OS running on the smartphone, and the version of the app [4].

Some applications that claim that they encrypt data only encode it. The most commonly used encoding for smartphones is base64. Encoding means transforming the data into a different format so that it is possible to decode it. However, for encryption, a key is used to transform data into a different format. The key is confidential, and it is possible to decrypt data only using the key. Some mobile forensic tools are capable of decoding data that has been encoded [4].

3. Related Work

In this chapter an overview of several scientific articles and books that have been written on mobile forensics and messaging applications is given. Also, the research gap is discussed.

Two journal articles were found that have information about 20 or 30 messaging applications. Despite analyzing many applications, these research papers did not include all the messaging applications used in this study. Secondly, these papers are outdated because mobile forensic tools and mobile applications evolve constantly. The paper analyzing 20 applications was published in 2015 and the other paper in 2016.

20 messaging applications

In the first journal article [63] by Walnycky *et al.*, 20 popular instant messaging applications for Android are analyzed. "This work shows which features of these instant messaging applications leave evidentiary traces allowing for suspect data to be reconstructed or partially reconstructed and whether network forensics or device forensics permits the reconstruction of that activity [63]." Twenty messaging applications from the Google Play store were selected based on keywords "chat", "chatting", "date", "dating", "message", and "messaging" and the number of downloads. Two devices were used: a smartphone running on Android and an iPad. For each application, they created two accounts on the devices. Their target device was the Android phone. iPad was used to exchange messages with the target device. They also created a wireless access point using a Windows 7 computer. Both the phone and the iPad were connected with the wireless access point. Before the examination, they installed all the messaging applications in their list to both devices. They sent messages between the devices. "The content of each message sent with each application was different [63]." The sent messages were pictures, videos, and plain text. Wireshark was used to capture and save the network traffic between two devices [63].

After the network analysis they performed logical acquisition of Android device using Micro Systemation's XRY. "A logical acquisition of a mobile device, much like a logical acquisition of a computer hard drive, misses deleted files and other remnants of data which might otherwise be found in unallocated space [63]." From the logical image they managed to find database files where these applications store data and chat logs in plaintext. They examined mainly chat logs and text data in unencrypted database files. They validated their results with Helium backup. All the activity traces that were found were documented [63].

The result was that Walnycky *et al.* managed to "reconstruct some or the entire message content from 16 of the 20 applications tested" [63]. They concluded that this reflects poorly on the security and privacy measures that these applications have, but it is useful for evidence collection purposes [63]. In most cases, they were able to reconstruct the following data: passwords, screenshots taken by applications, pictures, videos, audio files sent, messages sent, sketches, profile pictures, and more [63]. Walnycky *et al.* mention how, in the majority of cases, the messages or parts of the messages can be easily recovered by network sniffing, or forensic examination [63].

30 messaging applications

In the second journal article [64] by Azfar *et al.* 30 messaging applications were analysed. XRY was used to make logical forensic images. Their experimental setup used a Wi-Fi channel and two Google Nexus 4 phones with Android version 5.0.1. For application artifact analysis, a Windows 7 computer was used. For each application, they conducted an individual set of experiments. "After one set of experiments was concluded, the phone was wiped before installing the next app [64]." The results were summarized in a table. The table has the directory and file location for the data found [64].

Azfar *et al.* categorized the communication apps into three categories: Instant messaging (IM) apps, Voice over IP (VoIP) apps, and Augmentative and Alternative Communication (AAC) apps. IM apps provide real-time text transmission. Some of these apps allow to create a friends list, and some of these create the list from phonebook contacts. VoIP applications help to communicate easier and in a more affordable way to people who are located all over the world. Most VoIP apps have video call capability. AAC apps are meant for people with impairments. These apps can help express thoughts, needs, wants, and ideas using spoken, or written language [64].

The artifacts are categorized into four categories. These are User and contact information, Exchanged messages, Timestamps, and Others. The User and contact information category contains artifacts related to the user and the user's contacts. For example, phone numbers, contact ID, contact status messages. The Exchanged messages category has, for instance, the type of the message. An example of an artifact in the Timestamps group is the time when add request was sent. The artifacts considered in the Others group are extracted databases, members of a group chat, the duration of voice calls [64].

The taxonomy shows various data that can be recovered when examining different communication apps. The app categories are represented in one dimension and the artifact types in other. Their findings help forensic examiners to reconstruct the chronology of exchanged messages. Azfar *et al.* mention that their findings were accurate at the time of the research, but new releases of messaging apps may change how the data is stored on devices and the type of data that can be recovered. Their suggestions for future work were to improve the taxonomy by examining other apps and new releases of apps. They also recommended including the deletion/uninstallation of apps to determine if any artifacts are left on the device [64].

Other articles/books

There were also research papers that analyzed one, two, three, four or five messaging applications. In most papers, the phone that was used had an Android operating system. There were only a few papers that had both Android and iOS analyses. In some of them, the researchers used commercial tools, and in some of them, they used open-source tools.

Rathi *et al.* [65] analysed the following messaging applications: Viber, Telegram, WhatsApp, WeChat. The paper's main goal was to "analyze the most popular applications' encrypted data storage locations in Android devices" [65]. They also showed "how these applications store data in the Android file system" [65] and discussed encryption. They installed the applications on rooted and unrooted Android phones with different versions and populated the phones with test data. From tools, they used ADB and some other open-source tools. They stored the locations of artifacts in tables. They also discussed the challenges that they faced when collecting forensically important artifacts [65].

In the paper [66] by Anglano *et al.* the researchers "present a methodology for the forensic analysis of the artifacts generated on Android smartphones by Telegram Messenger" [66]. They also "show how to reconstruct the list of contacts, the chronology, and contents of the messages that have been exchanged by users, as well as the contents of files that have been sent or received" [66], and "the log of the voice calls made or received by the user" [66]. Anglano *et al.* say that their methodology can be applied to any application running on an Android device.

The experiments described in the paper by Anglano *et al.* were carried out on a virtualized Android device. To create virtualized smartphones Android Virtual Devices (AVD), they used the Android Mobile Device Emulator. Three different AVD configurations with different Android versions were used. For extraction, they used the Cellebrite UFED4PC platform. They validated their results by doing a subset of experiments on a real smartphone. With this method, it was possible to recover all the artifacts that were left by Telegram messenger. In the article, they said that “the results collected from this smartphone were identical to those obtained from the virtualized smartphones” [66].

A journal article [67] by Anglano *et al.* analyses the artifacts left on Android smartphones by WhatsApp. Like in the previous article, they also used software-emulated Android devices instead of real smartphones and validated their results on real smartphones. As a virtualization platform, they used YouWave. Only Android version 4.0.4 was used. Tools used in the work were: FTK Imager to extract the files, SqliteMan to read database files, Notepad++ to read textfiles. They show how to interpret the data stored in the contacts and chat databases and how to reconstruct the list of contacts, and the chronology of messages [67].

Iqbal *et al.* conducted an analysis of the ChatON Instant Messaging application [68]. The researchers used both Android and iOS phones for the experiments. They performed a detailed analysis of ChatON communication app. They used SQLite Database Browser to investigate database files and Plutil for Plist files [68].

Another article using both phones is “Investigating Social Networking applications on smartphones detecting Facebook, Twitter, LinkedIn and Google+ artifacts on Android and iOS platforms” [35] by Dezfouli *et al.*. In this paper, researchers examine four popular applications on Android and iOS phones. Examined apps are Facebook, Twitter, LinkedIn and Google+. Dezfouli *et al.* present a variety of artifacts that can be useful in a criminal investigation [35].

In a paper by Sgaras *et al.* [69] forensic acquisition and analysis of four instant messaging applications and VoIPs is presented for both Android and iOS platforms. The authors compare the forensic analysis of the following applications: WhatsApp, Skype, Viber, and Tango. They show the types of artifacts that can be found [69].

In the article “Forensic Acquisitions of WhatsApp Data on Popular Mobile Platforms” [70] researchers Shortall and Azhar used Android, iOS, and Windows phone and analyzed WhatsApp. They present a summary of what artifacts, for example, login, username, password, name, location, about each application were found [70].

There are also some articles where researchers have used only iOS phones. For example, “Forensic analysis of Kik messenger on iOS devices” [71] by Kenneth M. Ovens and Gordon Morison, where a detailed description of artifacts created by Kik on iOS phones is provided. They also showed that deleted images in Kik are recoverable and can be located and downloaded from Kik servers [71].

An article about Kik messenger on Android was also found. In [72] Al-Rawashdeh *et al.* present an investigation of Kik Messenger. The authors did a post-mortem analysis of the memory dumps. The "experimentation results show that sent and received messages and contact artifacts can be found in both memories even after the chat session is deleted, messages are individually deleted, contacts are deleted, app is closed, and even after the phone is locked [72]."

In "Network Forensics Analysis of iOS Social Networking and Messaging Apps" [73] the researchers Bhatt *et al.* analyzed 70 iOS apps, and 20 of them were messaging apps. They were able to trace and reconstruct at least one of the following in about 15 apps out of 20: entire message content, user's location, email or social networking credentials, profile images, or tweeted messages. They also investigated the network traffic of 50 applications to determine how the end user's data is shared over the network. A lot of information was shared in unencrypted form [73].

"Mobile forensics: analysis of the messaging application Signal" [74] by Samantha M. Judge is a Master's thesis that researched data recovery. The forensic tools used in the research were UFED 4P and UFED Physical Analyzer. The author also compared the results to the results of three open-source tools. The open-source tools used were iPhone Analyzer, iExplorer, and Autopsy. Four mobile devices were used in the experiments—two Android devices and two iOS devices. Only one device produced viable results about the Signal application. The other three devices produced minimal results without real message data [74].

The paper "Forensic investigations of popular ephemeral messaging applications on Android and iOS platforms" [75] by Azhar *et al.* reports mobile forensic investigations of ephemeral messaging applications, including Signal and Wire. Both proprietary and freeware tools were used. The analysis was done on a rooted Android phone and an iOS phone which was not jailbroken [75].

In [75] by Azhar *et al.*, various information was found from both phones. For example, from the Android phone, they found evidence of communications and media files in a cache folder. From the iOS device, they found account information, contacts, evidence of communications. No full ephemeral messages were found from the iOS phone. On the Android phone, full ephemeral communications were found only for Wire applications. One interesting finding was Snapchat's 'offensive words' detection, which may help to find evidence about inappropriate language usage and help to solve cyberbullying cases. The results showed that it was possible to get more information from the rooted Android phone. They also mentioned that physical acquisition might have gotten more data from iOS phone [75].

“Forensic analysis of secure ephemeral messaging applications on android platforms” [76] by M. A. Hannan Bin Azhar and Thomas E. A. Barton analyses Wickr Me and Telegram. The authors tried to recover the artifacts and then compared them to reveal the differences between the applications. An Android phone was used for the messaging applications. The artifacts were created by using the secure features of these applications. Used secure features were, for example, ephemeral messaging, the channel function, and encrypted conversations. The experiments helped to understand how secure messaging applications store data. Results showed that Wickr Me stored received messages in encrypted “.wic” files. For Telegram, the tools had built-in modules for recovering encrypted artifacts. The authors documented the artifacts and added additional information. A RAM dump technique helped to recover some plaintext artifacts [76].

The book [4] by Tamma *et al.* focuses on teaching the latest forensic techniques that help to investigate different mobile device platforms. For example, it describes forensic techniques for Android 8 to Android 10 and iOS 11 to iOS 13. It also gives an overview of Android, iOS, Windows 10 mobile operating systems, and the latest open source and commercial mobile forensic tools. Finally, the book guides the reader on how to examine third-party applications such as Facebook and WhatsApp. From the apps analyzed in this work, the book has WhatsApp and Skype analysis [4].

The book [17] by Skulkin *et al.* gives an overview of the Android platform and its architecture, how data is stored on Android and what Android forensics entails. The book describes how to recover deleted data and forensically analyze the application data using open source and commercial tools. The reader will get an overview of various physical and logical techniques and learn how to recover deleted data. It also describes third-party application analysis and analyses the following text-messaging applications: Facebook Messenger, Skype, Snapchat, Viber, Tango, WhatsApp, Kik, WeChat. Despite having the analysis of Facebook Messenger, Skype, Viber, WhatsApp, and Kik, it is missing Discord, Signal, Slack, Telegram, and Wickr Me. In the concluding chapters, the reader will also learn about malware analysis [17].

Even though there exist papers written on this topic, the papers are outdated because applications and mobile forensic tools change. The found papers do not include all the applications discussed in this work. The books do not have data about all of the applications discussed in this work.

4. Methodology

This is a qualitative empirical research where comparative case study [77] is used as the research strategy. Primary data is used in the research because the data was collected by the author [78]. Phenomenographic [79] and critical discourse analysis [80] were used for data collection (documents, expert data and conversations with practical and academical researchers).

The theoretical part of this work is based mainly on the discussion of forensic examiners, which is observed by an IT specialist with no previous experience with the forensics field. The messaging application data was investigated using an experimental setup where data was extracted from two smartphones. This chapter describes the methodology used for the experiments.

Forensic examiners have to ensure that they extract data from phones without changing the evidence. The book "Handbook of Research on Digital Crime, Cyberspace Security, and Information Assurance" [81], says the following about digital evidence: "Digital evidence is said to be forensically sound if it was collected, analyzed, handled and stored in a manner that is acceptable by the law, and there is reasonable evidence to prove so. Forensic soundness gives reasonable assurance that digital evidence was not corrupted or destroyed during investigative processes, whether on purpose or by accident [82]." To ensure that forensic images used in this work are forensically sound, the images were made in the laboratory of the Estonian National Criminal Police under the supervision of a forensic examiner and using tools that forensic examiners use in the law enforcement field all over the world.

Usually, it is necessary to take into account legal matters. Each country has laws (National Law, legal norms) about collecting the evidence and how the evidence should be presented in the court. In this thesis, forensic images are used only for research purposes.

In the digital forensics field, the process must be documented (chain of custody) and repeatable by other forensic examiners. This chapter describes the extraction process to ensure that the process would be repeatable.

4.1 Experimental Setup

Two smartphones were used for the experiments. Information about smartphones and other tools that were used can be found in Table 4 in the appendix. More specific information about smartphones is in Table 5. The table contains the software version, model name, model number, serial number, capacity, IMEI (International Mobile Equipment Identity) [83], carrier, and if the phone was rooted or unrooted. Android version 10 was used for the Android phone because, according to statistics from Statista, Android 9 was the most used Android version in 2020, and presumably, Android 10 will become the most used in the near future [84]. Even though the most used iOS version used in 2020 was iOS13 that had 83% of the iOS market, iOS12 was used because iOS13 does not support iPhone6 [85]. Furthermore, an HP laptop was used to read the reports that were made by the forensic tools.

SIM cards were inserted into both phones, the phones were connected to Wi-Fi, and messaging applications were installed. Then a small dataset was created and uploaded to both phones. The dataset was necessary so that the phone would have some data that can be sent between apps. The dataset had few image files, a video file, a sound file, and a pdf (Portable Document Format) file. These are the most shared file types on messaging applications. Next, an account was created on each phone and each installed messaging application. Each account was given a profile image. Also, all possible permissions were given to the applications. For example, the access to contacts, camera, microphone and photos.

After each application was set up, random messages were sent on each application between these two phones. The message types were mainly text messages, images, videos, sound files, pdf files. Also, the location, recorded voice, or video messages were shared and other possible functionalities, for example, liking messages, forwarding messages, pinning messages, were used where possible. When all/almost all different possibilities were used on each application, the data extraction started. The data was extracted with mobile forensic tools from the following companies: Cellebrite, Micro Systemation AB (MSAB), Compelson Labs, Magnet Forensics. All these companies provide tools for Law Enforcement, but some of them also provide tools to enterprises. All the tools are described in the next section.

4.2 Mobile Forensic Tools and Their Capabilities

Mobile forensic tools are quite easy to use. They provide instructions, and everything is visually easy to understand. The user interface of Cellebrite UFED can be seen in Figure 4.

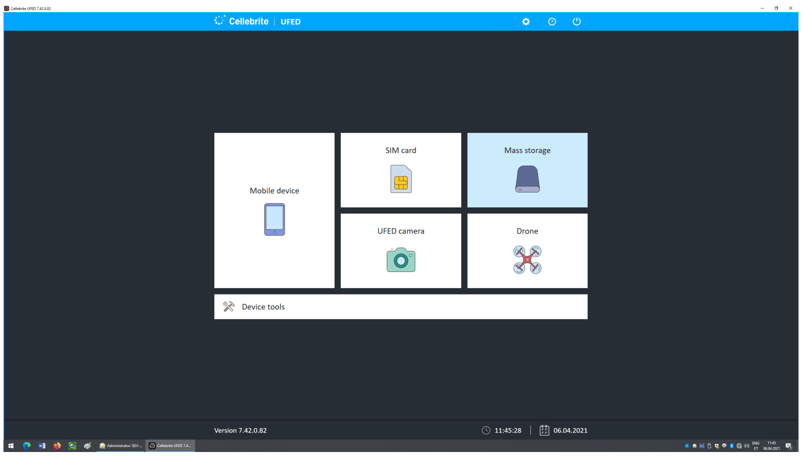


Figure 4. *Cellebrite UFED user interface.*

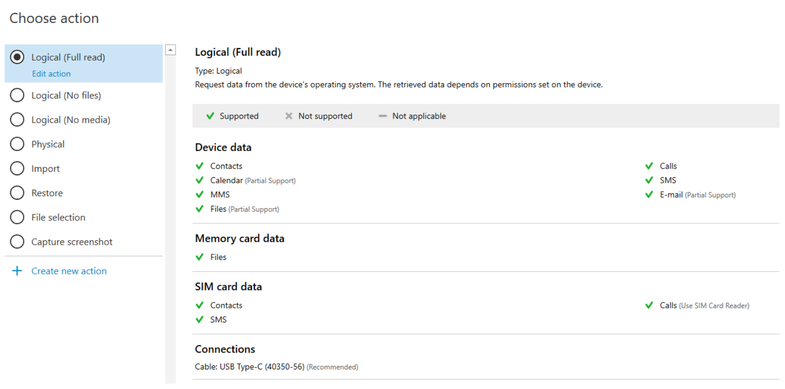
Mobile Forensic tools have similar capabilities. With Cellebrite UFED[1] it is possible "to bypass locks, perform advanced unlocks, perform logical/full file system/physical extractions, perform selective extraction of apps data and cloud tokens and much more" [86]. Cellebrite UFED is capable of making reports in the following formats: CSV, HTML, PDF, Threads, XLS, XML [87]. Cellebrite Physical Analyser is used to analyze UFD files generated by Cellebrite UFED and create reports in UFDR, pdf, and other file formats [88]. Cellebrite Reader[2] is used to read reports in UFDR file format. Cellebrite Reader is a free tool, and anyone can use it [89].

XRY Logical[3] enables to perform Logical data extraction [90]. An example of the capabilities of XRY for Android S9+ phones can be seen in Figure 5. For iPhone 6 XRY has fewer options. With this tool, it is possible to make reports in XLS file format.

XAMN Viewer[4] is a free tool that is used to read XRY reports [91]. XAMN Viewer artifact view can be seen in Figure 6.

MOBILedit [5] is also capable of doing logical extraction. It is capable of generating HTML, pdf, and Excel reports and also UFDR reports. HTML reports are useful because it is possible to save them on a disk. For iPhone, it is necessary to make an iTunes backup

[1]To be found at `https://www.cellebrite.com/en/ufed/`
[2]To be found at `https://www.cellebrite.com/en/reader/`
[3]To be found at `https://www.msab.com/products/xry/xry-logical/`
[4]To be found at `https://www.msab.com/products/xamn/viewer/`
[5]To be found at `https://www.mobiledit.com/forensic-express`


Choose action

- [✓ Logical (Full read)] Edit action
- Logical (No files)
- Logical (No media)
- Physical
- Import
- Restore
- File selection
- Capture screenshot
- Create new action

Logical (Full read)

Type: Logical

Request data from the device's operating system. The retrieved data depends on permissions set on the device.

✓ Supported ✕ Not supported — Not applicable

Device data

✓ Contacts
✓ Calendar (Partial Support)
✓ MMS
✓ Files (Partial Support)
✓ Calls
✓ SMS
✓ E-mail (Partial Support)

Memory card data

✓ Files

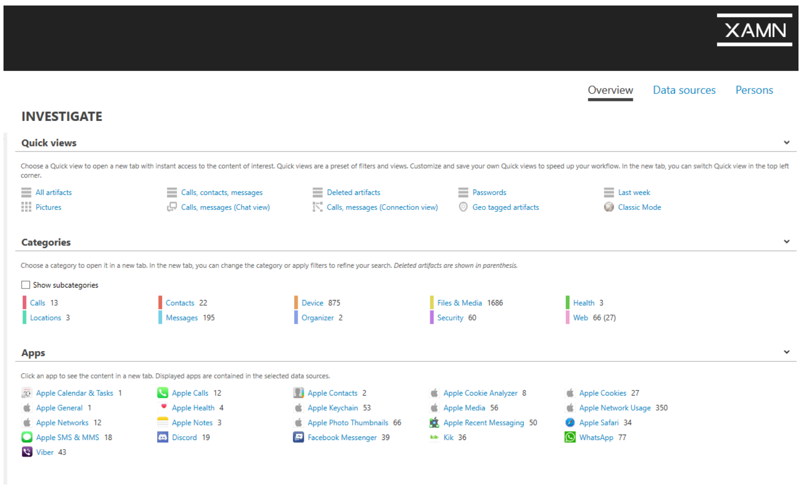
SIM card data

✓ Contacts
✓ SMS
✓ Calls (Use SIM Card Reader)

Connections

Cable: USB Type-C (40350-56) (Recommended)

Figure 5. *XRY capabilities.*



INVESTIGATE

Quick views

Choose a Quick view to open a new tab with instant access to the content of interest. Quick views are a preset of filters and views. Customize and save your own Quick views to speed up your workflow. In the new tab, you can switch Quick view in the top left corner.

All artifacts	Calls, contacts, messages	Deleted artifacts	Passwords	Last week
Pictures	Calls, messages (Chat view)	Calls, messages (Connection view)	Geo tagged artifacts	Classic Mode

Categories

Choose a category to open it in a new tab. In the new tab, you can change the category or apply filters to refine your search. *Deleted artifacts are shown in parenthesis.*

☐ Show subcategories

Calls 13	Contacts 22	Device 875	Files & Media 1686	Health 3
Locations 3	Messages 195	Organizer 2	Security 60	Web 66 (27)

Apps

Click an app to see the content in a new tab. Displayed apps are contained in the selected data sources.

Apple Calendar & Tasks 1	Apple Calls 12	Apple Contacts 2	Apple Cookie Analyzer 8	Apple Cookies 27
Apple General 1	Apple Health 4	Apple Keychain 53	Apple Media 56	Apple Network Usage 350
Apple Networks 12	Apple Notes 3	Apple Photo Thumbnails 66	Apple Recent Messaging 50	Apple Safari 34
Apple SMS & MMS 18	Discord 19	Facebook Messenger 39	Kik 36	WhatsApp 77
Viber 43				

Figure 6. *XAMN Viewer.*

before using the tool because the tool uses the backup for extraction. With Android, it can extract data by itself [92], [93].

Magnet AXIOM[6] has tools Magnet Acquire[7], AXIOM Process, and AXIOM Examine. Magnet Acquire is a forensic tool by Magnet Forensics for acquiring forensic images of any iOS or Android device, hard drive, and removable media [94]. AXIOM Process processes the image and enables the creation of a report. AXIOM Examine is used for reading these reports.

Magnet AXIOM by Magnet Forensics can be used both for phones and computers [4]. It is capable of doing both logical and filesystem acquisitions [4]. It enables to analyse extraction data while extracting the data so that it is not necessary to wait until the extraction process has ended [4].

[6]To be found at `https://www.magnetforensics.com/products/magnet-axiom/`
[7]To be found at `https://www.magnetforensics.com/resources/magnet-acquire/`

Magnet AXIOM is capable of recovering data from the Cloud [95]. AXIOM also has a GrayKey integration. GrayKey is a tool that helps to recover data from phones. [96]

4.3 Data Extraction and Reporting

All the data extraction was done in the Digital Forensics Laboratory of the Estonian National Criminal Police West Prefecture. They provided phones, cables, and computers with forensic tools (Table 4 in the appendix). With all the tools, logical images of the phones were made and reports were generated. Physical copies were not made because some tools did not allow it, and some tools required that the phone was rooted/jailbroken for a physical copy.

Few steps usually have to be done before using mobile forensic tools. On Android phones, it is necessary to enable developer mode and turn on the following settings under developer options: OEM unlocking and USB debugging. With both Android and iPhone, it is necessary to have “Stay Wake” turned on because forensic tools require that the devices are awake during extraction. Partly it is necessary so that the examiner could give necessary permissions to the tool. On the Android phone used in the thesis, it is possible to choose the “Stay Awake” option under the Settings and Display. On iPhone, it is under the Settings in the Display&Brightness section.

It is also necessary to connect the mobile device to the computer using a suitable USB cable, and in some cases, an adapter. In Figure 7 there is an example of a Cellebrite adapter. Cellebrite and MSAB tools have both a forensic kit with cables necessary for the extraction. Both kits also have an adapter. In the practical part, parts from both kits were used. With other tools, an ordinary USB cable suitable for the phone was used.

Most commercial tools usually have the option for detecting the device automatically before the extraction. After the device is connected to the computer with a USB cable, it is possible to choose the “detect” option. The detect option might detect the device correctly, or it might give multiple device choices from which the examiner has to choose the right one. Sometimes the device’s full model name is not written, and it only shows the more general model name. For example, instead of Samsung SM-G965F DS Galaxy S9+ Duos TD-LTE, there might be Samsung SM-G965F Galaxy S9+. In some cases, the forensic tool cannot detect the device automatically. In that case, there is also an option to search the device model from the device list manually.

In this work, the flight mode was not turned on before the extraction because it was not necessary to disconnect the device from the network.

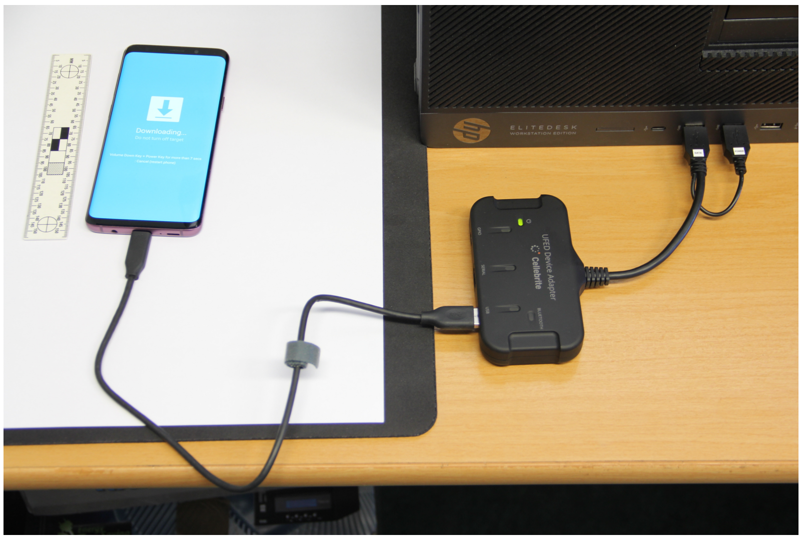


Figure 7. *Cellebrite adapter.*

In XRY Logical, the Logical (Full Read) option was used to extract data from Android and iPhone. With Android, it was necessary to choose the "downgrade all apps" option to get more data. All categories were selected for the extraction, and Android backup was enabled on the tool. With iPhone, it was necessary to set Auto-lock under Settings in Display&Brightness to "never" before using the tool. All file categories and time span "all" were selected, and iTunes encryption was enabled.

In Cellebrite UFED, the Advanced Logical copy was made of both phones. After that Cellebrite Physical Analyzer was used to generate the image. In the Physical Analyzer, AppGenie was used that found more information about applications. AppGenie is a built-in tool in UFED that analyzes applications thoroughly. For Cellebrite, it was necessary to use the Cellebrite adapter.

When working with Cellebrite, the Android phone got stuck in boot loop after the extraction. The example screen of the boot loop problem can be seen in Figure 8. Fortunately, the problem was fixed by repeating the extraction process from the beginning. During the extraction with Cellebrite, it is necessary to put the Android phone into Download mode. Download mode can be seen in Figure 9. For that, the examiner has to press the necessary buttons. Cellebrite also shows the steps that have to be taken to put the phone into Download mode. In our case the phone was able to stay in Download mode, but when leaving it, the phone stayed in boot loop. In Cellebrite, there is also a tool that helps to leave Download mode, but it did not fix the problem in our case. In the worst case, it might be necessary to flash the phone or take it apart to leave the Download mode, but this results in evidence loss.

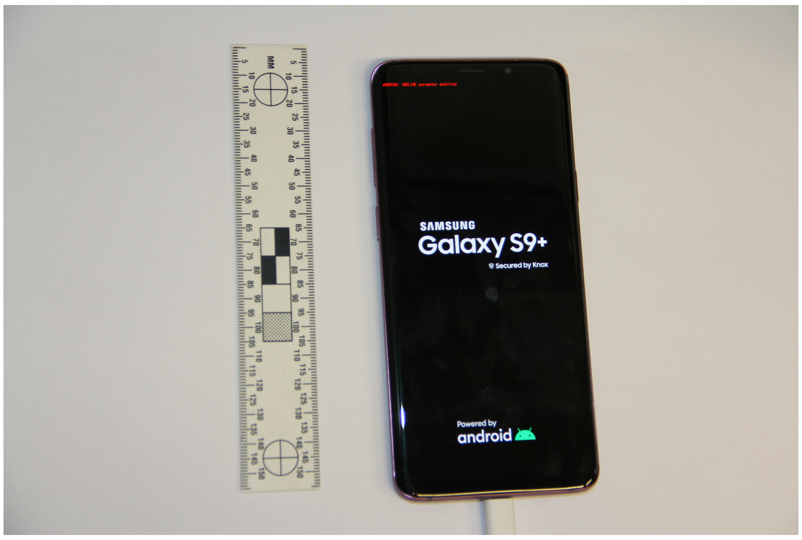


Figure 8. *Boot loop on Android phone.*

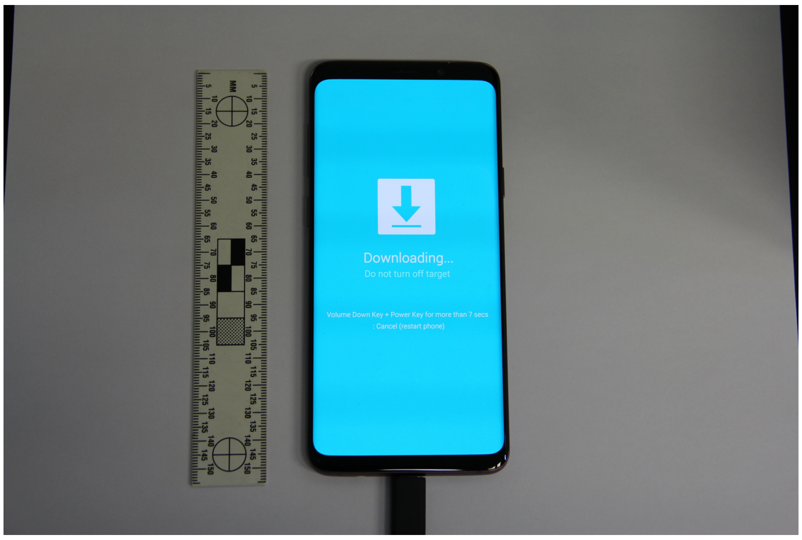


Figure 9. *Download mode on Android phone.*

With MOBILedit, a full content copy was made, and for Android, ADB backup was made because the tool enabled it. In the case of iOS, it was necessary to use iTunes backup with MOBILedit.

With Magnet Acquire, a quick image was made because the full image was not allowed. For the quick image, it was required to switch Android phone to image sharing mode PTP. PTP stands for Picture Transfer Protocol [97]. The image was in a zip file. Next, the AXIOM Process was used. It processed the image, and then AXIOM Examine generated the report.

All the extraction methods chosen with different tools are in Table 2. Extracting data from the iPhone took less time with all the tools because the phone's storage size (16GB) was smaller than the storage size of the Android phone (64GB).

Table 2. Extraction methods chosen with different tools.

Tool	**Android**	**iOS**
XRY	Logical (Full read)	Logical (Full read)
Cellebrite UFED	Decrypted Bootloader Full File System	Advanced Logical, Advanced Logical File System
MOBILedit	Full content	Full content
Magnet Aquire	Quick Image	Quick Image

Some forensic tools are capable of saving the forensic report in different languages. There are multiple languages to choose from. It is very useful because in different countries, the national or official language is used in legal processes. When the tool can generate the report in the necessary language, the reports do not have to be translated from English to the necessary language. It saves time for the examiners, and also, the correct terms are used in the report. Translated reports might have a poorer quality because translators usually are not familiar with terms used in the forensics field. The reports for this thesis were made in English.

5. Results

In this chapter, the results about the artifacts are presented.

After generating the forensic reports, messaging application data was gathered from the reports and organized into tables. The reports are in different formats. UFED reports are in .ufdr format, and reading them requires Cellebrite Reader. XRY reports are in .xry format and XAMN reader is necessary to read them. It is also possible to generate reports in pdf, HTML, and other formats that the tool allows. MOBILedit tool does not have a reader tool. With MOBILedit, it was possible to generate ADB backup, Excel files, HTML files, pdf files, and .ufdr report. Magnet AXIOM files can be read with their own tool called Magnet AXIOM Examine. Magnet AXIOM files can also be exported into Axiom Portable Case file or it is possible to make PDF, DOCX, XLSX reports.

The books "Practical Mobile Forensics - Fourth Edition" and "Learning Android Forensics - Second Edition" were used as examples when analyzing applications. Also, a poster[1] with Android and iOS artifacts was used.

It is recommended to start the examination with cache and database directories. Almost all applications use SQLite for databases. Important files that are related to these databases are rollback journals (JOURNAL), Write-Ahead Logs (WAL), and Shared Memory (SHM) files. These files may contain important information that can not be found from databases. The book also describes how to find important data from Android phone forensic reports about Facebook, WhatsApp, and Skype, and some other applications [4].

Besides SQLite, iOS devices use Plist and JSON files for application storage. Usually, it is good to start examining the applications that the tool lists. Next, Library and Documents should be checked for Plist files and finally Media directory for shared files (photos, videos, etc.). The SQLite database files can be read with DB Browser for SQLite and Plist files can be read with iBackup Viewer.

All the found paths are in tables in the appendices and are organized alphabetically. Almost all the paths have a description of the file or folder. Some descriptions of database files

[1]To be found at `https://www.sans.org/security-resources/posters/dfir-advanced-smartphone-forensics/30/download`

have a list of some of the important tables in these databases. When forensic tools did not detect (categorized as unrecognized), or it was not possible to detect what is inside the file or a folder, a hyphen was used instead of the description. Appendices also have the application package names and versions both on Android and iOS.

5.1 Application Artifacts on Android Phone

Android mostly uses SQLite for data storage, and stores preference files in the DAT or XML file formats. It is also good to start examining applications that the tool lists. Because Android application files might be in different locations, it is good to examine subdirectories of the /Root directory next. It is recommended to start with Databases and Cache directories, then Media and Cache partitions and Downloads directory. Application data might be in different places in the Media directory [4].

On Android, all applications store their data in the /data/data folder. Some data can also be found in the /sdcard folder if the permissions to the SD card were given to the application [4], [17]. In this work, SD card was not used in the Android phone. Under the /data/system folder, there is a packages.list file that contains information about all the apps. It contains package file names and data paths of applications [4]. The package name is generally the name of the folder where the application stores its data [17].

5.1.1 Messaging Application Artifacts on Android

As can be seen from Table 7 in the appendix, Discord stores its data within the com.discord package. Most applications have their package name in the following format: com.<app name>. Some applications have a different format—for example, Signal and Telegram. About Discord, the author found the APK file and a folder that had AndroidManifest.xml and a few images. APK file is an app package for distributing the app. It is possible to review the APK file using tools, for example, dex2jar (dex compiler), FileViewer Plus, SourceMeter, JSLint, FindBugs, and Java APK decompiler application. With the last tool it is possible to decompile the APK in a web browser [13]. In this work, the APK files were not examined.

When searching for Discord artifacts, a path to the localappstate.db was found. The localappstate.db was also found when searching for artifacts of other messaging applications. The localappstate.db contains application traces [98].

The AppActivity$Main file is related to launching the app [99]. It is in XML format and has launching parameters. The preferences file is also in XML format and contains app preferences.

From the forensic images, the author found important files for all examined messaging applications. These files are the APK file, description.info, description.info.xml, and icon.png.

The /bigTopDataDB.<user-id> contains email information [98]. This is not directly related to the app, but because an email contained information about Discord, the database was found.

The /calllog.db-wal has write-ahead logs of the call log. The mmssms.db contains SMS and MMS messages. The mmssms.db has a table called sms, and it is possible to find messaging application code messages from there.

Facebook stores its data within a com.facebook.katana package, but Facebook Messenger app stores its data within a com.facebook.orca package. Both packages are located in the /data/data folder [4]. The base.apk file was found. Folders that contain files related to app analysis are:

- app_analytics_beacon/normal
- app_analytics_beacon/high
- app_analytics/micro_batch

It was not possible to detect which type of files the /app_appcomponents folder contains. Few web application folders were also found. All folders related to the web application are in /data/data/com.facebook.orca/app_browser_proc_webview [100]. Some folders related to app error reporting were found.

Facebook Messenger has many files and folders where it was not possible to detect what is stored inside. For example:

- /app_file_poolcollector
- /app_graphservice
- app_gatekeepers

The /app_image folder contains images related to the app. Some of them are user's images, and some of them unknown images.

The /app_ras_blobs folder contains Facebook Messenger emoji file FacebookEmoji.ttf. The /app_videocache_logging folder has video cache logs, /app_webview has files related to the web view of the application. Preference files are in the following folders:

- /app_light_prefs
- /shared_prefs

In the com.facebook.orca folder, there are also multiple cache folders, /databases, and /file folder. Cache folders contain cache files. In /databases there were some databases that the author managed to determine and some that remained unknown. Most databases have a journal file for database management and a uid file with the uid of the database. For example, the author found a contacts_db2 and prefs_db. The contacts_db2 is an SQLite database that contains contacts. The prefs_db database contains metadata about the app and the user account [17]. Data about messages is stored in the threads_db2 database.

In /dex folder, there are multiple files. The deps file specifies "which files the sources in a directory tree may include" [101] and optimization_history_log is a log file.

In the /files folder there are the following folders:

- /audio
- /batterymetrics
- /ExoPlayerCacheDir
- /GkBootstrap
- /http
- /mobileconfig
- /looper_xplat
- /profilo
- /stickers
- /strings

The audio folder contains subfolders which contain audio files, /mobileconfig contains different configuration files, /stickers contains sticker files, and /strings has a language package.

Facebook Messenger stores multiple library folders on the phone. Facebook Messenger differs from Discord because it left some attachment links. There was a link to a location image and a Google map with the location on the map.

Kik artifacts were mainly found from three folders:

- /data/app/kik.android
- /data/user/0/kik.-android
- /data/data/kik.android

Some Kik messenger files were also in /com.google.android.webview. The base.apk and AndroidManifest.xml files are in /data/app/kik.android folder. The /data/data/kik.android folder has many cache folders and files, a folder for smileys, a folder for icons, a preference files folder, and some other folders. In /data/user/0/kik.android, the application stores cache files.

Signal also has some files in /com.google.android.webview like Facebook Messenger does. Signal has artifacts mainly in /data/data/org.thoughtcrime.securesms folder. This folder contains app avatars folder, app parts folder, stickers folder, a journal file, a .lock file, an installation file, a manifest file, and a preferences folder.

According to the book "Practical Mobile Forensics - Fourth Edition", some important artifacts that can be extracted when analyzing Skype are the following: username and IP address, profile picture, call logs, chat messages, files transferred and group chats [4]. Information about shared files is stored in Transfers table in /data/data/-com.skype.raider/files/<username>/main.db. Downloaded files are stored in Downloads folder. Received files are stored on SD card [4]. Some files are in /com.google.android.webview and in /data/data/com.skype.raider folders.

The database file of Skype is main.db. The author did not find the file. According to a book [13], the following tables can be found from the database: DbMeta, Contacts, Videos, SMSes, CallMembers, ChatMembers, Alerts, Conversations, Participants, VideoMessages, LegacyMessages, Calls, Accounts, Transfers, Voicemails, Chats, Messages, ContactGroups, AppSchemaVersion, MediaDocuments, MessageAnnotations, Translators, tracker_journal [13].

Slack left only a small amount of traces. Besides the main files, the author found /Notifications folder containing audio files and the profile photo of the user in /cache/file-upload folder.

Telegram left a considerable amount of artifacts. A more precise description of Telegram artifacts can be found in the subsection 5.1.2 about Telegram artifacts on Android phone.

Viber has most files in the following folders:

- /com.google.android.webview
- /data/app
- /data/data/com.viber.voip
- /data/media/0
- /storage/emulated/0
- /sdcard/viber

A lot of cache folders were found. Folders for thumbnails, emoticons and stickers were also found. Viber databases are viber_data which contains information about user's contacts and viber_messages which contains information about the app's usage and has viber_messages-journal for database management. Important tables in viber_data are phonebookcontact and phonebookdata. Important tables in viber_messages are messages and participants_info. In the /files folder there are the following folders:

- /.emoticons
- /.gems
- /.gif
- /.image
- /.import
- /.shsh
- /.stickers
- /.temp
- /.thumbnails
- /User photos
- /.video
- /Notifications

Three sdcard folders were also found. These are

- /User Photos
- /Viber Images
- /Viber Videos

WhatsApp stores its data within the com.whatsapp package. For example, a user's profile picture has the following path: /data/data/com.whatsapp/me.jpg. Chat messages are stored in /data/data/com.whatsapp/databases/msgstore.db. Important tables in that database are call_log, chat_list and messages. The database has a write-ahead log file /msgstore.db-wal.

The msgstore.db also has encrypted versions of the database. The encrypted WhatsApp databases have .crypt12 extension [102]. The author also found a second database /wa.db that was empty. The database also has a write-ahead log file.

Most shared images, videos and other shared files are stored in /sdcard/WhatsApp/Media [4]. WhatsApp stores different attachments in three types of folders. For example, audio files are stored in:

- /WhatsApp Audio/
- /WhatsApp Audio/Private/
- /WhatsApp Audio/Sent/

Documents are stored in:

- /WhatsApp Documents/
- /WhatsApp Documents/Private/
- /WhatsApp Documents/Sent/

Voice notes are stored in:

- /WhatsApp Voice Notes/

The /files folder contains different type of data. For example, images, libraries, wallpapers, and logs.

Wickr Me database is in /data/data/com.mywickr.wickr2/databases/wickr_db/wickr_db.decrypted. The database has a write-ahead log also. Some sent GIFs were found from subfolders in /data/data/com.mywickr.wickr2/files/. Like other applications, Wickr Me also had some cache folders.

It was decided to examine the artifacts of two messaging applications more thoroughly to give a better overview of the artifacts that messaging applications leave on the phone. The following subsection describes the artifacts of Telegram and WhatsApp more thoroughly.

5.1.2 Telegram artifacts on Android

The /data/app/ folder contains the APK files of third-party apps including Telegram [4]. AndroidManifest.xml describes essential information about the app to the Android operating system, the Android build tools, and Google Play [103]. It “contains the application’s package name, its functionality, permissions, hardware, and software requirements for installation" [13]. It is important to understand the permissions related to an app because it helps to understand the type of evidence to look from SQLite database files and the type of evidence to request from the provider [13]. Limiting the scope of the analysis helps to save time because examining one database file can take many days or even weeks [13].

Understanding the manifest file is also important from the mobile security perspective. Some apps can request more permissions than is necessary for the app. Also, some app permissions are low risk and some high risk [13].

The main data files and folders of the application are stored in the /data/data/org.telegram.messenger folder. The folder contains:

- /databases folder
- description files
- /files folder
- app icon image
- _manifest file
- /shared_prefs folder

The /databases folder contains com.google.android.datatransport.events and com.google.android.datatransport.events-journal which are both log files. The com.google.android.datatransport.events file was empty and com.google.android.datatransport.events-journal contains database android_metadata.

/org.telegram.messenger contains two description files (in text format and XML format). Both of them contain basic information about the application, and they can be opened with a text editor. The information they contain differs. The text file has the following information: name, package, version, is system flag, app size, data size, cache size, first install time, last update time, and installer package name, which did not have a value in our file. The XML file contains more information.

The /files folder has folders for accounts. The /files folder and account folders contain cache4.db. All cache4.db files are empty. The folder also contains cache4.db-wal file. The write-ahead log or "wal" file is a journal file "that records transactions that have been committed but not yet applied to the main database [104]."

The _manifest file has essential information about the app [103].

/shared_prefs contains key-value pairs of preferences that will persist even when the user closes the application [105]. The primary purpose of this file is to store user-specific configuration details [105].

Telegram artifacts were also found in the /storage/emulated/0 (same as /mnt/sdcard) that refers usually to the SD card, but because the phone used in our experiments did not have an external SD card then it might be possible that the Samsung phone has an internal "external" or non removable external storage [106], [107].

The folder /storage/emulated/0/Android/data/org.telegram.messenger/cache/ contains pictures shared in Telegram chats or used as a profile picture. It also contains .tgs files, which contain Telegram Animated Stickers [108]. TGS files are JSON files that are mostly GZip compressed [108]. The cache folder also contains cache_r.0 folder, GIFs, 0 Files, M Files, Microsoft Edge HTML Documents containing shared GIFs, a NOMEDIA file, and PRELOAD files. /storage/emulated/0/Download/ contains downloaded files.

The /storage/emulated/0/Telegram folder has the following folders: Telegram Audio, Telegram Documents, Telegram Images, Telegram Video. Telegram Audio contains shared audio files and a .nomedia file. The audio files are stored in OGG format. OGG files are compressed audio files [109]. NOMEDIA files are empty text files used to mark the folder as having no multimedia data so that multimedia players or file browser's search function would not scan the folder [110], [111]. NOMEDIA files are used on Android phones, they do not have a prefix, and they are named as .nomedia [110]. The /Telegram Documents folder contains shared audio and video files, .tgs files, and a .nomedia file. There was also an unknown video file that was not shared in the app. Telegram Images contains shared images and a .nomedia file. Telegram Video contains shared video files and a .nomedia file.

5.1.3 WhatsApp artifacts on Android Phone

Like with all the other applications, the AndroidManifest.xml was found. Files that the author found in case of many other applications also, were:

- localappstate.db
- calllog.db-wal
- app_phenotype_file

The localappstate.db has two tables: android_metadata, appstate. The android_metadata table has the location information. The appstate table has the package names of different applications and some additional information related to these apps. For example, if an automatical update is allowed, data delivery timestamp, first download time, email of the account, and the last time when the app was updated.

The author found two WhatsApp database files. These are the main files that contain WhatsApp artifacts on Android. The msgstore.db contains 128 tables. The most interesting tables for a forensic examiner might be call_log, chat, message and messages. The call_log table contains for example, the call id, the timestamp, if it was a video call, the duration of the call, and the result of the call. The app stores the subject in the chat table, the timestamp of when the chat was created, which message was read last, if the message was archived, the unseen message count, the missed calls count, and more. The message table, for example, has the status, the recipient count, origin, timestamp, timestamp when the message was received, message type, text data. The messages table has the status, data, timestamp, information about attachments, the location (latitude, longitude) where the message was sent from, the message receiving time, message sending timestamp, and other information.

The wa.db database was found to be empty. The related write-ahead logs were also found for both files.

The /files folder contains several folders. The Avatars folder has profile pictures. The names of the images stored there end with .j file extension. The decompressed folder contains libraries that are in ELF format. The manifest.json file in /files/downloadable folder has startup parameters for when the web application is launched [112]. The /files/downloadble folder also contains images and wallpapers. The author also found /files/Logs with log files and /files/.Shared with shared files.

It was not possible to detect what /data/data/com.whatsapp/lib-main/ contains, but /data/data/com.whatsapp/shared_prefs/ contains preference links. Besides /data/data/-com.whatsapp, data related to WhatsApp was also found in /data/media/0/WhatsApp/Media. It contains few database files, sticker files, sent and received attachments.

The author found the following WhatsApp folders from the SD card:

- /sdcard/WhatsApp/Databases/
- /sdcard/WhatsApp/Media/

Directories that contain cache files are

- /storage/emulated/0/Android/data/com.whatsapp/cache
- /storage/emulated/0/Android/data/com.whatsapp/cache/cache_r.0/

The author also found paths to databases and backups. From /storage/emulated/0/WhatsApp/Media/ the author found sent and received media files and voice notes.

5.2 Application Artifacts on iOS Phone

Most data on iOS phones is stored in the /private/var folder. Some important artifacts that multiple apps have in common are the call history, sms.db, interactionC.db.

Calls that are placed, missed, and received are logged in the call history [4]. In older iOS versions (up to iOS 7) [98] the call history is stored in call_history.db. In newer versions (from iOS 8) [98] it is stored in CallHistory.storedata. The call history is stored in the ZCALLRECORD table. The table has the following information: timestamps of calls, duration of calls, locations of phone numbers, phone numbers, service providers. The timestamps in this table are in Mac absolute time format [4]. Timestamps on iOS are stored in Unix timestamp or Mac absolute time format. Unix timestamps show “the number of seconds that have elapsed since Unix epoch time, which started at midnight on January 1, 1970. Mac absolute time is the number of seconds that have elapsed since Mac epoch time, which started at midnight on January 1, 2001 [4].” From iOS application data, it is possible to find also WebKit/Chrome time which shows “the number of microseconds since midnight on January 1, 1601 [4].”

Only limited number of calls are stored in the active call history database, and if the database is full, older call history is stored in the free pages of the database and can be recovered manually or using tools [4]. Most applications that had the calling function and where it was used left traces of call history. These are Discord, Skype, WhatsApp.

Sent and received SMS and MMS messages are stored in sms.db. In addition to the messages, the database contains the phone number of the remote party, date and time, and other carrier information. iOS applications that left a trace related to sms.db are Discord, Signal, Telegram, Viber, WhatsApp.

In the interactionC.db database the information about how the user interacts with different applications is stored [4]. The table ZINTERACTIONS contains information about if “the user reads a message, sends a message, performs a call, etc.” [4]. The ZCONTACTS table “contains information about contacts who were involved in the user’s interactions with the device” [4]. The applications that had a path to interactions.db are Discord, Facebook

Messenger, Signal, Telegram, WhatsApp, Wickr.

The DataUsage.sqlite database contains data about application usage [113]. Facebook Messenger, Kik, Signal, Skype, Telegram, Viber, WhatsApp, Wickr Me had traces of it.

Some property list files, also referred to as Plist files were found. These files are in binary format, and a Property List Editor or iBackup Viewer can be used to view them. The editor converts the binary format to ASCII format. Plist files might not always have the .plist file extension. Plist files contain less information than SQLite databases, but they can still be useful [4]. Plist files that were found in this work are Manifest.plist, group.com.kik.chat.plist, com.viber.plist, group.net.whatsapp.WhatsApp.shared.plist.

For each installed application, a subdirectory with a universally unique identifier (UUID) is created. The directory is located in /private/var/mobile/Containers/Data/Application directory. Most of the files stored there are in SQLite and Plist format [4].

5.2.1 Messaging Application Artifacts on iOS

In some forensic tools that were used, it is possible to read some of the messages.

As seen in Table 8 in the appendix, it was not possible to extract much data about Discord on the iOS phone. With four tools, the package location, a cookie file, call history, log of application traces, security code SMS, and some attachment URLs were found. Attachment URLs starting with media.discordapp open the attachment in the browser. URLs starting with cdn.discordapp download the image to the computer. According to an article [114], Discord attachments remain accessible using the URLs even after the attachment is deleted in the message.

Table 11 in the appendix has artifacts of Facebook Messenger on iOS phone. Compared to Discord, more traces were found. Several links to attachments were found, but most of the URLs had expired. Some examples of the links are also in the table. Only the URLs of the sent pictures are not expired. URLs of GIFs and icons are also not expired. The location had been shared in the app, but the tools did not show where or when. The database lightspeed-100065097170555.db contains three tables related to the messages: messages, attachments, contacts. WhatsApp chat artifacts contained a link to Facebook Messenger Room.

From Kik artifacts, a path to known and unknown profile pictures was found. Also, paths to attachments, data cache, some media file URLs and databases were found. The kik.sqlite database contains 17 tables. The table 14 in the appendix has some examples. The ZKIKATTACHMENT table contains encoded information about the content of attachments, timestamps, and some additional information. The ZKIKCHAT table contains information about the chats. About the user, last message, the timestamp, and additional information. Data about messages is in ZKIKMESSAGE table. The table ZKIKUSER has information about users with whom the app user has had a connection with or has searched for.

The Plist file group.com.kik.chat.plist contains all kinds of information. For example, some timestamps, user email, user names, URLs, attachment names, booleans.

Not many artifacts were found about the Signal app. Only the main files (cookies, interactionC.db, SMS/MMS messages, DataUsage.sqlite) and application package location were found.

Skype also had only a few artifacts. The package path, the location of cookies, and the call history were found.

Telegram also left only the main artifacts.

Viber left some interesting artifacts. The Contacts.data database contains data about Viber contacts. The folder containing app icons was found in /private/var/mobile/Containers/Shared/AppGroup/group.viber.share.container/com.viber/ViberIcons/. Also, one link to a GIF that might have been shared in the chat was found.

WhatsApp left a considerable amount of artifacts. For example, ChatStorage.sqlite database contains 18 tables. Table ZWAMESSAGE contains information about messages. It contains message dates, content, and other information.

ContactsV2.sqlite database contains four tables. The most important table being ZWAADRESSBOOKCONTACT which contains data about WhatsApp contacts. The database also has the following tables: Z_METADATA, Z_MODELCACHE, PRIMARYKEY.

The Plist file group.net.whatsapp.WhatsApp.shared.plist contains different types of data. For example, booleans, timestamps, strings, and numbers.

Also, a folder that contains pictures was found. For example, it contains status pictures, shared media, location thumbnails. “A thumbnail is an image with a reduced file size that is used as a placeholder for full-sized multimedia content [115].”

About Wickr Me the main artifacts (ChatStorage.sqlite, DataUsage.sqlite, interactionC.db) were found.

In the following subsections, Telegram and WhatsApp are described more in depth.

5.2.2 Telegram artifacts on iOS

Forensic tools did not detect many Telegram artifacts on the iOS phone. The author only found sms.db, interactionC.db, DataUsage.sqlite and backup files.

The sms.db database contains SMS and MMS data. It has 16 tables. A forensic examiner might be interested in the attachment, chat, deleted messages, handle, message tables. The attachment table contains the following important fields: guid, date when it was created, the name of the shared file, MIME type, the number of bytes that were transferred.

The chat table has, for example, guid, account id, chat identifier, group id. The guid is in the following format: SMS;-;<app name>. The guid for Telegram is SMS;-;telegram.

The deleted messages table was empty. The handle table has messaging application IDs, the country where the app is used, what type of service it is, uncanonicalized IDs, and person-centric IDs (can be empty).

Messages sent to or from the built-in SMS app are stored in the messages table. It is possible to find the codes for different messaging apps, the date of the messages, and the date when the message was read.

5.2.3 WhatsApp artifacts on iOS

About WhatsApp, the author found the application package location, ChatStorage.sqlite database, ContactsV2.sqlite database, Library folder, and Preferences folder. According to [13] the Library folder has user-related data including cache, cookies, and other personal information. The Preferences folder might contain usernames and passwords [13]. All these files and folders were in private/var/mobile/Containers/Shared/AppGroup/-group.net.whatsapp.WhatsApp.shared folder. The ChatStorage.sqlite database has 18 tables. Important tables are ZWAMEDIAITEM and ZWAMESSAGE. Other tables did not have much information. The ZWAMEDIAITEM table has information about attachments. For example, the file size, media origin, message, location, media URL date, the local path, media URL, thumbnail path. ZWAMESSAGE contains information about messages. For example, message status, message type, media item ID, message info ID, message date, sent date, message content. The information about media items sent with messages can be found from the ZWAMEDIAITEM table using media item ID.

When searching for WhatsApp artifacts, the author found CallHistory.storedata, interactionC.db, sms.db, DataUsage.sqlite. Same artifacts were found about many other messaging apps on iOS also.

The Media folder contains profile picture thumbnails. The Message folder has WhatsApp status pictures and shared media files.

5.3 Discussion and Future Work

It can be said that it is possible to find artifacts about all the applications used in this thesis. The applications that left the most artifacts were Facebook Messenger, Kik, Telegram, Viber, and WhatsApp.

Further work can be done on what happens when files are deleted and what happens to files during the app's usage. It is also possible to capture or save network traffic between the devices with Wireshark or Network Miner and analyze PCAP files. Moreover, one very important topic is encryption.

Further work can also be done on comparing the forensic tools. However because the field is changing so rapidly, it might not be necessary because NIST has forensic tool testing documentation [2].

[2]To be found at `https://www.nist.gov/itl/ssd/software-quality-group/computer-forensics-tool-testing-program-cftt/cftt-technical/mobile`

6. Summary

The main aim of this thesis was to find out where communication apps store their data on smartphones and describe what the found artifacts contain. For that an Android and an iOS phone were used, ten popular messaging applications were installed on them, messages were sent between phones, forensic images were made with four different forensic tools and reports were created. Then the reports were analyzed and data was gathered into tables used to describe the artifacts. Most artifacts that are in the tables are described in Chapter 5.

The following messaging applications were chosen for the analysis: Discord, Facebook Messenger, Kik Messenger, Signal, Skype, Slack, Telegram, Viber, WhatsApp, Wickr Me. The applications that left the most artifacts were Facebook Messenger, Kik, Telegram, Viber, and WhatsApp.

It was found that it is possible to find messages, attachments, shared location, and other types of data. Some messaging applications store some of their databases on the phone. It was not possible to find databases for all the apps from the forensic reports. Several applications had links to sent attachments or other media files related to the application.

The results of this thesis give an overview of what traces can be found about ten messaging applications with four forensic tools. The results can be useful for forensic examiners or people interested in smartphones, messaging applications or mobile forensic tools.

Further work can be done on finding out what happens when files are deleted and what happens to files during the app's usage. It is also possible to analyze network traffic. One very important topic is the encryption of messages.

Bibliography

[1] "Ascii table - ASCII character codes and html, octal, hex and decimal chart conversion," [Online]. Available: https://www.asciitable.com/ (visited on 05/05/2021).

[2] "Academic partner program (CAPP) - cellebrite," [Online]. Available: https://www.cellebrite.com/en/about/capp/ (visited on 05/05/2021).

[3] "CSV - comma separated values," [Online]. Available: https://www.abbreviations.com/term/2222 (visited on 05/05/2021).

[4] R. Tamma, O. Skulkin, H. Mahalik, and S. Bommisetty, *Practical Mobile Forensics - Fourth Edition*, 4th ed. Packt Publishing, Apr. 9, 2020, 384 pp., ISBN: 978-1-83864-752-0. [Online]. Available: https://subscription.packtpub.com/book/security/9781838647520 (visited on 03/05/2021).

[5] "RJ45 definition," [Online]. Available: https://techterms.com/definition/rj45 (visited on 05/05/2021).

[6] "Sim," [Online]. Available: https://dictionary.cambridge.org/dictionary/english/sim (visited on 05/05/2021).

[7] "TAR - TAR compressed file archive," [Online]. Available: https://www.abbreviations.com/term/42751 (visited on 05/05/2021).

[8] "XLS - microsoft excel spreadsheet," [Online]. Available: https://www.abbreviations.com/term/43177 (visited on 05/05/2021).

[9] "XML," [Online]. Available: https://dictionary.cambridge.org/dictionary/english/xml (visited on 05/05/2021).

[10] R. Umar, I. Riadi, and G. M. Zamroni, "Mobile forensic tools evaluation for digital crime investigation," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 8, no. 3, pp. 949–955, 2018. DOI: 10.18517/ijaseit.8.3.3591. [Online]. Available: https://www.researchgate.net/publication/326020731_Mobile_Forensic_Tools_Evaluation_for_Digital_Crime_Investigation.

[11] A. Watson. (Jun. 6, 2019). "5 real world investigations where UFED ultimate helped solve the case - cellebrite," Cellebrite, [Online]. Available: https://www.cellebrite.com/en/5-real-world-investigations-where-ufed-ultimate-helped-solve-the-case/ (visited on 03/25/2021).

[12] S. O'Dea. (Mar. 18, 2021). "Smartphone users 2020," Statista, [Online]. Available: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/ (visited on 03/24/2021).

[13] D. R. Hayes, *A Practical Guide to Digital Forensics Investigations, 2nd Edition*, 2nd. Pearson IT Certification, Oct. 2020, 720 pp., ISBN: 978-0-7897-5991-7. [Online]. Available: https://learning.oreilly.com/library/view/a-practical-guide/9780134878942/ (visited on 03/06/2021).

[14] Riigi Teataja. (May 30, 2016). "Code of criminal procedure – riigi teataja," [Online]. Available: https://www.riigiteataja.ee/en/eli/530052016003/consolide (visited on 05/14/2021).

[15] W. Chai. (Jan. 2021). "What is the CIA triad?" WhatIs.com, [Online]. Available: https://whatis.techtarget.com/definition/Confidentiality-integrity-and-availability-CIA (visited on 05/14/2021).

[16] Telegram. (Apr. 8, 2015). "Active sessions and two-step verification," Telegram, [Online]. Available: https://telegram.org/blog/sessions-and-2-step-verification (visited on 03/31/2021).

[17] O. Skulkin, D. Tindall, and R. Tamma, *Learning Android Forensics - Second Edition*. Packt Publishing, Dec. 28, 2018, 328 pp., ISBN: 978-1-78913-101-7. [Online]. Available: https://subscription.packtpub.com/book/networking_and_servers/9781789131017 (visited on 03/05/2021).

[18] J. Eichbaum. (Sep. 9, 2019). "Five continual challenges with smartphone forensics," MSAB, [Online]. Available: https://www.msab.com/2019/09/09/five-continual-challenges-with-smartphone-forensics/ (visited on 04/02/2021).

[19] "About USB-IF," [Online]. Available: https://www.usb.org/about (visited on 05/05/2021).

[20] MSAB, "Are there any "court-approved" mobile forensic tools?" MSAB, Research report, pp. 1–6. [Online]. Available: https://www.msab.com/downloads/ (visited on 03/31/2021).

[21] "Four critical success factors in mobile forensics: Getting the data is just the beginning," MSAB, Research report, pp. 1–9. [Online]. Available: https://www.msab.com/downloads/ (visited on 03/31/2021).

[22] S. Saleem, O. Popov, and O. K. Appiah-Kubi, "Evaluating and comparing tools for mobile device forensics using quantitative analysis," presented at the 4th International Conference on Digital Forensics & Cyber Crime, vol. 114, Lafayette, USA, Oct. 25, 2012. DOI: 10.1007/978-3-642-39891-9_17.

[23] Wi-Fi Alliance. (May 8, 2000). "Wireless ethernet compatibility alliance (WECA) awards new wi-fi interoperability certification," [Online]. Available: https://www.wi-fi.org/news-events/newsroom/wireless-ethernet-compatibility-alliance-weca-awards-new-wi-fi-interoperability (visited on 04/25/2021).

[24] "Hexdump(1) - linux manual page," [Online]. Available: https://man7.org/linux/man-pages/man1/hexdump.1.html (visited on 04/2013).

[25] Android Developers. (Mar. 11, 2021). "Platform architecture," Android Developers, [Online]. Available: https://developer.android.com/guide/platform (visited on 03/24/2021).

[26] J. Freeman. (Aug. 8, 2019). "What is an API? application programming interfaces explained," InfoWorld, [Online]. Available: https://www.infoworld.com/article/3269878/what-is-an-api-application-programming-interfaces-explained.html (visited on 05/05/2021).

[27] B. Gavin. (Aug. 23, 2018). "What is a DAT file (and how do i open one)?" How-To Geek, [Online]. Available: https://www.howtogeek.com/363326/what-is-a-dat-file-and-how-do-i-open-one/ (visited on 05/05/2021).

[28] "Example: Preferences in android," [Online]. Available: https://www.protechtraining.com/blog/post/example-preferences-in-android-75 (visited on 04/22/2021).

[29] A. O. S. Project. (Apr. 27, 2021). "Encryption | android open source project," [Online]. Available: https://source.android.com/security/encryption (visited on 04/22/2021).

[30] Android Open Source Project. (Mar. 2, 2021). "Frequently asked questions," Android Open Source Project, [Online]. Available: https://source.android.com/setup/start/faqs (visited on 04/22/2021).

[31] A. Spadafora. (Mar. 19, 2021). "Sky global apparently shuts down following police arrests," TechRadar, [Online]. Available: https://www.techradar.com/news/sky-global-apparently-shuts-down-following-police-arrests (visited on 03/31/2021).

[32] The Brussels Times. (Mar. 10, 2021). "When sky ECC fell, so too did belgian crime lords," The Brussels Times, [Online]. Available: https://www.brusselstimes.com/belgium/159176/cracking-of-sky-ecc-encrypted-messaging-service-brings-down-organised-crime-lords/ (visited on 03/31/2021).

[33] Eurojust. (Mar. 10, 2021). "New major interventions to block encrypted communications of criminal networks | eurojust | european union agency for criminal justice cooperation," Eurojust, [Online]. Available: https://www.eurojust.europa.eu/new-major-interventions-block-encrypted-communications-criminal-networks (visited on 03/31/2021).

[34] United States Department of Justice. (May 28, 2019). "Chief executive of communications company sentenced to prison for providing encryption services and devices to criminal organizations," [Online]. Available: https://www.justice.gov/usao-sdca/pr/chief-executive-communications-company-sentenced-prison-providing-encryption-services (visited on 03/31/2021).

[35] F. N. Dezfouli, A. Dehghantanha, B. Eterovic-Soric, and K.-K. R. Choo, "Investigating social networking applications on smartphones detecting facebook, twitter, LinkedIn and google+ artefacts on android and iOS platforms," *Australian Journal of Forensic Sciences*, vol. 48, no. 4, pp. 469–488, 2016. DOI: 10.1080/00450618.2015.1066854. [Online]. Available: https://www.researchgate.net/publication/282484336_Investigating_Social_Networking_applications_on_smartphones_detecting_Facebook_Twitter_LinkedIn_and_Google_artefacts_on_Android_and_iOS_platforms.

[36] J. Cohen. (Sep. 4, 2020). "iOS more popular in japan and US, android dominates in china and india," PCMAG, [Online]. Available: https://www.pcmag.com/news/ios-more-popular-in-japan-and-us-android-dominates-in-china-and-india (visited on 03/05/2021).

[37] "HFS - hierarchical file system," [Online]. Available: https://www.abbreviations.com/term/88847 (visited on 05/05/2021).

[38] "Apple file system reference," p. 181, Jun. 22, 2020.

[39] "JSON - java script object notation," [Online]. Available: https://www.abbreviations.com/term/1591694 (visited on 05/05/2021).

[40] "About discord | our mission and values," Discord, [Online]. Available: https://discord.com/ (visited on 03/24/2021).

[41] "Features – discord," [Online]. Available: https://support.discord.com/hc/en-us/sections/201110577-Features (visited on 04/22/2021).

[42] "Add support for end-to-end encryption," Discord, [Online]. Available: http://support.discord.com/hc/en-us/community/posts/360047118232-Add-support-for-end-to-end-encryption (visited on 04/23/2021).

[43] "GIF - graphics interchange format," [Online]. Available: https://www.abbreviations.com/term/20100 (visited on 05/05/2021).

[44] "Secret conversations | messenger help center," [Online]. Available: https://www.facebook.com/help/messenger-app/1084673321594605/ (visited on 03/24/2021).

[45] "Messenger secret conversations technical whitepaper," p. 15, May 18, 2017.

[46] "How long does facebook store private messages that i sent? - quora," [Online]. Available: https://www.quora.com/How-long-does-Facebook-store-private-messages-that-I-sent (visited on 04/23/2021).

[47] "Kik," [Online]. Available: https://www.kik.com/ (visited on 03/24/2021).

[48] K. Wagner. (Dec. 21, 2015). "Is your messaging app encrypted?" Vox, [Online]. Available: https://www.vox.com/2015/12/21/11621610/is-your-messaging-app-encrypted (visited on 04/23/2021).

[49] "Signal messenger: Speak freely," Signal Messenger, [Online]. Available: https://signal.org/en/index.html (visited on 03/24/2021).

[50] L. Stanton. "Signal messaging – where are the messages stored?" Alphr, [Online]. Available: https://www.alphr.com/signal-messaging-where-are-messages-stored/ (visited on 04/23/2021).

[51] "Skype | communication tool for free calls and chat," [Online]. Available: https://www.skype.com/en// (visited on 03/24/2021).

[52] "Does skype use encryption? | skype support," [Online]. Available: https://support.skype.com/en/faq/FA31/does-skype-use-encryption (visited on 04/23/2021).

[53] "Where work happens | slack," [Online]. Available: https://slack.com/intl/en-ee/ (visited on 03/24/2021).

[54] "Telegram FAQ," Telegram, [Online]. Available: https://telegram.org/faq (visited on 03/24/2021).

[55] "Features," Viber, [Online]. Available: https://www.viber.com/en/features/ (visited on 03/24/2021).

[56] "Viber account security and encryption - viber support knowledge base," [Online]. Available: https://help.viber.com/en/article/viber-account-security-and-encryption#:~:text=Viber's%20security%20measures&text=End-to-end%20encryption%20means,anyone%20else,%20not%20even%20Viber (visited on 04/23/2021).

[57] "About WhatsApp," WhatsApp.com, [Online]. Available: https://www.whatsapp.com/about/ (visited on 03/24/2021).

[58] WhatsApp. "WhatsApp security advisories," WhatsApp.com, [Online]. Available: https://www.whatsapp.com/security/advisories (visited on 05/14/2021).

[59] "WhatsApp security advisories," WhatsApp.com, [Online]. Available: https://www.whatsapp.com/security/advisories (visited on 04/23/2021).

[60] "Wickr me," Wickr, [Online]. Available: https://wickr.com/me/ (visited on 03/24/2021).

[61] "Can wickr read my messages?" Wickr Inc. [Online]. Available: https://support.wickr.com/hc/en-us/articles/115007723287-Can-Wickr-read-my-messages- (visited on 04/23/2021).

[62] PastorsLine. (Feb. 15, 2017). "Why cloud based text messaging beats apps like skype, WhatsApp, viber," PastorsLine, [Online]. Available: https://pastorsline.com/cloud-based-text-messaging-solution-trumps-applications-like-skype-imessenger-whatsapp-etc/ (visited on 04/22/2021).

[63] D. Walnycky, I. Baggili, A. Marrington, J. Moore, and F. Breitinger, "Network and device forensic analysis of android social-messaging applications," *Digital Investigation*, vol. 14, S77–S84, S1 2015. DOI: 10.1016/j.diin.2015.05.009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1742287615000547.

[64] A. Azfar, K.-K. R. Choo, and L. Liu, "An android communication app forensic taxonomy," *Journal of Forensic Sciences*, vol. 61, no. 5, pp. 1337–1350, 2016. DOI: 10.1111/1556-4029.13164. [Online]. Available: https://www.researchgate.net/publication/305623470_An_Android_Communication_App_Forensic_Taxonomy.

[65] K. Rathi, U. Karabiyik, T. Aderibigbe, and H. Chi, "Forensic analysis of encrypted instant messaging applications on android," presented at the 6th International Symposium on Digital Forensic and Security, ISDFS 2018 - Proceeding, vol. 2018-January, Antalya, Turkey, May 2018, pp. 1–6. DOI: 10.1109/ISDFS.2018.8

355344. [Online]. Available: https://ieeexplore.ieee.org/document/8355344.

[66] C. Anglano, M. Canonico, and M. Guazzone, “Forensic analysis of telegram messenger on android smartphones,” *Digital Investigation*, vol. 23, pp. 31–49, 2017. DOI: 10.1016/j.diin.2017.09.002. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1742287617301767.

[67] C. Anglano, “Forensic analysis of whatsapp messenger on android smartphones,” *Digital Investigation*, vol. 11, no. 3, pp. 201–213, 2014. DOI: 10.1016/j.diin.2014.04.003. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1742287614000437.

[68] A. Iqbal, A. Marrington, and I. Baggili, “Forensic artifacts of the ChatON instant messaging application,” in *2013 8th International Workshop on Systematic Approaches to Digital Forensics Engineering (SADFE)*, Hong Kong, China: IEEE, Nov. 2013, pp. 1–6, ISBN: 978-1-4799-4061-5. DOI: 10.1109/SADFE.2013.6911538. [Online]. Available: https://ieeexplore.ieee.org/document/6911538.

[69] C. Sgaras, M.-T. Kechadi, and N.-A. Le-Khac, “Forensics acquisition and analysis of instant messaging and VoIP applications,” in *Computational Forensics*, U. Garain and F. Shafait, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2015, pp. 188–199, ISBN: 978-3-319-20125-2. DOI: 10.1007/978-3-319-20125-2_16. [Online]. Available: https://link.springer.com/chapter/10.1007%2F978-3-319-20125-2_16.

[70] A. Shortall and M. A. H. B. Azhar, “Forensic acquisitions of WhatsApp data on popular mobile platforms,” in *2015 Sixth International Conference on Emerging Security Technologies (EST)*, Braunschweig, Germany: IEEE, Sep. 2015, pp. 13–17, ISBN: 978-1-4673-9799-5. DOI: 10.1109/EST.2015.16. [Online]. Available: https://ieeexplore.ieee.org/document/7429264.

[71] K. Ovens and G. Morison, “Forensic analysis of kik messenger on iOS devices,” *Digital Investigation*, vol. 17, pp. 40–52, 2016. DOI: 10.1016/j.diin.2016.04.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1742287616300329.

[72] A. M. Al-Rawashdeh, Z. A. Al-Sharif, M. I. Al-Saleh, and A. S. Shatnawi, “A post-mortem forensic approach for the kik messenger on android,” in *2020 11th International Conference on Information and Communication Systems (ICICS)*, ISSN: 2573-3346, Irbid, Jordan: IEEE, Apr. 2020, pp. 079–084, ISBN: 978-1-72816-

227-0. DOI: 10.1109/ICICS49469.2020.239559. [Online]. Available: https://ieeexplore.ieee.org/document/9079092.

[73] A. J. Bhatt, C. Gupta, and S. Mittal, "Network forensics analysis of iOS social networking and messaging apps," in *2018 Eleventh International Conference on Contemporary Computing (IC3)*, ISSN: 2572-6129, Noida, India: IEEE, Aug. 2018, pp. 1–6, ISBN: 978-1-5386-6835-1. DOI: 10.1109/IC3.2018.8530576. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8530576&tag=1.

[74] S. M. Judge, "Mobile forensics : Analysis of the messaging application signal," 2017, Accepted: 2020-07-09T14:40:08Z ISBN: 9781041189190. [Online]. Available: https://shareok.org/handle/11244/325058 (visited on 03/04/2021).

[75] H. Azhar, R. Cox, and A. Chamberlain, "Forensic investigations of popular ephemeral messaging applications on android and iOS platforms," *International Journal on Advances in Security*, vol. 13, no. 1, pp. 41–53, Jul. 2, 2020, Number: 1 & 2 Publisher: IARIA, ISSN: 1942-2636. [Online]. Available: http://www.iariajournals.org/security/sec_v13_n12_2020_paged.pdf (visited on 03/04/2021).

[76] M. A. H. B. Azhar and T. E. A. Barton, "Forensic analysis of secure ephemeral messaging applications on android platforms," *Communications in Computer and Information Science*, vol. 630, pp. 27–41, 2016. DOI: 10.1007/978-3-319-51064-4_3. [Online]. Available: https://www.researchgate.net/publication/312304533_Forensic_Analysis_of_Secure_Ephemeral_Messaging_Applications_on_Android_Platforms.

[77] S. McCombes. (May 8, 2019). "How to do a case study | examples and methods," Scribbr, [Online]. Available: https://www.scribbr.com/methodology/case-study/ (visited on 04/27/2021).

[78] Scribbr. (Mar. 6, 2021). "Research methods | definitions, types, examples," Scribbr, [Online]. Available: https://www.scribbr.com/category/methodology/ (visited on 03/06/2021).

[79] M. Orgill, "Phenomenography," in *Encyclopedia of the Sciences of Learning*, N. M. Seel, Ed., Boston, MA: Springer US, 2012, pp. 2608–2611, ISBN: 978-1-4419-1428-6. DOI: 10.1007/978-1-4419-1428-6_271. [Online]. Available: https://doi.org/10.1007/978-1-4419-1428-6_271 (visited on 04/27/2021).

[80] M. N. P. Johnson and E. McLean. "Critical discourse analysis - an overview | ScienceDirect topics," [Online]. Available: https://www.sciencedirect.com/topics/social-sciences/critical-discourse-analysis (visited on 04/27/2021).

[81] M. M. Cruz-Cunha and I. M. Portela, *Handbook of Research on Digital Crime, Cyberspace Security, and Information Assurance*. Portugal: IGI Global, Jul. 2014, 602 pp., Publication Title: https://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/978-1-4666-6324-4, ISBN: 978-1-4666-6324-4. [Online]. Available: www.igi-global.com/book/handbook-research-digital-crime-cyberspace/104750 (visited on 05/23/2021).

[82] D. Sule. "Forensic readiness and eDiscovery: Security & forensics book chapter | IGI global," [Online]. Available: https://www.igi-global.com/chapter/forensic-readiness-and-ediscovery/115757 (visited on 04/22/2021).

[83] "IMEI CHECK - free online IMEI number checker | IMEI.info," [Online]. Available: https://www.imei.info/ (visited on 05/05/2021).

[84] Statista. (Apr. 2020). "Android operating system share worldwide by OS version from 2013 to 2020*," Statista, [Online]. Available: https://www.statista.com/statistics/271774/share-of-android-platforms-on-mobile-devices-with-android-os/ (visited on 03/05/2021).

[85] "Supported iPhone models," Apple Support, [Online]. Available: https://support.apple.com/guide/iphone/supported-iphone-models-iphe3fa5df43/13.0/ios/13.0 (visited on 03/05/2021).

[86] *Product overview | cellebrite UFED*. [Online]. Available: https://cf-media.cellebrite.com/wp-content/uploads/2020/06/ProductOverview_Cellebrite_UFED_A4.pdf (visited on 04/07/2021).

[87] "Mobile forensic software comparison chart," [Online]. Available: http://ftp.qbssoftware.com/public/Info/paraben/DeviceSeizureComparisonChart.pdf.

[88] *Product overview | cellebrite physical analyzer*. [Online]. Available: https://cf-media.cellebrite.com/wp-content/uploads/2020/09/ProductOverview_Cellebrite_Physical_Analyzer_A4_web.pdf (visited on 04/07/2021).

[89] "Cellebrite reader," [Online]. Available: https://www.cellebrite.com/en/reader/ (visited on 04/07/2021).

[90] P. Eklund. "XRY logical," MSAB, [Online]. Available: https://www.msab.com/products/xry/xry-logical/ (visited on 04/07/2021).

[91] P. Eklund. "XAMN viewer," MSAB, [Online]. Available: https://www.msab.com/products/xamn/viewer/ (visited on 04/07/2021).

[92] "Forensic express," MOBILedit, [Online]. Available: https://www.mobiledit.com/forensic-express (visited on 04/07/2021).

[93] "Forensic express details," MOBILedit, [Online]. Available: https://www.mobiledit.com/forensic-express/details (visited on 04/07/2021).

[94] "Magnet ACQUIRE," Magnet Forensics, [Online]. Available: https://www.magnetforensics.com/resources/magnet-acquire/ (visited on 04/25/2021).

[95] "Magnet AXIOM," Magnet Forensics, [Online]. Available: https://www.magnetforensics.com/products/magnet-axiom/ (visited on 04/07/2021).

[96] M. Forensics. (May 18, 2016). "Magnet AXIOM process: Streamlining acquisition and processing," Magnet Forensics, [Online]. Available: https://www.magnetforensics.com/blog/magnet-axiom-feature-processing/ (visited on 04/07/2021).

[97] "PTP - picture transfer protocol," [Online]. Available: https://www.abbreviations.com/term/112526 (visited on 05/05/2021).

[98] H. Mahalik, D. L. Crognale, and M. Epifani, *The most relevant evidence per gigabyte*, 2021. [Online]. Available: https://www.sans.org/security-resources/posters/dfir-advanced-smartphone-forensics/30/download (visited on 04/29/2021).

[99] Discord. (May 7, 2021). "Get discord - talk, video chat & hang out with friends 73.8 apk," GET APK APP, [Online]. Available: https://getapk.app/download/com.discord/ (visited on 05/10/2021).

[100] Android Developers. (May 6, 2021). "Building web apps in WebView," Android Developers, [Online]. Available: https://developer.android.com/guide/webapps/webview (visited on 05/11/2021).

[101] "DEPS files," [Online]. Available: https://chromium.googlesource.com/chromium/src/+/master/buildtools/checkdeps/README.md (visited on 05/11/2021).

[102] Fileinfo. (Oct. 16, 2020). "CRYPT12 file extension - what is a .crypt12 file and how do i open it?" [Online]. Available: https://fileinfo.com/extension/crypt12 (visited on 05/12/2021).

[103] “App manifest overview,” Android Developers, [Online]. Available: https://developer.android.com/guide/topics/manifest/manifest-intro (visited on 04/29/2021).

[104] “WAL-mode file format,” [Online]. Available: https://www.sqlite.org/walformat.html (visited on 05/08/2021).

[105] A. Chugh. (Oct. 20, 2015). “Android shared preferences example tutorial,” JournalDev, [Online]. Available: https://www.journaldev.com/9412/android-shared-preferences-example-tutorial (visited on 05/08/2021).

[106] L. Knuth. (May 11, 2013). “Can you write to the sdcard folder when there is no sdcard?” Stack Overflow, [Online]. Available: https://stackoverflow.com/questions/16492087/can-you-write-to-the-sdcard-folder-when-there-is-no-sdcard (visited on 05/09/2021).

[107] Android Developers. (May 4, 2021). “Data and file storage overview,” Android Developers, [Online]. Available: https://developer.android.com/training/data-storage (visited on 05/09/2021).

[108] “TGS file extension - what is it? how to open a TGS file?” [Online]. Available: https://filext.com/file-extension/TGS (visited on 05/05/2021).

[109] “OGG file extension - what is an .ogg file and how do i open it?” [Online]. Available: https://fileinfo.com/extension/ogg (visited on 05/09/2021).

[110] “NOMEDIA file extension - what is a .nomedia file and how do i open it?” [Online]. Available: https://fileinfo.com/extension/nomedia (visited on 05/08/2021).

[111] “NOMEDIA file extension - what is it? how to open a NOMEDIA file?” [Online]. Available: https://filext.com/file-extension/NOMEDIA (visited on 05/08/2021).

[112] “Web application manifest,” [Online]. Available: https://www.w3.org/TR/appmanifest/ (visited on 05/09/2021).

[113] mac4n6. (Jan. 6, 2019). “Network and application usage using netusage.sqlite & DataUsage.
sqlite iOS databases,” mac4n6.com, [Online]. Available: http://www.mac4n6.com/blog/2019/1/6/network-and-application-usage-using-netusagesqlite-amp-datausagesqlite-ios-databases (visited on 05/03/2021).

[114] A. Vamshi. (Nov. 4, 2020). “Leaky chats: Accidental exposure and malware in discord attachments,” Netskope, [Online]. Available: `https://www.netskope.com/blog/leaky-chats-accidental-exposure-and-malware-in-discord-attachments` (visited on 05/03/2021).

[115] I. Digitalguide. (Sep. 5, 2019). “Thumbnails – little pictures, lots of power,” IONOS Digitalguide, [Online]. Available: `https://www.ionos.com/digitalguide/online-marketing/social-media/what-is-a-thumbnail/` (visited on 05/05/2021).

Appendices

Appendix 1 - Comparison of Messaging Applications

Table 3. Comparison of messaging applications.

Application	Main features	Supported mobile OS	Uses end-to-end encryption	Stores messages in the server
Discord	Text chat; voice and video calls; image, audio and video sharing for groups, streaming	Android, iOS	No	Yes
Facebook Messenger	Text chat; voice and video calls; image, audio and video sharing	Android, iOS, Windows Phone	In secret conversation mode	Yes
Kik	Text chat; image, audio and video sharing	Android, iOS	No	No
Signal	Text chat; voice and video calls; image, audio and video sharing	Android, iOS	Yes	No
Skype	Text chat; voice and video calls; image, audio and video sharing	Android, iOS, Windows Phone	No	Yes

Continues...

Table 3 – *Continues...*

Application	**Main features**	**Supported mobile OS**	**Uses end-to-end encryption**	**Stores messages in the server**
Slack	Text chat; voice and video calls; image, audio and video sharing	Android, iOS, Windows Phone	No	Yes
Telegram	Text chat; voice and video calls; image, audio and video sharing	Android, iOS, Windows Phone	No	Yes
Viber	Text chat; voice and video calls; image, audio and video sharing	Android, iOS, Windows Phone	Yes	No
WhatsApp	Text chat; voice and video calls; image, audio and video sharing	Android, iOS, Windows Phone	Yes	No
Wickr Me	Text chat; voice and video calls; image, audio and video sharing	Android, iOS	Yes	No

Appendix 2 - Tools Used

Table 4. Devices and tools used for the practical part.

Device/Tool	**Use**	**Company**	**Software/OS version**
Laptop	View backup files	HP	Windows 10
Samsung Galaxy S9+	Smartphone	Samsung	Android 10
iPhone 6	Smartphone	Apple	iOS 12.5.1
Cellebrite UFED	Logical image creator	Cellebrite	7.42.0.82
Cellebrite Physical Analyzer	Logical image analyzer	Cellebrite	12.5.1
Cellebrite Reader	Logical image viewer	Cellebrite	7.42.0.50
MSAB XRY Logical	Logical image creator	Micro Systemation AB (MSAB)	9.4
MSAB XAMN Viewer Logical	Logical image viewer	Micro Systemation AB (MSAB)	5.2.0
MOBILedit	Logical image creator and viewer	Compelson Labs	7.3.0.19270[1]
Magnet Acquire[2]	Logical image extractor	Magnet Forensics	2.37.0.24776
AXIOM Process	Logical image creator	Magnet Forensics	4.11.0.24063
AXIOM Examine	Logical image viewer	Magnet Forensics	4.11.0.24063
DB Browser for SQLite	For reading database files	-	3.12.1
iBackup Viewer	For reading iOS files	-	4.18.4.0

[1]New MOBILedit version was released on 2021-04-09.
[2]New Magnet AXIOM 5.0 was released on 2021-05-04.

Table 5. Information about smartphones used for the thesis.

Technical information	**Android**	**Apple iPhone 6 TD-LTE (A1586)**
Software version	10	12.5.1
Model name	Galaxy S9 Plus	iPhone 6
Model number	SM-G965F/DS	MG482FS/A
Serial number	213afaef16017ece	F17RXA1NG5MP
Capacity	64 GB	16 GB
IMEI	56626092089582; 56627092089580	35 541207 498440 0
Carrier	SUPER	EMT 36.0
Rooted/unrooted	Unrooted	Not jailbroken

Appendix 3 - Discord

Table 6. Details of Discord application.

	Android	**iOS**
Package Name	com.discord	com.hammerandchisel.discord
Version	67.12	65.0

Table 7. Discord artifacts in Android phone.

Path to a folder or file	**Description**
/data/app/com.discord-g3yGVx5i73im9OjySeWLpg==/base.apk	App package for distribution
/data/app/com.discord-g3yGVx5i73im9OjySeWLpg==/base.apk/AndroidManifest.xml	Essential information about the app
/data/app/com.discord-g3yGVx5i73im9OjySeWLpg==/base.apk/base.apk_embedded_7.jpg	Image in one colour
/data/data/com.android.vending/databases/localappstate.db	Database
/data/data/com.discord_.app.AppActivity$Main	Launcher [99]
/data/data/com.discord_preferences	Preferences
/data/data/com.discord.apk	App package for distribution
/data/data/com.discord/description.info	Basic information about the app
/data/data/com.discord/description.info.xml	Information about the app in XML format
/data/data/com.discord/icon.png	The icon of the app
/data/data/com.google.android.gm/databases/bigTopDataDB.1637290733	Table: item_messages
/data/data/com.samsung.android.providers.contacts/databases/calllog.db-wal	Write-ahead log. Table: calls

Continues...

Table 7 – *Continues...*

Path to a folder or file	Description
/data/user_de/0/com.android.providers.telephony/databases/mmssms.db	SMS/MMS

Table 8. Discord artifacts in iOS phone.

Path to a folder or file	Description
/private/var/mobile/Containers/Data/Application/3F79298C-31ED-4174-A6DE-F987942FC243	com.hammerandchisel.discord
/private/var/mobile/Containers/Data/Application/com.hammerandchisel.discord/Library/Cookies/Cookies.binarycookies	Cookies
/private/var/mobile/Library/CallHistoryDB/CallHistory.storedata	Call log [98]. Table: ZCALL-RECORD
/private/var/mobile/Library/CoreDuet/People/interactionC.db	Application traces [98]. Table: ZINTERACTIONS, ZCON-TACTS
/private/var/mobile/Library/SMS/sms.db	SMS/MMS messages [98]. Table: message, handle
`https://media.discordapp.net/attachments/828740250414219314/828748033360658432/video0.mp4`	Outgoing video attachment
`https://cdn.discordapp.com/attachments/828740250414219314/828748033360658432/video0.mp4`	Outgoing video attachment download
`https://media.discordapp.net/attachments/828740250414219314/828748033348206622/image0.jpg`	Outgoing image
`https://cdn.discordapp.com/attachments/828740250414219314/828748033348206622/image0.jpg`	Outgoing image download

Appendix 4 - Facebook Messenger

Table 9. Details of Facebook Messenger application.

	Android	**iOS**
Package Name	com.facebook.orca	com.facebook.Messenger
Version	306.0.0.17.114	306.1

Table 10. Facebook Messenger artifacts in Android phone.

Path to a folder or file	**Description**
/data/app/com.facebook.system-9OQaAkThgBTtpFtFoezfXw==/base.apk	App package for distribution
/data/data/com.facebook.orca/app_analytics_beacon/normal/	Analytics
/data/data/com.facebook.orca/app_analytics_beacon/high/	Analytics
/data/data/com.facebook.orca/app_analytics/micro_batch/com.facebook.orca/100065384016508/18724/449394/	Analytics
/data/data/com.facebook.orca/app_analytics/normal/com.facebook.orca/100065384016508/18711/449081/	Analytics
/data/data/com.facebook.orca/app_appcomponents/versions/	-
/data/data/com.facebook.orca/app_browser_proc_webview/Default/	Related to the web application. Cookies
/data/data/com.facebook.orca/app_browser_proc_webview/Default/GPUCache/	Related to the web application
/data/data/com.facebook.orca/app_browser_proc_webview/Default/GPUCache/index-dir/	Related to the web application
/data/data/com.facebook.orca/app_browser_proc_webview/Default/Local Storage/leveldb/	Related to the web application. Logs

Continues...

Table 10 – *Continues...*

Path to a folder or file	**Description**
/data/data/com.facebook.orca/ app_browser_proc_webview/Default/Session Storage/	Related to the web application
/data/data/com.facebook.orca/app_errorreporting/ crashlog/	Error reporting
/data/data/com.facebook.orca/app_errorreporting/ reports/	Error reporting
/data/data/com.facebook.orca/app_errorreporting/ sess_browser_000000001-1616689634240-c68283b2-0daf-52d5-81d0-05ef4b8ba402/	Error reporting
/data/data/com.facebook.orca/app_errorreporting/ sess_fwkstartlog_000000009-1617777015092-bf18a763-ef13-d2d0-2fb2-3cd99f34bb02/	Error reporting
/data/data/com.facebook.orca/app_errorreporting/ sess_videoplayer_000000001-1616689414290-4b744795-9baf-77a3-6e06-c3a0b980eb05/	Error reporting
/data/data/com.facebook.orca/app_errorreporting/ sess__000000028-1617861289413-1a848242-72da-a64d-809a-abfda5b43294/	Error reporting
/data/data/com.facebook.orca/ app_file_poolcollector/	-
/data/data/com.facebook.orca/ app_file_poolreports/	-
/data/data/com.facebook.orca/ app_funnel_analytics_beacon/overall/	Analytics
/data/data/com.facebook.orca/ app_funnel_backup/com.facebook.orca/	-
/data/data/com.facebook.orca/ app_funnel_reliability_counters/com.facebook.orca/	-
/data/data/com.facebook.orca/app_image/	Contains images related to the app in jpeg format. Has images related to the user and also some unknown images.
/data/data/com.facebook.orca/app_gatekeepers/	-
/data/data/com.facebook.orca/app_gatekeepers/users/	-

Continues...

Table 10 – *Continues...*

Path to a folder or file	**Description**
/data/data/com.facebook.orca/app_graphservice/ graph_metadata.bin	-
/data/data/com.facebook.orca/ app_graph_service_cache/100065384016508/	Cache
/data/data/com.facebook.orca/app_light_prefs/ com.facebook.orca/authentication	Preferences
/data/data/com.facebook.orca/app_light_prefs/ com.facebook.orca:browser/	Preferences
/data/data/com.facebook.orca/app_light_prefs/ com.facebook.orca:videoplayer/	Preferences
/data/data/com.facebook.orca/app_qpl/	-
/data/data/com.facebook.orca/app_ras_blobs/ FacebookEmoji.ttf/	Facebook Messenger emoji
/data/data/com.facebook.orca/ app_sessionless_gatekeepers/	-
/data/data/com.facebook.orca/ app_videocache_logging/	Video cache logs
/data/data/com.facebook.orca/app_webview/Default/	Related to the web application
/data/data/com.facebook.orca/app_webview/Default/ Cookies	Related to the web application. Table: cookies
/data/data/com.facebook.orca/ app_xma_dash_disk_cache/	Cache
/data/data/com.facebook.orca/cache/fb_temp/	Contains temporary files for images and videos sent through the application. The storage time of these files is unclear [17].
/data/data/com.facebook.orca/cache/image/	Contains multiple folders with image and video files [17].
/data/data/com.facebook.orca/com.facebook.orca.apk	App package for distribution
/data/data/com.facebook.orca/databases/ contact_ranking_db	Database
/data/data/com.facebook.orca/databases/ contact_ranking_db-journal	File for database management

Continues...

Table 10 – *Continues...*

Path to a folder or file	**Description**
/data/data/com.facebook.orca/databases/ contact_ranking_db-uid	Contains the unique identifier of the database
/data/data/com.facebook.orca/databases/ contacts_db2	Contains information about the contacts that the user has added and that are in the phonebook. Only these phonebook contacts are in the database that use Facebook Messenger [17].
/data/data/com.facebook.orca/databases/ contacts_db2-journal	File for database management
/data/data/com.facebook.orca/databases/contacts_db2-uid	Contains the unique identifier of the database
/data/data/com.facebook.orca/databases/graph_cursors	-
/data/data/com.facebook.orca/databases/graph_cursors-journal	File for database management
/data/data/com.facebook.orca/databases/graph_cursors-uid	Contains the unique identifier of the database
/data/data/com.facebook.orca/databases/ inbox_units_db	-
/data/data/com.facebook.orca/databases/ inbox_units_db-journal	File for database management
/data/data/com.facebook.orca/databases/ inbox_units_db-uid	Contains the unique identifier of the database
/data/data/com.facebook.orca/databases/ location_sharing.db	Contains data about shared locations
/data/data/com.facebook.orca/databases/matching_db	-
/data/data/com.facebook.orca/databases/ messaging_emoji_db	Emoji database
/data/data/com.facebook.orca/databases/ msys_database_100065384016508	-
/data/data/com.facebook.orca/databases/ offline_mode_db	-
/data/data/com.facebook.orca/databases/ omnistore_100065384016508_v01.db	-

Continues...

Table 10 – *Continues...*

Path to a folder or file	**Description**
/data/data/com.facebook.orca/databases/prefs_db	Contains metadata about the app and the account [17].
/data/data/com.facebook.orca/databases/savedvideos.db	
/data/data/com.facebook.orca/databases/savedvideos.db-journal	File for database management
/data/data/com.facebook.orca/databases/search_cache_db	
/data/data/com.facebook.orca/databases/search_cache_db-journal	File for database management
/data/data/com.facebook.orca/databases/search_cache_db-uid	Contains the unique identifier of the database
/data/data/com.facebook.orca/databases/smstakeover_db	-
/data/data/com.facebook.orca/databases/stickers_db	Contains data about stickers. Table: stickers
/data/data/com.facebook.orca/databases/threads_db2	Contains data about messages [17]. Table: messages, threads
/data/data/com.facebook.orca/databases/tincan_db_100065384016508	-
/data/data/com.facebook.orca/description.info	Basic information about the app
/data/data/com.facebook.orca/description.info.xml	Information about the app in XML format
/data/data/com.facebook.orca/dex/deps	Specifies "which files the sources in a directory tree may include" [101]
/data/data/com.facebook.orca/dex/mdex_lock	-
/data/data/com.facebook.orca/dex/mdex_status2	-
/data/data/com.facebook.orca/dex/odex_lock	-
/data/data/com.facebook.orca/dex/optimization_history_log	A log file
/data/data/com.facebook.orca/dex/regen_stamp	-
/data/data/com.facebook.orca/icon.png	The icon of the app

Continues...

Table 10 – *Continues...*

Path to a folder or file	**Description**
/data/data/com.facebook.orca/files/audio/ v2.ols100.1/76/	Audio files
/data/data/com.facebook.orca/files/batterymetrics/ main_process	-
/data/data/com.facebook.orca/files/batterymetrics/ metrics_com.facebook.orca_browser	-
/data/data/com.facebook.orca/files/batterymetrics/ metrics_com.facebook.orca_fwkstartlog	-
/data/data/com.facebook.orca/files/ ExoPlayerCacheDir/videocache/0/	Cache
/data/data/com.facebook.orca/files/GkBootstrap/	-
/data/data/com.facebook.orca/files/http/historical/ orca_network_map	-
/data/data/com.facebook.orca/files/mobileconfig/	Contains different configuration files
/data/data/com.facebook.orca/files/mobileconfig/ sessionless.data/	Contains different configuration files
/data/data/com.facebook.orca/files/mobileconfig/ 100065384016508.data/	Contains different configuration files
/data/data/com.facebook.orca/files/looper_xplat/ 100065384016508/	-
/data/data/com.facebook.orca/files/profilo/	-
/data/data/com.facebook.orca/files/stickers/	Facebook Messenger stickers
/data/data/com.facebook.orca/files/strings/en_GB-281615725-59be190e1d918a766e527ed249f74ecf. langpack	Language package
/data/data/com.facebook.orca/lib-assets/	Libraries
/data/data/com.facebook.orca/lib-main/	Libraries
/data/data/com.facebook.orca/lib-superpack-br/	Libraries
/data/data/com.facebook.orca/lib-superpack-zstd/	Libraries
/data/data/com.facebook.orca/lib-superpack-xz/	Libraries
/data/data/com.facebook.orca/lib-zstd/	Libraries
/data/data/com.facebook.orca/lib-xzs/	Libraries
/data/data/com.facebook.orca/_manifest	Manifest file
/data/data/com.facebook.orca/modules/	Logs

Continues...

Table 10 – *Continues...*

Path to a folder or file	**Description**
/data/data/com.facebook.orca/modules/effects_0/dex/	-
/data/data/com.facebook.orca/modules/openh264_0/lib-compressed/arm64-v8a/	Libraries
/data/data/com.facebook.orca/shared_prefs/	Preferences
file:///data/user/0/com.facebook.orca/cache/fb_temp/USER_SCOPED_TEMP_DATA_MSGR_VIDEO_FOR_UPLOAD_1617657109340_6784945684334738739.mp4	Video file
`https://external.xx.fbcdn.net/static_map.php?v=2012&osm_provider=2&ccb=4-4&size=545x280&zoom=15&markers=58.37151905,24.53483634&language=en`	Shared location image
`https://l.facebook.com/l.php?u=https://maps.google.com/maps?q=58.37151905%2C24.53483634&hl=en&h=AT1Yn0KGRU8eB2a0Z1bjg_PArAEYMslTG2ND8Z2KbNGVFvbiQSOqsiWnSjcG02l_c4Tc9oNR8l0jNqDx-4sat7tc3dvNUkNuLYfRvw1RiaH1ool2IlSlqbzI-nI8P_JC1GLuALMu7VeOv8M&s=1`	Shared location in Google Maps
`https://scontent.ftll2-1.fna.fbcdn.net/v/t1.6435-1/cp0/e15/q65/p130x130/169323772_119186263594612_2642261821810956631_n.jpg?_nc_cat=102&ccb=1-3&_nc_sid=dbb9e7&_nc_ohc=LnscgSi4ywYAX-6PmOj&_nc_ad=z-m&_nc_cid=0&_nc_ht=scontent.ftll2-1.fna&tp=3&_nc_rmd=260&oh=dc0b95abcfb739fdcbab93af591b9f9e&oe=60906B19`	URL signature has expired

Table 11. Facebook Messenger artifacts in iOS phone.

Path to a folder or file	**Description**
Apple_iPhone 6 (A1586).zip/Backup Service/7b0f88aab509380622546f45cd9c3955443c05fe/ Snapshot/Manifest.plist	Log entries
/private/var/mobile/Containers/Data/Application/ 8A258FB5-F088-483C-A606-3F6D7529DF62/	com.facebook.Messenger
/private/var/mobile/Containers/Data/Application/ com.facebook.Messenger/Library/Cookies/ Cookies.binarycookies	Cookies
/private/var/mobile/Containers/Shared/AppGroup/ group.com.facebook.Messenger/lightspeed-100065097170555.db	Table: messages, attachments, contacts
/private/var/mobile/Library/CoreDuet/People/ interactionC.db	Application traces. Table: ZINTERACTIONS
/private/var/wireless/Library/Databases/ DataUsage.sqlite	Application traces [98]. Table: ZLIVEUSAGE, ZPROCESS
`https://msngr.com/mknlntjbvlmf`	Link to Messenger room that was found from WhatsApp chat
https://www.facebook.com/rsrc.php/v3/y1/r/ zmBgmlzGnQE.png	Unknown image
`https://cdn.fbsbx.com/v/t59.2708-21/161409581_745863096126025_9034018691516572145_n.mp4?_nc_cat=1&ccb=1-3&_nc_sid=041f46&_nc_ohc=tJCiCjz7R8oAX8oqW-K&_nc_ht=cdn.fbsbx.com&oh=6ff7268e498f758ef49923f8873af0f4&oe=606D0616`	Attachment. The URL signature has expired
`https://scontent.xx.fbcdn.net/v/t1.6435-1/p120x120/169323772_119186263594612_2642261821810956631_n.jpg?_nc_cat=102&ccb=1-3&_nc_sid=7206a8&_nc_ohc=LnscgSi4ywYAX8InFmo&_nc_ad=z-m&_nc_cid=0&_nc_ht=scontent.xx&tp=6&oh=9131c612d75e64baf2e72208a24e4591&oe=60924D3D`	Profile picture

Continues...

Table 11 – *Continues...*

Path to a folder or file	**Description**
`https://scontent.ftll2-1.fna.fbcdn.net/v/t1.6435-1/cp0/e15/q65/c0.3.120.120a/p120x120/169413450_112997430889724_6668785669617385689_n.jpg?_nc_cat=101&ccb=1-3&_nc_sid=7206a8&_nc_ohc=wFfY8dgB91kAX8lqY2a&_nc_ad=z-m&_nc_cid=0&_nc_ht=scontent.ftll2-1.fna&tp=5&_nc_rmd=260&oh=726153756a18dbb5baa7dff8d6da7ba1&oe=608F5B96`	Contact profile picture
`https://scontent.xx.fbcdn.net/v/t39.1997-6/851576_553802514654880_121410364_n.webp?_nc_cat=1&ccb=1-3&_nc_sid=0572db&_nc_ohc=cYZT8OyK9K4AX8GCYcx&_nc_ad=z-m&_nc_cid=0&_nc_ht=scontent.xx&oh=40deac37a2c9651c555816f1e4a290a1&oe=608FDE96`	GIF
`https://scontent.xx.fbcdn.net/v/t1.15752-9/fr/cp0/e15/q65/169279193_150936500271670_5375304289316079412_n.jpg?_nc_cat=105&ccb=1-3&_nc_sid=58c789&_nc_ohc=qnNOBFzY3z8AX-VdvNq&_nc_ad=z-m&_nc_cid=0&_nc_ht=scontent.xx&tp=14&oh=2aee4d565ff73c73e642708587e776c2&oe=608FE184`	Attachment
`https://scontent.xx.fbcdn.net/v/t39.1997-6/39219952_1505096246303779_1177236259362308096_n.webp?_nc_cat=1&ccb=1-3&_nc_sid=0572db&_nc_ohc=rQlHDB111wwAX95gY5k&_nc_ad=z-m&_nc_cid=0&_nc_ht=scontent.xx&oh=61bedc7a1cf232f374d06ea7aac51ec5&oe=60927151`	Thumbs up icon

Continues...

Table 11 – *Continues...*

Path to a folder or file	**Description**
`https://video.xx.fbcdn.net/v/t42.3356-2/167967550_893464211216485_4448107510414656691_n.mp4/video-1617656758.mp4?_nc_cat=101&ccb=1-3&_nc_sid=060d78&_nc_ohc=_3Fc-uzx0jsAX-dd46E&vabr=300016&_nc_ht=video.xx&oh=2a6269fb09d07b2133e281b1591e1929&oe=606C8BC3&dl=1`	Video attachment. The URL signature has expired

Appendix 5 - Kik

Table 12. Details of Kik application.

	Android	**iOS**
Package Name	kik.android	com.kik.chat
Version	15.32.0.23731	15.26.0.15876

Table 13. Kik artifacts in Android phone.

Path to a folder or file	**Description**
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/kik.android/kik.android.apk	App package for distribution
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/kik.android/description.info	Basic information about the app
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/kik.android/description.info.xml	Information about the app in XML format
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/kik.android/icon.png	The icon of the app
/data/app/kik.android-3rMlb2aouVFXxvl83fCu4g==/base.apk	App package for distribution
/data/app/kik.android-3rMlb2aouVFXxvl83fCu4g==/base.apk/AndroidManifest.xml	Essential information about the app
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/cache/	Cache

Continues...

Table 13 – *Continues...*

Path to a folder or file	**Description**
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/cache/chatPicsBig/	Cache for images
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/cache/chatVids/	Cache for received or sent videos
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/cache/contentpics/	Cache for received or sent pictures. Contains some unknown pictures
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/cache/emojis/	Cache
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/cache/gifs/	Cache
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/cache/gifs_/	Cache
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/cache/profpics/	Cache. Profile pictures
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/cache/profpics/-7263415841623653164/	Cache. Profile pictures
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/cache/sponsoredresponse/	Cache
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/cache/widget_screenshots/	Cache
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/files/smileys/	Smileys
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/kik_content_preview_cache/ 1509540283-1682335973/	Cache. A received or sent picture that was drawn in the app was found here
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/okhttp_cache/	Cache
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/volleyCardsIcons/	Icons
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/xdata_cache/	Cache
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/xdata_cache/browser_history_item_list/	Cache

Continues...

Table 13 – *Continues...*

Path to a folder or file	**Description**
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/xdata_cache/smiley_list/	Cache
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/xdata_cache/sticker_pack/	Cache
/data/data/kik.android/app_com.quantcast/	-
/data/data/kik.android/app_pccache/5/ 077382A9BEFAEBD913 AE4F3B3D6F5850560C7C27/	Cache
/data/data/kik.android/app_pccache/5/ 077382A9BEFAEBD913 AE4F3B3D6F5850560C7C27/pcam.jar/	Cache. AndroidManifest.xml, META-INF
/data/data/kik.android/app_pccache/5/ 077382A9BEFAEBD913 AE4F3B3D6F5850560C7C27/pcam.jar/META-INF/	Cache
/data/data/kik.android/app_pccache/5/ 077382A9BEFAEBD913 AE4F3B3D6F5850560C7C27/oat/	Cache
/data/data/kik.android/app_webview/	metrics_guid, pref_store files
/data/data/kik.android/app_webview/Default/ Application Cache/Cache/ba23d8ecda68de77_0/	Cache
/data/data/kik.android/app_webview/Default/ Application Cache/Cache/index-dir/	Cache
/data/data/kik.android/app_webview/Default/ databases/	Related to the web application
/data/data/kik.android/app_webview/Default/ GPUCache/	Related to the web application
/data/data/kik.android/app_webview/Default/Local Storage/leveldb/	Related to the web application. Contains log files
/data/data/kik.android/app_webview/Default/Session Storage/	Related to the web application. Contains log files
/data/data/kik.android/databases/kikDatabase.db	-
/data/data/kik.android/databases/kikDatabase.db-journal	-

Continues...

Table 13 – *Continues...*

Path to a folder or file	**Description**
/data/data/kik.android/databases/42ddbcd6-f318-4b65-97f8-6e74a98b02dc.kikDatabase.db	Table: KIKcontactsTable, messagesTable, KikFriendAttributionTableName, KIKContentTable. Contact
/data/data/kik.android/files/	JSON files, LOCK file. QC-SessionId
/data/data/kik.android/files/staging/thumbs/	Thumbnails
/data/data/kik.android/files/.com.google.firebase.crashlytics/	JSON files
/data/data/kik.android/files/.com.google.firebase.crashlytics/report-persistence/sessions/606EB62903B500014C3B31EA6AFDB25A/	-
/data/data/kik.android/data/data/kik.android/files/.com.google.firebase.crashlytics-ndk/606C08340232-0001-1C00-31EA6AFDB25A/	-
/data/data/kik.android/_manifest	-
/data/data/kik.android/shared_prefs/	Preferences
/data/data/kik.android/thirdpartyfiles/	Gif and a picture of the gif that was sent or received
/data/data/kik.android/databases/	.alternatesTable, .alternatesTable-journal, .db, .db-journal, .smileyTable, .smileyTable-journal, .events, .events-journal, .db-shm, .db-wal files. Contains SQLite, SQLite Shared Memory, SQLite Write-Ahead Log and other files
/data/data/kik.android/cache/WebView/Default/HTTP Cache/2927b896778c090c_0/2927b896778c090c_0_embedded_1.jpg	Image
/data/data/temp/kikTmpOriginalPicFile	-
/data/user/0/kik.android/cache/	Cache
/data/user/0/kik.android/cache/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/tempVids/	Cahce

Table 14. Kik artifacts in iOS phone.

Path to a folder or file	**Description**
/private/var/mobile/Containers/Data/Application/ 0DECD505-53FB-4437-8D6A-B51C334862F0	com.kik.chat
/private/var/mobile/Containers/Shared/AppGroup/ group.com.kik.chat/cores/private/f6052205ad6c4910 a4a2c133477b01a9/profpix/	Known and unknown profile pictures
/data/data/kik.android/42ddbcd6-f318-4b65-97f8-6e74a98b02dc/cache/profpics/	-
/private/var/mobile/Containers/Shared/AppGroup/ group.com.kik.chat/cores/private/f6052205ad6c4910 a4a2c133477b01a9/attachments/	Attachments
/private/var/mobile/Containers/Shared/AppGroup/ group.com.kik.chat/cores/private/f6052205ad6c4910 a4a2c133477b01a9/content_manager/data_cache/	Pictures and other media files
/private/var/mobile/Containers/Shared/AppGroup/ group.com.kik.chat/cores/private/f6052205ad6c4910 a4a2c133477b01a9/kik.sqlite	Table: ZKIKMESSAGE, ZKIKUSER, ZKIKATTACH-MENT
/private/var/wireless/Library/Databases/ DataUsage.sqlite	Application traces. Table: ZLIVEUSAGE, ZPROCESS
Apple_iPhone 6 (A1586).zip/Backup Service/ 7b0f88aab509380622546f45cd9c3955443c05fe/ Snapshot/Manifest.plist	Log entries
/private/var/mobile/Containers/Data/Application/ com.kik.chat/Library/Cookies/Cookies.binarycookies	Cookies
/private/var/mobile/Containers/Shared/AppGroup/ group.com.kik.chat/Library/Preferences/ group.com.kik.chat.plist	-
/private/var/mobile/Containers/Shared/AppGroup/ group.com.kik.chat/cores/private/ f6052205ad6c4910a4a2c133477b01a9/profpix/ thumb_kikteam@talk.kik.com	Contact profile picture thumb-nail
`http://profilepics.cf.kik.com/9wG3z RZW8sLxLnpmyOfwNE7ChYk`	Contact profile picture. Ac-cess denied

Continues...

Table 14 – *Continues...*

Path to a folder or file	**Description**
`https://platform.kik.com/content/files/2d5ae490-96d7-426b-99d4-ca18688835f5?k=578029136a02d05e30e7e30913d8cac1d8d3f4dc`	Video URL
`https://platform.kik.com/content/files/0a96fbae-b0c9-41a5-9f50-7183045a37ef?k=cc99a259731a007713a20f45fad8a67de93b4b00`	Image URL.
`https://platform.kik.com/content/files/f7f151e6-e536-452c-bff3-5bb7ace42a57?t=kdx_6aenHWsMMeSkCNAuslvkrliPqcfpPoWLGvPKnpPN7LfLkLFdBNLMAqO3x30yknBRYXf69nZmPX4FT5dm3eEjmDP6zNJW--3x5u9AtbKh-CgJ3vu1RYgJddGimyEE`	Image URL

Appendix 6 - Signal

Table 15. Details of Signal application.

	Android	**iOS**
Package Name	org.thoughtcrime.securesms	org.whispersystems.signal
Version	5.5.5	5.8.1

Table 16. Signal artifacts in Android phone.

Path to a folder or file	**Description**
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/org.toughtcrime.securesms/org.thoughtcrime.securesms.apk	App package for distribution
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/org.toughtcrime.securesms/description.info	Basic information about the app
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/org.toughtcrime.securesms/description.info.xml	Information about the app in XML format
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/org.toughtcrime.securesms/icon.png	The icon of the app
/data/app/org.thoughtcrime.securesms-_lOu5HhRl1hYs52XJfOumQ==/base.apk	App package for distribution
/data/app/org.thoughtcrime.securesms-_lOu5HhRl1hYs52XJfOumQ==/base.apk/AndroidManifest.xml	Essential information about the app
/data/data/org.thoughtcrime.securesms/app_avatars/	Avatars

Continues...

Table 16 – *Continues...*

Path to a folder or file	**Description**
/data/data/org.thoughtcrime.securesms/app_parts/	MMS files
/data/data/org.thoughtcrime.securesms/app_stickers/	Stickers in MMS type
/data/data/org.thoughtcrime.securesms/databases/ com.google.android.datatransport.events-journal	-
/data/data/org.thoughtcrime.securesms/files/ generatefid.lock	-
/data/data/org.thoughtcrime.securesms/files/ PersistedInstallation.W0RFRkFVTFRd+MTozMTIz MzQ3NTQyMDY6YW5kcm9pZDphOTI5N2IxNTI4N zlmMjY2.json	JSON file
/data/data/org.thoughtcrime.securesms/_manifest	-
/data/data/org.thoughtcrime.securesms/shared_prefs/	Preferences
/data/user_de/0/com.android.providers.telephony/ databases/mmssms.db	SMS/MMS

Table 17. Signal artifacts in iOS phone.

Path to a folder or file	**Description**
/private/var/mobile/Containers/Data/Application/ com.estmob.paprika/Library/Cookies/ Cookies.binarycookies	Cookies
/private/var/mobile/Library/CoreDuet/People/ interactionC.db	Application traces. Table: ZINTER-ACTIONS, ZCONTACTS
/private/var/mobile/Library/SMS/sms.db	SMS/MMS. Table: message, handle
/private/var/wireless/Library/Databases/ DataUsage.sqlite	Application traces . Table: ZLIVEUSAGE, ZPROCESS
Apple_iPhone 6 (A1586).zip/Backup Service/ 7b0f88aab509380622546f45cd9c3955443c05fe/ Snapshot/Manifest.plist	Log entries
/private/var/mobile/Applications/ org.whispersystems.signal	org.whispersystems.signal

Appendix 7 - Skype

Table 18. Details of Skype application.

	Android	**iOS**
Package Name	com.skype.raider	com.skype.skype
Version	8.70.0.77	8.69.0.85

Table 19. Skype artifacts in Android phone.

Path to a folder or file	**Description**
/com.google.android.webview/com.skype.raider/ com.skype.raider.apk	App package for distribution
/com.google.android.webview/com.skype.raider/ description.info	Basic information about the app
/com.google.android.webview/com.skype.raider/ description.info.xml	Information about the app in XML format
/com.google.android.webview/com.skype.raider/ icon.png	The icon of the app
/data/data/com.skype.raider/com.skype.raider.apk	App package for distribution
/data/data/com.skype.raider/description.info	Basic information about the app
/data/data/com.skype.raider/description.info.xml	Information about the app in xml format
/data/data/com.skype.raider/icon.png	The icon of the app
/data/app/com.skype.raider-hQ3qofyQYLv2An-_6fzt5g==/base.apk	App package for distribution
/data/app/com.skype.raider-hQ3qofyQYLv2An-_6fzt5g==/base.apk/AndroidManifest.xml	Essential information about the app
/data/data/com.android.vending/databases/ localappstate.db	Table: appstate
/data/data/com.skype.raider/app_phenotype_file/	-
/data/data/com.skype.raider/app_webview/Default/	Related to the web application

Continues...

Table 19 – *Continues...*

Path to a folder or file	**Description**
/data/data/com.skype.raider/app_webview/Default/Cookies	Table: cookies
/data/data/com.skype.raider/app_webview/GPUCache/	Related to the web application
/data/data/com.skype.raider/app_webview/GPUCache/index-dir/	Related to the web application
/data/data/com.skype.raider/app_webview/Local Storage/leveldb/	Related to the web application
/data/data/com.skype.raider/cache/6120114BE9726-AC99B463E854AA456757528F41976FE51C42EF-E7C5DDE5B3A7F/FileCache/processed.jpeg	Profile picture
/data/data/com.skype.raider/cache/image_cache/v2-.ols100.1/61/Urs2FWIESGbimS9PGEFA0qpGhXI.cnt	Cache
/data/data/com.skype.raider/cache/lenssdk_data/edit/documents/0f14f23e-c218-42b3-92de-ef4b000756a5/videoEntity/videoaf0afe5f-d4ee-4630-bf43-41d514c786bbvideo.mp4	Cache
/data/data/com.skype.raider/cache/skype-4228/DbTemp	Cache
/data/data/com.skype.raider/databases/s4l-live&58;.cid.ab1383ae6c66204a.db	Missed incoming calls, contacts, location, chats
/data/data/com.skype.raider/files/CS_files/	-
/data/data/com.skype.raider/files/http-cache/	Cache. Contains pictures, icons
/data/data/com.skype.raider/files/shared.xml	Contains user's username and last IP address [17]
/data/data/com.skype.raider/files/SkypeRT/	Configuration files
/data/data/com.skype.raider/lib-main/	Libraries
/data/data/com.skype.raider/live_external/cache/	Cache
/data/data/com.skype.raider/shared_prefs/	Preferences
/data/media/0/Download/bad-owl-4.mp3	Shared audio file
/data/media/0/Download/bad-owl-habitat.pdf	Shared document
/sdcard/Android/data/com.skype.raider/cache/	Cache
/storage/emulated/0/Android/data/com.skype.raider/cache/	Cache

Continues...

Table 19 – *Continues...*

Path to a folder or file	**Description**
/storage/emulated/0/Android/data/com.skype.raider/ cache/cache_r.0/	Cache
`https://az705183.vo.msecnd.net/dam/skype/media/concierge-assets/avatar/avatarcnsrg-144.png`	Skype icon
`https://avatar.skype.com/v1/avatars/echo123?auth_key=242200701&cacheHeaders=true&returnDefaultImage=false`	Icon
`https://avatar.skype.com/v1/avatars/live%3A.cid.\protect\@normalcr\relax7e5c626cb84f4595?auth_key=-1936285518&cacheHeaders=true&returnDefaultImage=false`	-

Table 20. Skype artifacts in iOS phone.

Path to a folder or file	**Description**
/private/var/mobile/Applications/com.skype.skype	com.skype.skype
/private/var/mobile/Containers/Data/Application/ com.skype.skype/Library/Cookies/ Cookies.binarycookies	Cookies
/private/var/mobile/Library/CallHistoryDB/ CallHistory.storedata	Call log. Table: ZCALL-RECORD

Appendix 8 - Slack

Table 21. Details of Slack application.

	Android	**iOS**
Package Name	com.Slack	com.tinyspeck.chatlyio
Version	21.03.20.0	21.03.20

Table 22. Slack artifacts in Android phone.

Path to a folder or file	**Description**
/com.Slack/com.Slack.apk	App package for distribution
/com.Slack/description.info	Basic information about the app
/com.Slack/description.info.xml	Information about the app in XML format
/com.Slack/icon.png	The icon of the app
/data/app/com.Slack-XLQRSB7OhV4jSdloLrD29Q==/base.apk	App package for distribution
/data/app/com.Slack-XLQRSB7OhV4jSdloLrD29Q==/base.apk/AndroidManifest.xml	Essential information about the app
/data/data/Android/media/com.Slack/Notifications/	Notification audio files
/data/data/com.android.vending/databases/localapp state.db	Table: appstate
/data/data/com.Slack/cache/file-upload/IMG-20210405-WA0006.jpg	Profile photo

Table 23. Slack artifacts in iOS phone.

Path to a folder or file	**Description**
/private/var/mobile/Applications/com.tinyspeck.chatlyio	com.tinyspeck.chatlyio

Continues...

Table 23 – *Continues...*

Path to a folder or file	**Description**
/private/var/wireless/Library/Databases/ DataUsage.sqlite	Application traces. Table: ZLIVEUSAGE, ZPROCESS
Apple_iPhone 6 (A1586).zip/Backup Service/ 7b0f88aab509380622546f45cd9c3955443c05fe/ Snapshot/Manifest.plist	Log entries

Appendix 9 - Telegram

Table 24. Details of Telegram application.

	Android	**iOS**
Package Name	org.telegram.messenger	ph.telegra.Telegraph
Version	7.6.0	7.6.1

Table 25. Telegram artifacts in Android phone.

Path to a folder or file	**Description**
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/org.telegram.messenger/org.telegram.messenger.apk	App package for distribution
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/org.telegram.messenger/description.info	Basic information about the app
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/org.telegram.messenger/description.info.xml	Information about the app in XML format
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/org.telegram.messenger/icon.png	The icon of the app
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/org.telegram.messenger/live_external/cache/	Cache

Continues...

Table 25 – *Continues...*

Path to a folder or file	**Description**
/com.google.android.webview/com.super.selfservice/backup/flipboard.boxer.app/backup/f/.Fabric/appcenter/as_cached_content/error/r/app_data/org.telegram.messenger/live_specific/	Contains Telegram Audio, Documents, Images, Video folders
/data/app/org.telegram.messenger-iQ_usM3jGVolPD-wogi3ow==/base.apk/AndroidManifest.xml	Essential information about the app
/data/app/org.telegram.messenger-iQ_usM3jGVolPD-wogi3ow==/base.apk/base.apk_embedded_7.jpg	Image file
/data/app/org.telegram.messenger-iQ_usM3jGVolPD-wogi3ow==/base.apk/base.apk_embedded_6.jpg	Image file
/data/data/com.android.vending/databases/localappstate.db	Application traces [98]. Table: appstate
/data/data/com.samsung.android.providers.contacts/databases/calllog.db-wal	Write-ahead log. Table: calls
/data/data/org.telegram.messenger/app_phenotype_file/	-
/data/data/org.telegram.messenger/databases/	1) com.google.android.datatransport.events, 2) com.google.android.datatransport.events-journal
/data/data/org.telegram.messenger/description.info	Description file in text and XML format
/data/data/org.telegram.messenger/files/cache4.db	Cache
/data/data/org.telegram.messenger/files/account1/cache4.db	Cache
/data/data/org.telegram.messenger/files/account2/cache4.db	Cache
/data/data/org.telegram.messenger/files/cache4.db-wal	Write-ahead log
/data/data/org.telegram.messenger/files/ShortcutInfoCompatSaver_share_targets/	-
/data/data/org.telegram.messenger/icon.png	App icon
/data/data/org.telegram.messenger/_manifest	-

Continues...

Table 25 – *Continues...*

Path to a folder or file	**Description**
/data/data/org.telegram.messenger/shared_prefs/	Preferences [98]
/data/user_de/0/com.android.providers.telephony/ databases/mmssms.db	SMS/MMS
/storage/emulated/0/Android/data/org.telegram. messenger/cache/	1) TGS files, 2) images, 3) cache_r.0 folder, 4) cache_vts_org.telegram. messenger_default.0
/storage/emulated/0/Telegram/Telegram Audio/	Audio files
/storage/emulated/0/Telegram/Telegram Documents/	Documents
/storage/emulated/0/Telegram/Telegram Images/	Images
/storage/emulated/0/Telegram/Telegram Video/	Video files
/storage/emulated/0/Download/	Downloaded files

Table 26. Telegram artifacts in iOS phone.

Path to a folder or file	**Description**
/private/var/mobile/Library/SMS/sms.db	SMS/MMS messages. Table: message, handle
/private/var/mobile/Library/CoreDuet/People/ interactionC.db	Application traces. Table: ZCONTACTS, ZINTERACTIONS
/private/var/wireless/Library/Databases/ DataUsage.sqlite	Application traces. Table: ZLIVEUSAGE, ZPROCESS. Log entries
Apple_iPhone 6 (A1586).zip/Backup Service/ 7b0f88aab509380622546f45cd9c3955443c05fe/ Snapshot/Manifest.plist	Log entries

Appendix 10 - Viber

Table 27. Details of Viber application.

	Android	**iOS**
Package Name	com.viber.voip	com.viber
Version	15.0.0.0	15.0.0.1

Table 28. Viber artifacts in Android phone.

Path to a folder or file	**Description**
/com.google.android.webview/com.super.selfservice/ backup/com.viber.voip/com.viber.voip.apk	App package for distribution
/com.google.android.webview/com.super.selfservice/ backup/com.viber.voip/description.info	Basic information about the app
/com.google.android.webview/com.super.selfservice/ backup/com.viber.voip/description.info.xml	Information about the app in XML format
/com.google.android.webview/com.super.selfservice/ backup/com.viber.voip/icon.png	The icon of the app
/com.google.android.webview/com.super.selfservice/ backup/com.viber.voip/live_external/cache/	Cache
/com.google.android.webview/com.super.selfservice/ backup/com.viber.voip/live_external/cache/ Image-FetcherThumb/	Thumbnails
/com.google.android.webview/com.super.selfservice/ backup/com.viber.voip/live_external/files/.emoticons/	Emoticons
/com.google.android.webview/com.super.selfservice/ backup/com.viber.voip/live_external/cache/video-cache/	Video cache
/data/app/com.viber.voip– T51yyMLbIMLenaJNFOVOQ==/base.apk	App package for distribution
/data/app/com.viber.voip– T51yyMLbIMLenaJNFOVOQ==/base.apk/ AndroidManifest.xml	Essential information about the app

Continues...

Table 28 – *Continues...*

Path to a folder or file	**Description**
/data/data/com.android.vending/databases/localapp state.db	Table: appstate
/data/data/com.samsung.android.providers.contacts/ databases/calllog.db-wal	Write-ahead log. Table: calls
/data/data/com.viber.voip/app_pccache/5/077382A9B EFAEBD913AE4F3B3D6F5850560C7C27/pcam.jar/ META-INF/	Cache
/data/data/com.viber.voip/app_pccache/5/077382A9B EFAEBD913AE4F3B3D6F5850560C7C27/oat/	Cache
/data/data/com.viber.voip/app_webview/Default/	Cache
/data/data/com.viber.voip/app_webview/Default/ Cookies	Table: cookies
/data/data/com.viber.voip/app_webview/Default/ GPUCache/	Cache
/data/data/com.viber.voip/app_webview/Default/ GPUCache/index-dir/	Cache
/data/data/com.viber.voip/app_webview/Default/Local Storage/leveldb/	Related to the web application
/data/data/com.viber.voip/app_webview/Default/ Session Storage/	Related to the web application
/data/data/com.viber.voip/cache/ab_triggers/-1040279302.jpg	Image
/data/data/com.viber.voip/cache/volley/-42124738-1249527265/-42124738-1249527265_embedded_1.jpg	Image
/data/data/com.viber.voip/databases/viber_data	Contains information about the user's contacts. Table: phonebookcontact, phonebookdata [17]
/data/data/com.viber.voip/databases/viber_messages	Contains information about the app's usage [17]. Table: messages, participants_info
/data/data/com.viber.voip/databases/viber_messages-journal	File for database management. Table: messages

Continues...

Table 28 – *Continues...*

Path to a folder or file	**Description**
/data/data/com.viber.voip/files/.com.google.firebase.crashlytics/	JSON files
/data/data/com.viber.voip/files/.com.google.firebase.crashlytics/log-files/	Logs
/data/data/com.viber.voip/files/preferences/	Preferences
/data/data/com.viber.voip/files/preferences/activated_sim_serial	Preferences
/data/data/com.viber.voip/files/preferences/display_name	Preferences
/data/data/com.viber.voip/files/preferences/reg_viber_phone_num	Preferences
/data/data/com.viber.voip/files/.com.google.firebase.crashlytics-ndk/6063F3C40245-0001-5E5C-4CAB428A4B2A/	JSON files
/data/data/com.viber.voip/files/ShortcutInfoCompatSaver_share_targets/	-
/data/data/com.viber.voip/files/ShortcutInfoCompatSaver_share_targets/ShortcutInfoCompatSaver_share_targets_bitmaps/	-
/data/data/com.viber.voip/shared_prefs/	Preferences
/data/media/0/Android/data/com.viber.voip/files/.emoticons/(paperclip)_scaled_79.png	Emoticon
/data/media/0/Android/data/com.viber.voip/files/.stickers/40100/r144_00040100_orig.png	Sticker
/data/media/0/Movies/Viber/video-5bb97bfd2edae6d9f62b43cacdd471d3-V.mp4	Shared video
/data/media/0/Pictures/Viber/	Shared pictures
/data/user_de/0/com.viber.voip/shared_prefs/	Preferences
/storage/emulated/0/Android/data/com.viber.voip/cache/ImageFetcherThumb/	Thumbnails
/storage/emulated/0/Android/data/com.viber.voip/cache/video-cache/9/	Cache
/storage/emulated/0/Android/data/com.viber.voip/files/.emoticons/	Emoticons

Continues...

Table 28 – *Continues...*

Path to a folder or file	**Description**
/storage/emulated/0/Android/data/com.viber.voip/ files/.gems/	Moving pictures in HTML document
/storage/emulated/0/Android/data/com.viber.voip/ files/.gif/	GIF
/storage/emulated/0/Android/data/com.viber.voip/ files/.image/	-
/storage/emulated/0/Android/data/com.viber.voip/ files/.import/	-
/storage/emulated/0/Android/data/com.viber.voip/ files/.shsh/	-
/storage/emulated/0/Android/data/com.viber.voip/ files/.stickers/5200/	Stickers
/storage/emulated/0/Android/data/com.viber.voip/ files/.temp/	-
/storage/emulated/0/Android/data/com.viber.voip/ files/.thumbnails/	Thumbnails
/storage/emulated/0/Android/data/com.viber.voip/ files/User photos/	Photos
/storage/emulated/0/Android/data/com.viber.voip/ files/User photos/.thumbnails/	Thumbnails
/storage/emulated/0/Android/data/com.viber.voip/ files/.video/	Video files
/storage/emulated/0/Android/media/com.viber.voip/ Notifications/	Notification audio files
/storage/emulated/0/viber/media/.stickers/40100/ .*40100_orig.png	A sticker
/sdcard/viber/media//User Photos/	Photos
/sdcard/viber/media//Viber Images/	Images
/sdcard/viber/media//Viber Videos/	Videos

Table 29. Viber artifacts in iOS phone.

Path to a folder or file	**Description**
/private/var/mobile/Containers/Data/Application/ 9E2EBA1D-D627-4A91-BA88-EF773572C103	com.viber

Continues...

Table 29 – *Continues...*

Path to a folder or file	**Description**
/private/var/mobile/Containers/Data/Application/ com.viber/Documents/Attachments/ 1617661819534021.jpg	Attachment
/private/var/mobile/Containers/Data/Application/ com.viber/Library/Cookies/Cookies.binarycookies	Cookies
/private/var/mobile/Containers/Data/Application/ com.viber/Library/Preferences/com.viber.plist	com.viber.plist
/private/var/mobile/Containers/Shared/AppGroup/ group.viber.share.container/com.viber/database/ Contacts.data	Table: ZVIBERMESSAGE, ZPHONENUMBER, ZMEMBER, ZATTACHMENT, ZABCONTACTNUMBER
/private/var/mobile/Containers/Shared/AppGroup/ group.viber.share.container/com.viber/ViberIcons/	Profile pictures
/private/var/mobile/Containers/Data/Application/ com.viber/Documents/Attachments/	Attachments
/private/var/wireless/Library/Databases/ DataUsage.sqlite	Application traces. Table: ZLIVEUSAGE, ZPROCESS
/private/var/mobile/Library/SMS/sms.db	SMS/MMS messages. Table: message, handle
`https://media.tenor.com/images/6cad22ea49eccb886699a403da413b78/tenor.gif`	GIF
Apple_iPhone 6 (A1586).zip/Backup Service/7b0f88aab509380622546f45cd9c 3955443c05fe/Snapshot/Manifest.plist	Log entries

Appendix 11 - WhatsApp

Table 30. Details of WhatsApp application.

	Android	**iOS**
Package Name	com.whatsapp	net.whatsapp.WhatsApp
Version	2.21.6.17	2.21.50.15

Table 31. WhatsApp artifacts in Android phone.

Path to a folder or file	**Description**
/com.google.android.webview/com.super.selfservice/ backup/com.whatsapp/com.whatsapp.apk	App package for distribution
/com.google.android.webview/com.super.selfservice/ backup/com.whatsapp/description.info	Basic information about the app
/com.google.android.webview/com.super.selfservice/ backup/com.whatsapp/description.info.xml	Information about the app in XML format
/com.google.android.webview/com.super.selfservice/ backup/com.whatsapp/icon.png	The icon of the app
/com.google.android.webview/com.super.selfservice/ backup/com.whatsapp/live_external/cache/	Cache
/com.google.android.webview/com.super.selfservice/ backup/com.whatsapp/live_specific/.Shared/	Related to the web application
/com.google.android.webview/com.super.selfservice/ backup/com.whatsapp/live_specific/Backups/	Related to the web application
/com.google.android.webview/com.super.selfservice/ backup/com.whatsapp/live_specific/Databases/	Related to the web application
/com.google.android.webview/com.super.selfservice/ backup/com.whatsapp/live_specific/Media/.Statuses/	Related to the web application
/com.google.android.webview/com.super.selfservice/ backup/com.whatsapp/live_specific/Media/WhatsApp Animated Gifs/	GIFs

Continues...

Table 31 – *Continues...*

Path to a folder or file	**Description**
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Animated Gifs/Private/	GIFs
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Animated Gifs/Sent/	GIFs
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Audio/	Audio
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Audio/Private/	Audio
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Audio/Sent/	Audio
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Documents/	Documents
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Documents/Private/	Documents
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Documents/Sent/	Documents
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Images/	Images
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Images/Private/	Images
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Images/Sent/	Images
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Stickers/	Stickers

Continues...

Table 31 – *Continues...*

Path to a folder or file	**Description**
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Video/	Videos
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Video/Private/	Videos
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Video/Sent/	Videos
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Voice Notes/	Voice notes
/com.google.android.webview/com.super.selfservice/backup/com.whatsapp/live_specific/Media/WhatsApp Voice Notes/202113/	Voice notes
/data/app/com.whatsapp-wqnyze3k8efLWS_pIBk0Eg==/base.apk	Essential information about the app
/data/app/com.whatsapp-wqnyze3k8efLWS_pIBk0Eg==/base.apk/AndroidManifest.xml	Essential information about the app
/data/data/com.android.vending/databases/localappstate.db	Table: appstate
/data/data/com.samsung.android.providers.contacts/databases/calllog.db-wal	Write-ahead log. Write-ahead log
/data/data/com.whatsapp/app_phenotype_file/	-
/data/data/com.whatsapp/databases/msgstore.db	Table: call_log, chat_list, messages. Location message
/data/data/com.whatsapp/databases/msgstore.db-wal	Write-ahead log. Table: messages
/data/data/com.whatsapp/databases/msgstore.db/IMG-20210405-WA0006.jpg	Table: message_thumbnails
/data/data/com.whatsapp/databases/wa.db	Empty
/data/data/com.whatsapp/databases/wa.db-wal	Write-ahead log. Table: wa_contacts
/data/data/com.whatsapp/files/app_state/	-

Continues...

Table 31 – *Continues...*

Path to a folder or file	**Description**
/data/data/com.whatsapp/files/Avatars/	Profile pictures. For example, me.j or 37257924180@s.whatsapp.net.j
/data/data/com.whatsapp/files/decompressed/libs.spk.zst/	Libraries in ELF (Executable and Linkable Format)
/data/data/com.whatsapp/files/downloadable/	manifest.json
/data/data/com.whatsapp/files/downloadable/filter_YHbwxhPS2U4WtSgbh9e47EKR_cmhYwWErgJoiPpIzuQ/	Filter images
/data/data/com.whatsapp/files/downloadable/wallpaper/thumbnails/dark/	Dark wallpapers
/data/data/com.whatsapp/files/downloadable/wallpaper/thumbnails/light/	Light wallpapers
/data/data/com.whatsapp/files/me.jpeg	Profile picture
/data/data/com.whatsapp/files/Logs/	Logs
/data/data/com.whatsapp/files/.Shared/	Shared files
/data/data/com.whatsapp/files/ViewOnce/.nomedia	-
/data/data/com.whatsapp/files/WhatsApp Images/	Images, .nomedia
/data/data/com.whatsapp/files/WhatsApp Video/	Video files, .nomedia
/data/data/com.whatsapp/lib-main/	-
/data/data/com.whatsapp/shared_prefs/	Preferences
//data/media/0/WhatsApp/Media/.Statuses/62d0f7a0dca74394858d5223ed8e918b.mp4	Video file
/data/media/0/WhatsApp/Databases/msgstore-2021-03-27.1.db.crypt12/msgstore-2021-03-27.1.db	Table: messages
/data/media/0/WhatsApp/Databases/msgstore-2021-03-27.1.db.crypt12/msgstore-2021-03-27.1.db/2fef8c979e3c41eeb4fb5fac1efd593a.jpg	Table: message_thumbnails
/data/media/0/WhatsApp/Databases/msgstore-2021-03-27.1.db.crypt12/msgstore-2021-03-27.1.db	Table: messages
/data/media/0/WhatsApp/Databases/msgstore-2021-03-27.1.db.crypt12/msgstore-2021-03-27.1.db/Ap2hVbW3Da_8idKFxKUVgS7AVbDymv55tXbDVZgCAUE-.enc	Table: message_thumbnails

Continues...

Table 31 – *Continues...*

Path to a folder or file	**Description**
/data/media/0/WhatsApp/Media/.Statuses/ a73d7709b445490b8dac85781f52aea4.jpg	Status image
/data/media/0/WhatsApp/Media/WhatsApp Stickers/STK-20210405-WA0000.webp	Sticker
/data/media/0/WhatsApp/Media/WhatsApp Animated Gifs/Sent/VID-20210405-WA0001.mp4	Video
/data/media/0/WhatsApp/Media/WhatsApp Images/IMG-20210405-WA0003.jpg	Image
/data/media/0/WhatsApp/Media/WhatsApp Video/VID-20210405-WA0005.mp4	Video
/data/media/0/WhatsApp/Media/WhatsApp Documents/Sent/DOC-20210405-WA0010	PDf file
/data/media/0/WhatsApp/Media/WhatsApp Documents/bad-owl-4.mp3	Audio
/data/media/0/WhatsApp/Media/WhatsApp Documents/SA_Beat.mp3	Audio
/data/media/0/WhatsApp/Media/WhatsApp Documents/SA_iTunes.mkv	Video
/data/media/0/WhatsApp/Media/WhatsApp Audio/Sent/AUD-20210405-WA0013	Audio
/data/media/0/WhatsApp/Media/WhatsApp Voice Notes/202115/PTT-20210405-WA0014.opus	Voice
/data/media/0/WhatsApp/Media/WhatsApp Images/IMG-20210405-WA0007.jpeg	Image
/data/user_de/0/com.android.providers.telephony/ databases/mmssms.db	SMS/MMS
/sdcard/WhatsApp/Databases/	Databases
/sdcard/WhatsApp/Media/	Media files
/storage/emulated/0/Android/data/com.whatsapp/cache/	Cache files
/storage/emulated/0/Android/data/com.whatsapp/ cache/cache_r.0/	Cache files
//storage/emulated/0/WhatsApp/Databases/	Databases
/storage/emulated/0/WhatsApp/Backups/	Backups
/storage/emulated/0/WhatsApp/Media/.Statuses/	-

Continues...

Table 31 – *Continues...*

Path to a folder or file	**Description**
/storage/emulated/0/WhatsApp/Media/WhatsApp Animated Gifs/Sent/	Sent GIFs
/storage/emulated/0/WhatsApp/Media/WhatsApp Audio/	Audio files
/storage/emulated/0/WhatsApp/Media/WhatsApp Audio/Sent/	Sent audio files
/storage/emulated/0/WhatsApp/Media/WhatsApp Documents/	Documents
/storage/emulated/0/WhatsApp/Media/WhatsApp Documents/Sent/	Sent documents
/storage/emulated/0/WhatsApp/Media/WhatsApp Images/	Images
/storage/emulated/0/WhatsApp/Media/WhatsApp Images/Sent/	Sent images
/storage/emulated/0/WhatsApp/Media/WhatsApp Video/	Videos
/storage/emulated/0/WhatsApp/Media/WhatsApp Video/Sent/	Sent videos
/storage/emulated/0/WhatsApp/Media/WhatsApp Voice Notes/202115/	Voice notes

Table 32. WhatsApp artifacts in iOS phone.

Path to a folder or file	**Description**
Apple_iPhone 6 (A1586).zip/Backup Service/ 7b0f88aab509380622546f45cd9c3955443c05fe/ Snapshot/Manifest.plist	Log entries
/private/var/mobile/Containers/Data/Application/ 1545879F-7A20-421E-A45B-1329C9B5D994	net.whatsapp.WhatsApp
/private/var/mobile/Containers/Shared/AppGroup/ group.net.whatsapp.WhatsApp.shared/ ChatStorage.sqlite	Table: ZWAMESSAGE. Message
/private/var/mobile/Containers/Shared/AppGroup/ group.net.whatsapp.WhatsApp.shared/ ChatStorage.sqlite	Table: ZWAMESSAGE, ZWAMEDIAITEM. Location

Continues...

Table 32 – *Continues...*

Path to a folder or file	**Description**
/private/var/mobile/Containers/Shared/AppGroup/ group.net.whatsapp.WhatsApp.shared/ ChatStorage.sqlite	Table: ZWAMESSAGE, ZWAGROUPMEMBER, ZWACHATSESSION. Sent or read message
/private/var/mobile/Containers/Shared/AppGroup/ group.net.whatsapp.WhatsApp.shared/ ChatStorage.sqlite	Table: ZWAMESSAGE, ZWAMEDIAITEM, ZWAGROUPMEMBER, ZWACHATSESSION. Messages with attachments
/private/var/mobile/Containers/Shared/AppGroup/ group.net.whatsapp.WhatsApp.shared/ ChatStorage.sqlite/	Vcards: Telia.vcf, SUPER info.vcf
/private/var/mobile/Containers/Shared/AppGroup/ group.net.whatsapp.WhatsApp.shared/ ContactsV2.sqlite	Table: ZWAADDRESS-BOOKCONTACT
/private/var/mobile/Containers/Shared/AppGroup/ group.net.whatsapp.WhatsApp.shared/Library/ Preferences/group.net.whatsapp.WhatsApp.shared. plist	Related to sent messages
/private/var/mobile/Library/CallHistoryDB/ CallHistory.storedata	Call log. Table: ZCALL-RECORD
/private/var/mobile/Library/CoreDuet/People/ interactionC.db	Application traces. Table: ZINTERACTIONS. Log entries
/private/var/mobile/Library/SMS/sms.db	SMS/MMS messages. Table: message, handle
/private/var/mobile/Containers/Shared/AppGroup/ group.net.whatsapp.WhatsApp.shared/Media/Profile/	Profile picture thumbnails
/private/var/mobile/Containers/Shared/AppGroup/ group.net.whatsapp.WhatsApp.shared/Message/ Media/0@status/	WhatsApp status images
/private/var/mobile/Containers/Shared/AppGroup/ group.net.whatsapp.WhatsApp.shared/Message/ Media/37254250893@s.whatsapp.net/	Shared media, picture. Location thumbnails
/private/var/wireless/Library/Databases/ DataUsage.sqlite	Application traces. Table: ZLIVEUSAGE, ZPROCESS

Appendix 12 - Wickr Me

Table 33. Details of Wickr Me application.

	Android	**iOS**
Package Name	com.mywickr.wickr2	com.mywickr.wickr
Version	5.76.5	5.76.8

Table 34. Wickr Me artifacts in Android phone.

Path to a folder or file	**Description**
/Android/data/com.mywickr.wickr2/cache/	Cache
/com.google.android.webview/com.mywickr.wickr2/ com.mywickr.wickr2.apk	App package for distribution
/com.google.android.webview/com.mywickr.wickr2/ description.info	Basic information about the app
/com.google.android.webview/com.mywickr.wickr2/ description.info.xml	Information about the app in XML format
/com.google.android.webview/com.mywickr.wickr2/ icon.png	The icon of the app
/com.google.android.webview/com.mywickr.wickr2/ live_external/cache/	Cache
/data/app/com.mywickr.wickr2-Av12E8vIav2IdPx1-tzgmA==/base.apk	Essential information about the app
/data/app/com.mywickr.wickr2-Av12E8vIav2IdPx1-tzgmA==/base.apk/AndroidManifest.xml	Essential information about the app
/data/data/com.android.vending/databases/localapp state.db	Table: appstate
/data/data/com.mywickr.wickr2/databases/wickr_db/ wickr_db.decrypted	Table: Wickr_Message, Wickr_User
/data/data/com.mywickr.wickr2/databases/wickr_db/ wickr_db.decrypted/a33376787a48afadb743b335dc59 acfe3a294959b021494e664f299ecb3e3534	-

Continues...

Table 34 – *Continues...*

Path to a folder or file	**Description**
/data/data/com.mywickr.wickr2/databases/wickr_db-wal/wickr_db.decrypted-wal	Write-ahead log
/data/data/com.mywickr.wickr2/files/enc/a69e8d9e-cfba-4c2c-abb7-503503f71ee0/image/gif.gif	GIF
/data/data/com.mywickr.wickr2/files/enc/5294a8d7-1cf8-460d-af69-a30f660ce342/image/gif.gif	GIF
/storage/emulated/0/Android/data/com.mywickr.wickr2/cache/	Cache

Table 35. Wickr Me artifacts in iOS phone.

Path to a folder or file	**Description**
Apple_iPhone 6 (A1586).zip/Backup Service/7b0f88aab509380622546f45cd9c3955443c05fe/Snapshot/Manifest.plist	Log entries
/private/var/mobile/Library/CoreDuet/People/interactionC.db	Application traces. Table: ZINTERACTIONS. Log entries
/private/var/wireless/Library/Databases/DataUsage.sqlite	Application traces. Table: ZLIVEUSAGE, ZPROCESS. Log entries