Digital Investigation 22 (2017) 88–106

Contents lists available at ScienceDirect

Digital Investigation

journal homepage: www.elsevier.com/locate/diin

Forensic analysis of Telegram Messenger for Windows Phone

J. Gregorio, A. Gardel*, B. Alarcos

Instituto Universitario de Ciencias Policiales, Universidad de Alcalá, 28871 Alcalá de Henares, Madrid, Spain

ARTICLE INFO

Article history:
Received 8 March 2017
Received in revised form
14 July 2017
Accepted 18 July 2017
Available online 18 August 2017

Keywords:
Forensic analysis methodology
Telegram Messenger
Instant messaging
Smartphones
Windows Phone

ABSTRACT

This article presents a forensic analysis methodology for obtaining the digital evidence generated by one of today's many instant messaging applications, namely "Telegram Messenger" for "Windows Phone", paying particular attention to the digital forensic artifacts produced. The paper provides an overview of this forensic analysis, while focusing particularly on how the information is structured and the user, chat and conversation data generated by the application are organised, with the goal of extracting related data from the information. The application has several other features (e.g. games, bots, stickers) besides those of an instant messaging application (e.g. messages, images, videos, files). It is therefore necessary to decode and interpret the information, which may relate to criminal offences, and establish the relation of different types of user, chat and conversation.

Introduction

The wide variety of forms of communication available today include voice calls, text messages, multimedia messages, emails, VoIP calls and instant messaging. Thanks to the development of fast data networks (e.g. WiFi, 3G, 4G) and the use of digital devices (e.g. smartphones, tablets, smartwatches), such communication is established immediately. In addition, specific functions have been developed to verify the information transmitted (e.g. text, images, videos, documents), facilitating user interaction. However, the speed at which these new technologies are changing and the high number of applications available whose primary function is instant messaging (IM), render it necessary to conduct detailed studies of this kind of application. Different platforms (mobile or desktop environments) host a wide range of applications whose main but not exclusive function is IM (e.g. Facebook Messenger, iMessage, Line, Signal, Snapchat, Tango, Telegram, QQ, Threema, Viber, WeChat, WhatsApp) (Husain et al., 2010).

Similarly, many other applications (e.g. POF, MeetMe, Wallapop) are not primarily intended for instant communication between users but nevertheless include this function. IM has become an essential means of communication used on countless occasions, far outstripping voice calls or text messages (SMS) (Lundgren, 2015; Woollaston, 2013). IM applications are no longer used solely for personal communication but are also increasingly employed in business and professional environments as a means of official communication, and as a vehicle for criminal acts such as threats, phishing, cyberbullying, grooming and terrorist propaganda (Ragan, 2015; Cuthbertson, 2015; Engel, 2015; Kharpal, 2015; Clare Foges, 2015).

This paper presents a new forensic analysis methodology is proposed and applied to the information generated by the mobile application "Telegram Messenger" for "Windows Phone" (WP), since no previous studies of the information stored by this IM application for this platform have been found in the literature.

Forensic analysts face various problems in relation to these applications, including constant upgrades and new features released with each new version (e.g. setting the frequency with which to delete messages on the recipient device, sending different file types, maximum size of files to send, voice calls via data or VoIP). There are several commercial forensic tools available that forensic analysts generally rely on to analyse the information generated by these applications; however, these tools do not always interpret all the information on the artifacts; they might produce false positives, or not cover the application or version in question). No single forensic tool covers all IM applications, or all of their features. Consequently, several of these tools are required in order to cover the full spectrum of mobile applications on the market. Unfortunately, many commercial tools base the range of applications they cover on the number of application downloads or even on client requests for the analysis of a specific application.

* Corresponding author.
E-mail addresses: jesus.gregorio@edu.uah.es (J. Gregorio), alfredo.gardel@uah.es (A. Gardel), bernardo.alarcos@uah.es (B. Alarcos).

http://dx.doi.org/10.1016/j.diin.2017.07.004
1742-2876/

Forensic analysts cannot afford to be limited by these constraints or to rely solely on the information processing capacity of these tools, since they may not identify all the applications installed or may only perform a rudimentary analysis of the information. Thus, none of the commercial forensic tools examined in the present study (Cellebrite 2016a,b; Oxygen Forensics 2016; Magnet Forensics 2016) offered satisfactory support for "Telegram Messenger" for WP, rendering it necessary to analyse and interpret the data stored by this application.

The rest of this article is organised as follows. The pertinent literature on forensic analysis and digital security is discussed in the "Related work" section. Then, the proposed methodology and the steps to conduct the analysis are described in the "Methodology for Forensic Analysis" section. The data structure used in Telegram Messenger Application for Windows Phone is described in next section "Data structure of Telegram Messenger for Windows Phone". The commercial and/or open source tools used and the results obtained are detailed in the "Forensic Analysis of data extracted from Telegram Messenger" section. Some use cases are shown in the "Forensic use cases" section and the paper ends with the "Conclusions".

Related work

The "Windows Phone" operating system does not currently have a large market share (International Data Corporation, 2016; Statista, 2016), being far surpassed by others. However, its mere existence, the constant development of new applications and their potential use to commit criminal acts all render it necessary to conduct technical studies of mobile applications, since they may at some point be subject to forensic analysis. Few technical forensic studies have been conducted on the applications available in Windows Phone Store. Similarly, the list of applications and updates compiled by some commercial forensic tools (Cellebrite 2016a,b; Magnet Forensics, 2014) is relatively brief. Although there are numerous studies on the forensic analysis of instant messaging applications, the present study differs not only in terms of the platform selected and its data management system, but also as regards how the information should be related. In contrast to the relational databases found on other systems, on "WP", "Telegram Messenger" stores information in different data files which contain fixed and variable data structures, rendering it difficult to relate the data.

As stated above, no technical studies have addressed this particular IM application for WP, although several papers are relevant to the forensic analysis methodology proposed here. Husain and Sridhar (2009) conducted a forensic analysis of three different instant messaging applications for the Apple iPhone (AIM, Yahoo! Messenger and Google Talk). Their study demonstrated the different IM artifacts that can be recovered, which include the password, conversation timestamp and conversation details. Some of this data were obtained in our study of the "Telegram Messenger" IM application.

In a more recent study, Aditya Mahajan et al. (2013) conducted a forensic analysis of data on two social messaging applications (WhatsApp and Viber), and determined the kind of data and information (e.g. chat, voice calls, sending and receiving images, audio) stored on the device's internal memory on Android smartphones. As shown in the section "Forensic analysis", the present study could prove helpful in relation to the structure of data stored in the application "Telegram Messenger" for "Android" devices.

One important issue for forensic analysis is to reconstruct the timeline of the extracted data. Anglano (2014) has shown how to reconstruct the chronology of contacts and the messages exchanged by users from the chat database, analysing deleted messages, when these were exchanged and which users exchanged them. The author provides a description of the information extracted from a study of the artifacts generated by WhatsApp Messenger, which is one of the methods included in the analysis methodology presented here. Besides conducting a forensic analysis of one of the applications with the largest market share, he also examined the artifacts, by generating different scenarios (or evidence of use).

Also of relevance is the paper by Satrya et al. (2016), comparing IM on Android smartphones, in which the authors describe a digital forensic analysis of three social IM applications on Android smartphones: Telegram ("Secret Chat"), Line ("Hidden Chat") and KakaoTalk ("Secret Chat"). They performed an offline and live logging forensic analysis of normal chat and private chat.

The forensic analysis methodology proposed here is based on a combination of different analyses (open knowledge analysis, analysis of artifacts and analysis of source code) of the different functionalities (e.g. users, normal chat, secret chat, conversations) to determine the type of data and information stored in the internal memory and how these are related on social messenger applications. The previous studies analyzed in this paper were based mainly on the analysis of artifacts.

Methodology for forensic analysis

The proposed methodology for extracting and processing information in a forensic analysis of an IM application consists of the three steps described below; these can be combined together to provide insight into the data and their interpretation.

1. **Open knowledge**: A study of the various open data sources available, including technical studies, books, related blogs and others, to obtain information on the application. Depending on the case, this information should be verified by the forensic analyst. Due to the constant release of new versions and features, it is not sufficient to understand the basics of an application; rather, it is necessary to know about each new feature that has been added, modified or removed, in order to possess a detailed knowledge of the tool's potential. Sometimes, the developer may provide information about application features, functions and data structure and organisation (schema, functions, data folder and files). This step should be repeated with each new version of the application under examination.
2. **Analysis of artifacts**: The study of open data sources should be followed by a study of artifacts or the traces these generate on digital devices (e.g. log records, data files). For this, the various commercial and open source forensic tools available should be deployed in the forensic laboratory to identify, acquire, preserve, document, analyse and present the data extracted from the digital evidence, without altering the elements studied and enabling the reproduction of this process at any time by another analyst (ISO/IEC 2012). This step should be repeated several times to obtain the various artifacts and digital evidence generated by the application.
3. **Source code**: The last step is to study the application programming, obtaining information from the source code itself. This method presents an added complication in that the analyst must know or even learn the programming language in which the application is written. A source code study yields a more technical and detailed level of knowledge about how the application works. When the source code it is not available, it may be necessary to carry out reverse engineering to discover the behavior of the application. This step should be repeated with each new version of the application.