Post-mortem digital forensic artifacts of TikTok Android App

Patricio Domingues*
ESTG - Polytechnic Institute of Leiria
Instituto de Telecomunicações
Computer Science and Communication Research Centre
Leiria, Portugal
patricio.domingues@ipleiria.pt

Ruben Nogueira*
ESTG - Polytechnic Institute of Leiria
Leiria, Portugal
2171569@my.ipleiria.pt

José Carlos Francisco*
ESTG - Polytechnic Institute of Leiria
Leiria, Portugal
2170996@my.ipleiria.pt

Miguel Frade*
ESTG - Polytechnic Institute of Leiria
Computer Science and Communication Research Centre
Leiria, Portugal
miguel.frade@ipleiria.pt

ABSTRACT

TikTok is a social network known mostly for the creation and sharing of short videos and for its popularity for those under 30 years old. Although it has only appeared as Android and iOS apps in 2017, it has gathered a large user base, being one of the most downloaded and used app. In this paper, we study the digital forensic artifacts of TikTok's app that can be recovered with a post mortem analysis of an Android phone, detailing the databases and XML with data that might be relevant for a digital forensic practitioner. We also provide the module tiktok.py to extract several forensic artifacts of TikTok in a digital forensic analysis of an Android phone. The module runs under Autopsy's Android Analyzer environment. Although TikTok offers a rich set of features, it is very internet-dependent, with a large amount of its inner data kept on the cloud, and thus not easily accessible in a post mortem analysis. Nonetheless, we were able to recover messages exchanged through the app communications channels, the list of TikTok users that have interacted with the TikTok account used at the smartphone, photos linked to the app and in some circumstances, TikTok's videos watched by the smartphone's user.

CCS CONCEPTS

• **Applied computing** → **Computer forensics**; **Evidence collection, storage and analysis**.

KEYWORDS

TikTok, Android apps, digital forensics

*All authors contributed equally to this research.

ARES 2020, August 25–28, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-8833-7/20/08…$15.00
https://doi.org/10.1145/3407023.3409203

ACM Reference Format:
Patricio Domingues, Ruben Nogueira, José Carlos Francisco, and Miguel Frade. 2020. Post-mortem digital forensic artifacts of TikTok Android App. In *The 15th International Conference on Availability, Reliability and Security (ARES 2020), August 25–28, 2020, Virtual Event, Ireland*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3407023.3409203

1 INTRODUCTION

Smartphones and their ubiquitous access to Internet have contributed to significant changes in our lives. The smartphone has become a many roles device, helping us to listen to music, watch videos, take pictures, record videos, receive/send email, guiding us to avoid traffic or in unfamiliar territories and in planning trips, just to name a few of the smartphone contributions. Nelson [18] reports that, on average, regular users touch their smartphone as much as 2 617 times a day. This indicates how much the smartphone has invaded and transformed our society. A side effect is that the smartphone accumulates a wealth of important data regarding its regular user. Thus, the growing trend of smartphones apprehended by law enforcement and then submitted for digital forensic analysis is of no surprise [8, 20], a tendency that is expected to continue [16, 17].

The emergence of the smartphone has also been instrumental to the growth of social networks. With its multimedia capabilities, such as multiple cameras, audio capture, GPS and ubiquitous access to Internet, a smartphone allows to easily share content to social networks, such as posting a photo on Facebook, liking a post on Instagram, or sharing one's opinion on Twitter, just to name a few activities that can be performed on social networks.

TikTok is a social network centered on the creation and sharing of videos, namely music-based. The videos are extremely short, with a duration ranging from 15 to 60 seconds, and should be, to achieve success, funny and engaging [19]. The origins of TikTok lies in the social media service `Musical.ly`, which allowed for the creation and sharing of short lip-sync videos, and then brought features such as filters and effects. In 2017, Musical.ly was acquired by the ByteDance Chinese company, and its main functionalities were integrated into TikTok in August 2018 [28]. Currently, the URL address `musical.ly` redirects to TikTok's website. Despite being mostly based on smartphones – to fully engage with TikTok, one has to install either the Android or the iOS app –, TikTok has

achieved tremendous popularity. Indeed, although it has only became available in 2017, TikTok smartphone apps are in the top 10 of the most downloaded applications of the 2010 – 2019 decade. Since its inception, TikTok for Android has more than 1 000 million installs as shown by Google Play[1]. Chapple [9] reports that TikTok recorded more than 315 million installs during the first quarter of 2020, more than any other app ever in a quarter. This is a testimonial of the tremendous popularity achieved by TikTok, particularly in the under-20 population, often referred as *generation Z* [26]. Some official entities such as World Health Organization (WHO), United Nations Higher Commissioner for Refugees (UNHCR) and newspapers such as the Washington Post have an active presence in TikTok. Likewise, many brands are also using TikTok as a promotion and marketing channel. Nonetheless, the vast majority of TikTok users are young individuals.

TikTok's app contains templates, filters and makes available visual effects to ease the creation of video content by users [12], allowing users to easily add audio from a vast music repository. One of the most popular feature of the platform is the possibility of creating a video that uses the audio of another video. As a social network, TikTok also promotes interaction among users of the platforms, with functionalities such as *likes*, allowing users to follow and to be followed by other users – when two users follow each other, they are designated as *friends* – as well as commenting on posted videos, and messaging each others. More recently, the app allows posting videos in comments. The app is heavily dependent on the internet, having very limited functionality when there is no internet connectivity.

For content classification and discovery, TikTok relies heavily on hashtags, with the platform recommending users to devote particular efforts to appropriately label their contents with appealing and meaningful hashtags. In fact, the interface for posting a video within the app has a dedicated space devoted to the insertion of hashtags. Hashtags are also of great importance for searching videos on the platform (e.g., #freezerfun, #tiktoktutorial), as there is no search facility to lookup specific videos. The app has also a passive mode, where videos are suggested to the user, based on the user's past interactions and reactions to watched videos.

The app can be used with or without login into a TikTok account. When no login is provided, most functionalities of the social network are not available, with the user being restricted to just watching content, without the possibility to comment, publish videos, exchange messages or expressing likes. In fact, exchanging messages requires not only to be logged on TikTok, but also to provide the phone number to the app.

TikTok provides a web interface[2], where non-registered users can watch videos suggested by the platforms, or the ones that are currently trending. The web interface also allows to access the posted content of public accounts. For this purpose, one just has to supply the username in the now traditional, at least for social networks, *@username* format. This corresponds to one of three distinct identifiers assigned by TikTok to an user account. Indeed, in TikTok, an account is identified through three distinct elements: *i*) an `userID`, sometimes referred as `uid`, which is a 19-digit numeric identifier set by TikTok at account creation; *ii*) a nickname which is set by the user, and *iii*) a so called `uniqueID`, which is also set by the user and corresponds to the `username`. The `uniqueID` is the identifier that can be used with the `@` symbol. For instance, to access the web interface of an account whose username is *example*, one just has to use the URL `https://www.tiktok.com/@example`. Figure 1 shows TikTok app displaying the public content of the `@terraqueoXXI` account, namely three videos. The same content can be viewed through a web browser with the URL https://www.tiktok.com/@terraqueoXXI. TikTok lets an user to change his/her `username`. Furthermore, there are sites on the internet that given a TikTok identifier will provide the other identifiers besides some metrics such as the number of posted videos, the number of received likes, etc. One example is https://commentpicker.com/tiktok-id.php.

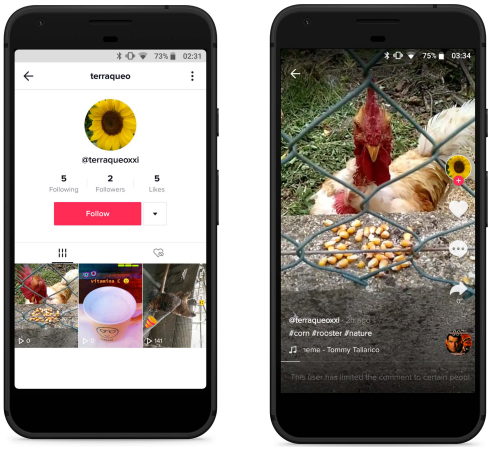

Figure 1: TikTok app displaying the public content of an account (left) and playing one of the video (right)

As TikTok is popular among young people, the needs to analyze someone's activities in TikTok is often related to situations involving young people. Examples include the disappearance of an individual, often minor, or his/her involvement in illegal activities. Other occurrences are malicious individuals that use the social network to establish contact and gain the trust of minors [11, 23], to spread fake news or extremist views [27] despite TikTok efforts to eliminate such content [24].

In this paper, we study the digital forensic artifacts that can be collected with a post-mortem analysis of an Android smartphone where TikTok's app has been used to interact with the platform. At the time of writing, and according to gs.statcounter.com, Android is the most widely used operating system (OS), with a global share of 39.13%, with 33.1% for Windows and 17.23% for Apple iOS.

We believe that the main contributions of this work are as follows: *i*) the identification and analysis of the digital artifacts of TikTok's Android app; *ii*) the creation of a TikTok module – `Tiktok.py` – for the Android Analyzer environment of the popular digital forensic Autopsy software. This way, we aim to enlarge the knowledge of the digital forensic community about TikTok's for Android, as well as providing open source software able to extract and explore digital forensic artifacts of TikTok.

[1] https://play.google.com/store/apps/details?id=com.ss.android.ugc.trill
[2] https://www.tiktok.com

The reminder of the paper is as follows. Section 2 reviews related work, while Section 3 details the main artifacts of TikTok, namely its SQLite databases and XML files. Section 4 presents our `TikTok.py` module than can be run under Autopsy Android Analyzer. Finally, Section 5 concludes the paper.

2 RELATED WORK

We review related work, first focusing on studies regarding digital forensics of Tiktok, and then on research related to forensic analysis of social networks applications, preferentially for Android.

The anonymous author *BTF_117* provides a detailed Open Source Investigation (OSINT) of TikTok through several blog posts, with the main goal of studying the possibility of gathering data about a specific TikTok user [7]. In his detailed research, the author sets a so called man-in-the-middle infrastructure to intercept the HTTPS network traffic exchanged between the client sides of TikTok – both the web interface and the phone app are studied – and TikTok servers. Captured traffic is mostly JSON-formatted data that provides a significant amount of information about the targeted user, including personal data and published video. In fact, the collected data are similar to the data available in the `aweme_user.xml` XML file as shown later in Section 3.4.1. Contrary to BTF_117's work, our study focuses on the content that exists in the phone. We believe that both approaches are complementary and can provide valuable data when combined.

In his digital forensic blog, Brignoni [6] analyzes the forensic artifacts of TikTok, focusing on message exchange and on some XML files. Our observations confirm that Brignoni's results regarding messages are still valid, indicating some stability of the app in the way it deals and stores messages.

The Oxygen Forensic Detective software provides support for collecting and parsing TikTok's data on Android physical dumps and on iOS [11], since version 12.0. Note that Oxygen Forensic Detective is a commercial product, while we provide an open source implementation based on Autopsy's Android Analyzer to extract relevant digital artifacts from an Android phone.

Other research regarding phone applications for social networks from a digital forensic perspective include: Al Mutawa et al. [1] who studied in 2012 the apps of the then dominant social networks – Facebook, Twitter and MySpace – in iOS, Android and Blackberry mobile devices; Knox et al. [14] who analyzed iOs and Android Hapn app; Shetty et al. [21] who evaluated the security of Android dating apps; Walnycky et al. [25] who assessed device-stored data and network traffic of 20 messaging apps for Android; and Alyahya & Kausar [2] who performed extraction and interpretation of digital forensics of the Android version of Snapchat app. Azfar et al. [5] have established a taxonomy for digital forensic of Android social applications, examining 30 Android apps such as Facebook, Linkein, Pinterest, Snapchat, Tumblr and Twitter, just to name the most popular ones. Works focusing on instant messaging applications include Anglano's [3] study of WhatsApp's app on Android, and Anglano et al. [4] who analyzed the Telegram messenger app.

Device	Android	TikTok
Aquaris BQ M5	7.1.2	16.0.41
ASUS Zenfone 3	8.0	15.1.0
Pixel 3 (image)	10.0	14.7.5

Table 1: Testing environment

3 DIGITAL FORENSIC ARTIFACTS

3.1 Methodology

The study of TikTok was performed as follows. TikTok was installed and used in two rooted smartphones, a BQ M5 and an ASUS Zenfone 3, with the former running Android OS 7.1.2 and the latter Android 8.0. Furthermore, an Android 6.0 emulator was also used to install and assess TikTok. Several versions of TikTok app were assessed, namely, 15.1.0 in the ASUS Zenfone 3, while the BQ M5 smartphone was kept update with the latest available version of the app, which was 16.0.41.

We also resorted to the Android 10 forensic image that is available on the Internet for forensic training [13]. This image comes from a Google Pixel 3 device, and has TikTok app, version 14.7.5, installed, with some interactions with TikTok network, namely posting a video, liking another one, exchanging a couple of messages and posting a comment on yet another video. Both physical devices were used to interact with TikTok social network, each with its own TikTok account. Whenever required, root-level extraction was performed through ADB interface, and extracted content analyzed to detect forensic artifacts left by TikTok app usage. For each SQLite3 database, we applied tools for recovering deleted data, so that we could gather transient data and thus better understand the purpose and use cases of each database. The Android 10 image was used to further test our Autopsy-based module for TikTok extraction.

The three digital environments used for testing are summarized in Table 1. We found no meaningful differences in the collected artifacts from the three TikTok versions, except for the video cache, where the most recent version of TikTok had an additional directory and encoding, as detailed in Section 3.3.2. Therefore, unless otherwise noted, reported artifacts are from the highest TikTok version, that is 16.0.41. Table 2 lists the permissions requested by this version of the TikTok Android app.

Next, we analyze TikTok's Android app forensics artifacts. We split our analysis in two: *i*) non-private data that can be accessed without root access and *ii*) private data that require root access, and henceforth a rooted phone. Table 3 lists the paths for both non-private and private data of TikTok.

3.2 Easily accessible data/artifacts

Accessible data are located in a hierarchy of directories that exists within the `/sdcard/Android/data/com.zhiliaoapp.musically` directory. We call these data *easily accessible* as root privileges are not required and thus they are, in a certain way, easily accessible. From a digital forensic point of view, the most relevant data are located in the subdirectories *i*) `picture` and *ii*) `prefs`, both inside

Permission	Observation
Contacts	Read contacts
Camera	Take pictures/videos
Location	Near approximate and GPS
Storage	Read and write storage
Phone	Read status and identity
Microphone	Record audio
Identity	Add or remove accounts
WiFi	WiFi information
Device/Apps History	List of running apps

Table 2: Permissions requested by TikTok Android app

Path	Comment
/sdcard/Android/data/com.zhiliaoapp.musically	Easily accessible
/data/data/com.zhiliaoapp.musically	Private app storage

Table 3: Main directories hosting TikTok's app data

```
├── app_librarian
│   └── 15.0.1.6181184523
├── app_textures
├── app_webview
│   └── Default
├── cache
│   ├── aweme_publish
│   ├── cache
│   ├── cachev2
│   ├── hashedimages
│   ├── instrument
│   └── WebView
├── code_cache
├── databases
├── no_backup
└── shared_prefs
```

Figure 2: TikTok's app private storage directory hierarchy

SQlite database filenames	
Cookies	lib_log_queue.db
userID_im.db	npth_log.db
androidx.work.workdb	OriginalSound
apm_monitor_t1.db	psdkmon.db
aweme.db	runtimeBehavior
db_im_xx	share.db
DeviceRegisterMonitor.db	showAd.db
downloader.db	splashsdk.db
feedback.db	ss_app_log.db
free_flow	ss_push_log.db
gecko_clean_statistic.db	storage_db
gecko_local_info.db	TIKTOK.db
google_app_measurement_local.db	video.db
i18n_live	Web Data

Table 4: List of SQLite database of TikTok for Android

the `cache` subdirectories, highlighting the fact that it is the disk cache of the app. As the name implies, the `picture` directories holds pictures. It reflects the image library Fresco organization, as TikTok App uses this Facebook's open source software library to deal with image loading, namely lazy loading, and caching [10, 22]. Specifically, the images – JPEG and WEBP – are kept in subdirectories named from 0 to 99. The images corresponds to content proposed to the user of the app, namely profile pictures and mostly posters from videos.

The directory `prefs` holds several JSON files. One file is called `local_prefs.json` and holds data regarding the HTTPS protocol used by TikTok, namely names of HTTPS servers, which are mostly from the `musical.ly` domains, a reference of the origins of TikTok. Interestingly, the `local_prefs.json` file has several references to the emerging QUIC network protocol [15], also known as HTTP3, and also refers to Google's web interface to its DNS service over HTTPS (https://dns.google.com:443).

3.3 Root only accessible data/artifacts

Forensically speaking, the richer data are kept in `/data/data/com.zhiliaoapp.musically`, that is the private App storage. Access to this directory requires root level and hence the phone has to be *rooted*. Figure 2 shows the directory hierarchy. The directory hierarchy follows Android's rules regarding directory naming. This is the case for the directories *databases*, *cache* and *shared_prefs*, whose names are self explanatory.

3.3.1 Databases. The studied version of TikTok Android app – 16.0.41 – has 28 SQLite3 databases, with all but two – `Cookies` and `WebData` – located under the `database` directory, while the `Cookies` and `WebData` databases are located in the subdirectory `app_webview/Default`.

The databases are listed in Table 4. Note that `userID` prefix in the database identified as `userID_im.db` represents TikTok's userID, which is, as stated earlier, a 19-digit integer number linked to the TikTok's account. By replacing `USERID` in `https://m.tiktok.com/h5/share/usr/USERID.html` with the 19-digit identifier, one can access the web interface of `userID`'s account, but only if the account is public [6].

We review the most meaningful databases and its tables from a digital forensic point of view. We omit from the discussion Android's mandatory table `android_metadata`, which is present in every database, and holds a single field, named `locale`, and a single record, always with the same content: the language string (e.g., *en_UK*). Regarding the `sqlite_sequence` table, it only exists in a database when there is at least one autoincrement field in the

database. This table is useful to assess whether tables with autoincrement fields have held any data, since an empty `sqlite_sequence` table means that no autoincrement value was ever requested and thus no data were inserted in these tables. Furthermore, unless noted otherwise, timestamps are in UNIX Epoch format, reporting the number of milliseconds since midnight Universal Time Coordinated (UTC), January 1^{st}, 1970.

Cookies. The `cookies` database has only one meaningful table, also named `Cookies`. It holds the session cookies of the app access through the web view. The date/time fields of the table – `creation_utc`, `expires_utc` and `last_access_utc` – are in a slightly modified Microsoft Filetime 64 format, since they represent the number of tenths of nanoseconds and not hundredths of nanoseconds as the original Microsoft format. The fields `creation_utc` and `last_access_utc` are interesting since their correspond to access by the app to TikTok's social network.

ss_app_log.db. This database logs events linked to the app. The `queue` table is used as a queue for sending reports about the usage and the crashes of the app. The table `event` keeps event, mostly internal events of the app, as well user interaction with the app. We believe this table is for analytic of the app usage and for reporting crashes, as one of the field of the `queue` table is `is_crash`. A logged event includes, as fields of the `event` table, a category, a tag, the userID, a timestamp and a JSON-formatted string named `ext_json` that carries the data of the event. An example of event is an `event_V3` category, with tag `device_info` and holding detailed data about the smartphone hardware – board, CPU, chipset, memory size, screen width and DPI, etc. – and the Android OS. The event table is rotated, with up to 500 of the oldest entries being removed when a given threshold is reached.

userID_im.db. This database holds data regarding the account identified by `userID`. Relevant data kept in this database are the messages exchanged between `userID` and other TikTok users. Messages are organized in conversations, where a conversation is a set of sent/received messages between `userID` and another TikTok user. The designation *message* encompasses not only text messages, but also other formats, such as images (animated or not), videos, hashtags and audio.

TikTok allows for the use of multiple accounts within the same app, although at a given moment, the app can only be logged in one account. This functionality is supported directly in the app, through the option `Addaccount` entry, available in the `Settingsandprivacy` menu. Internally, the app keeps one `userID_im.db` per account that have been logged in via the app, and does not delete the `userID_im.db` database when the user logs out of the `userID` account. This is a rationale behavior, as the user might want to log in again in the account in a near future.

Data regarding conversations are kept in three tables – `conversation_core`, `conversation_list` and `conversation_settings`, while messages are kept in the `msg` table. Specifically, `msg` table holds the messages exchanged between `userID` and other TikTok's users. The main fields of `msg` table are shown in Table 5. Each entry in the table represents an exchanged message. Data allow to retrace messages exchanged between two TikTok users, each of them identified by the text field `conversation_id` that has the following format:

Field	datatype	Description
msg_uuid	text	Unique message identifier
msg_server_id	bigint	ID of message server
conversation_id	text	0:1:{p1}:{p2}, p1=TikTokID1, p2=TikTokID2
type	integer	message type
deleted	integer	0=deleted msg, 1=non-deleted
created_time	datetime	message creation timestamp
status	integer	status of the message
sender	bigint	sender's tikTokID
content	text	message content in JSON format
read_status	integer	0=unread message, 1=read message

Table 5: Main fields of msg table from UserID database

Type	Description
5	animated GIF
7	text
8	video
15	animated GIF located in a remote server (e.g, giphy.com or tenor.com)
19	hashtag
22	audio
25	profile

Table 6: Interpretation of type field of table msg

`0:1:ID1:ID2`, where `ID1` and `ID2` are the TikTokID of the users engaged in the message, while the field `sender` keeps the TikTokID of the message's sender. The field `type` indicates the type of the message through an integer field. In our testing environments, we have identified the values for `type`. We list their meanings in Table 6.

The field `content` holds a JSON representation of the message, which depends on the type of message. Interestingly, a deleted message can still exist in the database with the `delete` field representing whether the message was deleted (=1) or not (=0). Similarly, the `read_status` field indicates whether the message has already been read (=1) or not (=0). The status of a message is kept in the integer field `status`, whose values have the following meaning: 2=*sent*, 3=*error while sending* and 5=*received*. The relationship between messages and the conversation they belong to is done through the `conversation_id` field, which acts as a primary key in any of conversation related tables, and as a foreign key in the `msg` table.

db_im_xx. The `db_im_xx` database keeps data about each users with whom `userID` has interacted with. It comprises solely of two tables, with only one, `SIMPLE_USER`, providing meaningful data. The `SIMPLE_USER` has as primary key the TikTok ID. For each listed user, the most relevant data are the `NICK_NAME`, the `AVATAR_THUMB` which holds JSON-formated content of the user's avatar including an URL to the thumbnail, `UNIQUE_ID` which represents the name handle and `FOLLOW_STATUS`. This last field stores the relationship

between `userID` and the listed user: =0 does not follow but can be followed by `userID`, =1 follows and =2 follows and is followed, that is a *friend* in TikTok's parlance.

lib_log_queue.db. This database has only one meaningful table: `queue`. It is associated with monitoring service and in our experiments, the table had only a single record that contained a JSON string holding data regarding monitoring services, kept in the field `value`. Within the JSON data, there was a detailed description of the smartphone hardware, of the OS (e.g., the ROM version), information regarding the data carrier, and the type of network access (e.g., "4G"). The JSON data also encompassed some TikTok's internal ID such as the already know userID, but also a `device_id` and an `install_id`, among other data.

video.db. The `video.db` database logs the HTTPS interaction between the app and TikTok's video repositories, as the name of the only table of the database expresses: `video_http_header_t`. Each row of the table keeps track of a video. The main fields of the table are `key`, which are unique values represented in 32-character hexadecimal (128 bits, e.g. A76D7A943A1185919B8DAD308CC918BB), that might be a MD5 checksum. The name kept in the `key` field is also used as the filename of the video if it exists in the video cache described in Section 3.3.2.

Other relevant fields are 1) `mime` which represents the MIME type of the video (e.g, "video/MP4"), 2) `contentLength` which holds the size in bytes of the video and 3)`extra` which is a JSON-formatted string, itself with three fields: *requestUrl*, *requestHeaders* and *responseHeaders*. These three fields correspond to traditional HTTP protocol elements, respectively, the *URL*, the *request header* and the *response header*. The content *response header* can provide some useful data, namely the UTC-based date/time of the HTTP server that has performed the response. Although the URL is not accessible outside of the app, as the URL requires proper authentication, the video might still be present in the video cache of the app, having as filename the value kept in the `key` field.

3.3.2 cache. The `cache` directory hosts two cache-named subdirectories: `cache` and `cachev2`. In coherence with the name, both directories are used for caching, more precisely, caching of videos. While `cache` holds videos in MP4 format and use as filename for each video, a 128-bit identifier that match the corresponding entry, if it exists, in the `video.db` database (Section 3.3.1) .

The `cachev2` directory stores videos in files whose names have `mdl` extension but that could not be decoded in well known video players such as Windows Media Player and VLC, with both players declaring the files as unrecognized format. In fact, these files had the first 128 bytes filled with the zero byte. Furthermore, for some of the `mdl` files, there was also another file, with the same base name, but with a `mdlnodeconf` extension. While some of these `mdlnodeconf` files were zero-sized, others had around 300 bytes. We found out that the non-zero `mdlnodeconf` held the bytes of the MP4 format header of the corresponding `mdl` file. Indeed, copying these bytes to the beginning of the `mdl` file, thus overwriting the zero-byte sequence, allow the now patched file to be played in the VLC media player. For other `mdl` files that either a zero-sized `mdlnodeconf` file or not `mdlnodeconf` file at all, we were able to make them also playable under VLC by overwriting the leading zero bytes of the file with a MP4 file header, for either an H264 or H265 video, based on the format of the `mdl` file. Figure 3 shows the 64 bytes used to *patch* an H265 video. We plan in future work to analyze the divergent behavior of file formats found in `cache` vs. `cachev2`.

XML file	Comment
appsflyer-data.xml	Flyer SDK
aweme_user.xml	Data about user
com.google.android.gms.measurement.prefs.xml	App timestamps
LoginSharePreferences.xml	Login preferences
search.xml	Performed searches

Table 7: Most relevant XML files

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000: 00 00 00 1c 66 74 79 70 69 73 6f 6d 00 00 02 00  ....ftypisom....
00000010: 69 73 6f 6d 69 73 6f 32 6d 70 34 31 00 00 6f fa  somiso2mp41..o.
00000020: 6d 6f 6f 76 00 00 00 6c 6d 76 68 64 00 00 00 00  moov...lmvhd....
00000030: 00 00 00 00 00 00 00 00 00 00 03 e8 00 00 81 27  ...............'
```

Figure 3: First 64 bytes of a valid TikTok's MP4 file

3.4 XML files

Besides SQLite databases, TikTok app also uses a large number of XML files. All XML files are located in the reserved area of the app and thus require root privileges to access. Of the 104 XML that existed in the reserved area, 100 were located on `/data/data/com.zhiliaoapp.musically/shared_prefs/` directory. We now review the XML files that contains data that might be relevant in a digital forensic examination, noting that all the reviewed files are located in the `shared_prefs` directory. Table 7 lists the most relevant XML files for a digital forensic examination.

3.4.1 aweme_user.xml. This file holds data regarding TikTok's user. To preserve space, only a shortened list of main data is given in Table 8. Other relevant fields include whether the TikTok account is linked to Facebook, Twitter, Weibo, Youtube or Instagram, and the number of *followers*, *friends* and of *followed accounts*. The `aweme_user.xml` file should definitely be consulted in a digital forensic examination.

3.4.2 search.xml. As the name suggests, the XML file `search.xml` keeps track of the recent searches performed by the app user. The history of recent searches are kept in the XML file under the entity `recent_history`. Note that the XML file solely holds the string searched: no results and no timestamps are kept.

3.4.3 appsflyer-data.xml. The `appsflyer-data.xml` is the XML file of the AppsFlyer Software Development Kit (SDK), which is an SDK to collect stats for advertisers to gauge the acquisition and retention rate of ad campaigns. For forensic purposes, the file keeps 1) the timestamp and also 2) the human representation of when the app was first installed, 3) from where it was installed and 4) who was the referrer for the installation. It also contains a code that represents the user within the AppsFlyer ad tracking software.

Name	Comment
nickname	(self explanatory)
uid	userID
uniqueID	@name
avatar_url	URL to avatar
register_time	timestamp of account creation
country_code	country phone prefix
bind_phone	last 4 digits of phone number (if provided)
email	email address (if provided)
birthday	(if provided)
location	(if provided)
gender	(if provided)
share_url	public URL of account
is_minor	user is minor or not
is_blocked	whether user is blocked by any other account
is_blocking	whether user is blocking by any other account

Table 8: Main fields of aweme_user.xml file

Finally, the XML file also contains two parameters that reference the phone IMEI – collectIMEI, collectIMEIForceByUser – , but at least in the scenarios that we analyzed, no IMEI was present in the XML file.

3.4.4 com.google.android.gms.measurement.prefs.xml. The com.google.android.gms.measurement.prefs.xml file is part of Google Global Mobile Service (GMS). GMS is, according to Google description, a collection of Google applications and APIs that help support functionality across devices. The XML file holds several timestamps such as when the app was installed (`app_install_time`) and when it was opened for the first time (`first_open_time`).

3.4.5 LoginSharePreferences.xml. When login into the TikTok network is enabled and performed by the user with the phone number, the `LoginSharePreferences.xml` file holds the full phone number associated to the account. It also holds TikTok's userID.

4 TIKTOK MODULE FOR AUTOPSY'S ANDROID ANALYZER

Autopsy is a well known open source software for digital forensics. It harbors within a functional graphical user interface, a plethora of functionalities needed by digital forensic practitioners. Some of these functionalities are provided by external tools that are called within Autopsy. Examples include the PhotoRec tool for file recovery and carving, Tika for detection of file format, SOLR as the content indexer and search engine and RegRipper – a set of PERL scripts – to analyze and extract data from Windows OS, just to name a few. Autopsy is extensible through modules that can be developed either in the Java language or in Jython, a Python version that runs within a java virtual machine. It should be noted that Autopsy has a dynamic release schedule, with at least two releases per year, besides the modules that are developed by external members of the project.

Autopsy also provides dedicated support for Android through its Android Analyzer environment, which allows for the development of modules to extract and/or analyze from Android forensic images. To develop a python module for Autopsy's Android Analyzer, one has to follow the structure of Android Analyzer, implementing the required classes and methods.

For the purpose of extracting TikTok specific data from Android OS images, we have developed `Tiktok.py`, a python module for Autopsy's Android Analyzer, available at https://github.com/labcif/T4AA. Currently, the module parses the `userID_im.db` and the `db_im_xx` databases, but we aim to provide further support for the other databases that contains meaningful forensic artifacts, as well for the XML files that hold important data for a digital forensic practitioner dealing with TikTok.

Figure 4, 5 and 6 display different views created by our Tiktok.py module within Autopsy in the examination of a testbed case. To preserve privacy, some of the data has been removed, and for due to space constraint, Figure 4 and 5 represent only partial view of the data showed by Autopsy. Figure 4 shows the contacts, while Figure 5 displays the messages exchanged within TikTok. Figure 6 uses Autopsy's *communication interface* to display the *communication graph*, that is, the graph showing with whom the TikTok account has exchanged messages, each node representing a TikTok account. The central node is the TikTok account being analyzed. while the thickness of each edge denotes the number of exchanged messages between the connected TikTok accounts.

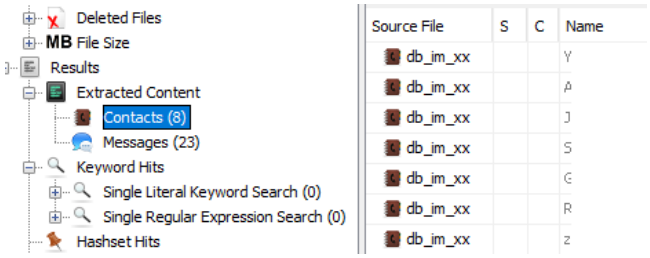

Figure 4: TikTok contacts - tiktok.py module

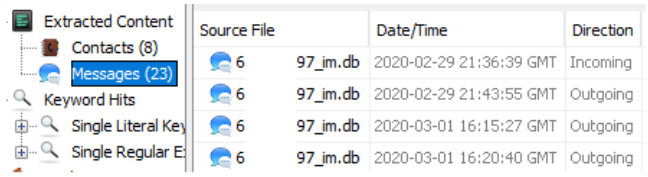

Figure 5: TikTok exchanged messages - tiktok.py module

5 CONCLUSION

TikTok is a social network based on the creation, sharing and appreciation of short videos that is appealing to a young user base. Both the Android and iOS versions of the app have achieved record levels of installation since its inception. In this paper, we presented the

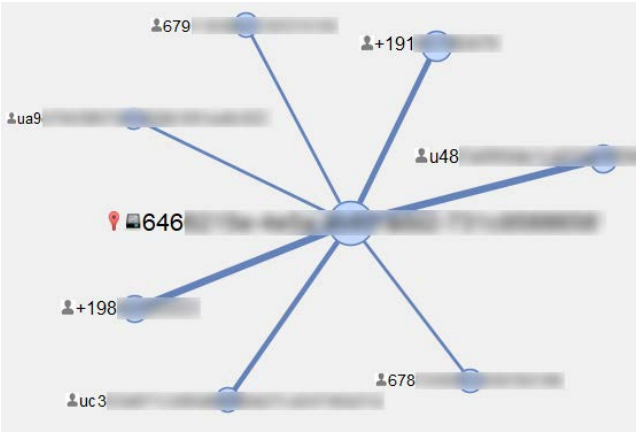

Figure 6: Exchange of messages among TikTok users - tiktok.py module

digital forensic artifacts that can be recovered from a post mortem analysis of an Android smartphone, and how to interpret the artifacts. For this purpose, we separated the artifacts in two classes: the ones accessible without the need of having root privileges within the smartphones and the artifacts only available within a rooted device. The later provides a wider wealth of data valuable for digital forensic practitioners, like the user accounts with whom the analyzed TikTok user have interacted, the exchanged messages within the member of the social network with the analyzed account and caches of TikTok's videos. TikTok is an app with many functionalities and a quite complex internal structure. For instance, version 16.0.41 of the app has 28 SQLite3 databases, several caches and as much as 104 XML files. All of these structures harbor a wide range of data, and for a better efficiency, guidance and proper tools are needed to quickly select and interpret the most interesting artifacts. This is the goal of this study and of the TikTok module for Autopsy.

In future work, we plan to add functionality to the Tiktok module for Autopsy's Android Analyzer. We aim also to study scenarios where a smartphone is configured with several TikTok accounts, to determine not only the left artifacts but also what can be and what cannot be attributed when multiple accounts access TikTok from the same device.

ACKNOWLEDGMENTS

This work was partially supported by CIIC under the FCT project UIDB-04524-2020, by FCT/MCTES and EU funds under the project UIDB/EEA/50008/2020. The authors would like to thank the anonymous reviewers for their insightful comments and suggestions.

REFERENCES

[1] Noora Al Mutawa, Ibrahim Baggili, and Andrew Marrington. 2012. Forensic analysis of social networking applications on mobile devices. *Digital Investigation* 9 (2012), S24–S33.
[2] Tadani Alyahya and Firdous Kausar. 2017. Snapchat analysis to discover digital forensic artifacts on Android smartphone. *Procedia Computer Science* 109 (2017), 1035–1040.
[3] Cosimo Anglano. 2014. Forensic analysis of WhatsApp Messenger on Android smartphones. *Digital Investigation* 11, 3 (2014), 201–213.
[4] Cosimo Anglano, Massimo Canonico, and Marco Guazzone. 2017. Forensic analysis of telegram messenger on Android smartphones. *Digital Investigation* 23 (2017), 31–49.
[5] Abdullah Azfar, Kim-Kwang Raymond Choo, and Lin Liu. 2017. Forensic taxonomy of Android social apps. *Journal of forensic sciences* 62, 2 (2017), 435–456.
[6] Alexis Brignoni. 2018. Finding TikTok messages in Android. Retrieved May 14, 2020 from https://abrignoni.blogspot.com/2018/11/finding-tiktok-messages-in-android.html
[7] Btf_117. 2020. TikTok OSINT: targeted user investigation. https://medium.com/@BTF117/tiktok-osint-targeted-user-investigation-9e206f8bb794
[8] P. Cedillo, J. Camacho, K. Campos, and A. Bermeo. 2019. A Forensics Activity Logger to Extract User Activity from Mobile Devices. In *2019 Sixth International Conference on eDemocracy eGovernment (ICEDEG)*. 286–290.
[9] Craig Chapple. 2020. TikTok Crosses 2 Billion Downloads After Best Quarter For Any App Ever. Retrieved July 3, 2020 from https://sensortower.com/blog/q1-2020-data-digest
[10] Facebook. 2020. Fresco - An Image Management Library. https://frescolib.org/
[11] Oygen Forensic. 2019. Oxygen Forensic Detective. Retrieved July 3, 2020 from https://blog.oxygen-forensic.com/whos-knocking-tiktok/
[12] John Herrman. 2019. How TikTok Is Rewriting the World. *The New York Times* 10 (2019). https://www.nytimes.com/2019/03/10/style/what-is-tik-tok.html
[13] Binary Hick. 2020. Android 10 Image Now Available! Retrieved July 3, 2020 from https://thebinaryhick.blog/2020/02/15/android-10-image-now-available/
[14] Shawn Knox, Steven Moghadam, Kenny Patrick, Anh Phan, and Kim-Kwang Raymond Choo. 2020. What's really 'Happning'? A forensic analysis of Android and iOS Happn dating apps. *Computers & Security* (2020), 101833.
[15] Adam Langley, Alistair Riddoch, Alyssa Wilk, Antonio Vicente, Charles Krasic, Dan Zhang, Fan Yang, Fedor Kouranov, Ian Swett, Janardhan Iyengar, et al. 2017. The quic transport protocol: Design and internet-scale deployment. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. 183–196.
[16] Laoise Luciano, Ibrahim Baggili, Mateusz Topor, Peter Casey, and Frank Breitinger. 2018. Digital Forensics in the Next Five Years. In *Proceedings of the 13th International Conference on Availability, Reliability and Security*. 1–14.
[17] R. Montasari and R. Hill. 2019. Next-Generation Digital Forensics: Challenges and Future Paradigms. In *2019 IEEE 12th International Conference on Global Security, Safety and Sustainability (ICGS3)*. 205–212.
[18] Patrick Nelson. 2016. We touch our phones 2,617 times a day, says study. *Network World* 7 (July 2016), 353–375.
[19] Zea Qiyang and Heekyoung Jung. 2019. Learning and Sharing Creative Skills with Short Videos: A Case Study of User Behavior in TikTok and Bilibili. In *International Association of Societies of Design Research (IASDR), Design Revolution*.
[20] Darren Quick and Kim-Kwang Raymond Choo. 2014. Impacts of increasing volume of digital forensic data: A survey and future research challenges. *Digital Investigation* 11, 4 (2014), 273–294. https://doi.org/10.1016/j.diin.2014.09.002
[21] Rushank Shetty, George Grispos, and Kim-Kwang Raymond Choo. 2017. Are you dating danger? an interdisciplinary approach to evaluating the (in) security of Android dating apps. *IEEE Transactions on Sustainable Computing* (2017).
[22] Yoo-jeong SONG, Soo-bin OU, and Jong-woo LEE. 2016. An Analysis of Existing Android Image Loading Libraries: Picasso, Glide, Fresco, AUIL and Volley. *DEStech Transactions on Engineering and Technology Research* imeia (2016).
[23] Mikael Thalen. 2020. Man pleads guilty to coercing children on TikTok into producing child porn. Retrieved July 3, 2020 from https://www.dailydot.com/irl/tiktok-child-porn-criminal-charges/
[24] TikTok. 2020. Community Guidelines. Retrieved July 3, 2020 from https://www.tiktok.com/community-guidelines
[25] Daniel Walnycky, Ibrahim Baggili, Andrew Marrington, Jason Moore, and Frank Breitinger. 2015. Network and device forensic analysis of Android social-messaging applications. *Digital Investigation* 14 (2015), S77–S84.
[26] Y. Wang, T. Gu, and S. Wang. 2019. Causes and Characteristics of Short Video Platform Internet Community Taking the TikTok Short Video Application as an Example. In *2019 IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW)*.
[27] Gabriel Weimann and Natalie Masri. 2020. Research Note: Spreading Hate on TikTok. *Studies in Conflict & Terrorism* 0, 0 (2020), 1–14. https://doi.org/10.1080/1057610X.2020.1780027
[28] SHI Xiaoye. 2019. *Analysis of ByteDance*. Ph.D. Dissertation. Swiss Federal Institute of Technology Zurich.