Digital forensic artifacts of the Your Phone application in Windows 10

Patricio Domingues[a,b,c], Miguel Frade[a,c], Luis Miguel Andrade[a], João Victor Silva[a]

[a] *School of Technology and Management - Polytechnic Institute of Leiria, Leiria, Portugal*
[b] *Instituto de Telecomunicações, Portugal*
[c] *Computer Science and Communication Research Centre, Portugal*

Abstract

Your Phone is a Microsoft system that comprises two applications: a smartphone app for Android 7+ smartphones and a desktop application for Windows 10/18.03+. It allows users to access their most recent smartphone-stored photos/screenshots and send/receive short message service (SMS) and multimedia messaging service (MMS) within their Your Phone-linked Windows 10 personal computers. In this paper, we analyze the digital forensic artifacts created at Windows 10 personal computers whose users have the Your Phone system installed and activated. Our results show that besides the most recent 25 photos/screenshots and the content of the last 30-day of sent/received SMS/MMS, the contact database of the linked smartphone(s) is available in a accessible SQLite3 database kept at the Windows 10 system. This way, when the linked smartphone cannot be forensically analyzed, data gathered through the Your Phone artifacts may constitute a valuable digital forensic asset. Furthermore, to explore and export the main data of the Your Phone database as well as recoverable deleted data, a set of python scripts – Your Phone Analyzer (YPA) – is presented. YPA is available wrapped within an Autopsy module to assist digital practitioners to extract the main artifacts from the Your Phone system.

Keywords: Digital forensic, Windows 10, smartphone, phone contacts, SMS, photos

1. Introduction

Since their inception in 2007, with the presentation of the first iPhone, smartphones have become part of our daily life, often being regarded as indispensable [1]. Smartphones have been replacing several technological devices of our digital life, such as digital cameras, global positioning system (GPS), music players, calendar/agenda, watches, to name just a few. One of the main reasons for the huge success of smartphones is their ability to provide ubiquitous access to the internet and, consequently, to a whole range of services. Consider, for instance, taking and sharing a photo: what took days several decades ago is now done in seconds, at a fraction of the cost. The usefulness of the smartphone goes well beyond sharing photos, being used to deal with email, to make and receive calls, to send and receive messages, to access social networks, to schedule main events in the calendar, or simply put, to maintain important elements of one's life. This makes smartphones valuable assets, carrying data that might be greatly useful in police investigations. In fact, seldom there is a digital investigation that does not require the forensic analysis of at least one smartphone [2]. However, besides the large variety of manufacturers and models [3], smartphones and their OS have been adopting encryption at the storage level for middle to high end models, with new encryption algorithms targeting low-end devices so that all can benefit from storage encryption [4]. This makes digital forensic analysis of these devices more difficult and time consuming, and sometimes just impossible, hardening the task of digital forensic examiners who are already overloaded with large volumes of data and devices [3].

Although more smartphones are sold per year than traditional computing devices such as laptop and desktop machines, personal computers (PC) are still widely used for professional and personal tasks. According to statcounter.com data, Microsoft Windows operating systems (henceforth, OS) are still the most used OS, with 37.43% of the global market share in December 2018 [5]. Mobile OS Android and iOS have a combined market share of roughly 50%, with 36.49% for Android and 13.16% for Apple iOS. While Microsoft's own attempt to enter the market of mobile OS was abandoned [6], the company has been publishing applications for Android and iOS, making available not only well known software such as Office and the Edge browser in these mobile platforms, but also other features and applications that interconnect Windows and non-Windows mobile devices such as Windows 10's Timeline service [7]. The Your Phone ecosystem, which comprises a pair of

applications – one for Android and another one for Windows 10 – is precisely an effort of Microsoft to build a cross platform and cross device ecosystem bridging Windows 10 with Android OS. In the current version, Windows' Your Phone application allows to perform two main operations directly from the desktop/laptop without touching the smartphone: *i*) access to the 25 most recent photos/screenshots of the smartphone and to *ii*) send/receive short message services (SMS) and multimedia messaging services (MMS). The photo/screenshot feature aims to eliminate the need for the user to send photos to her/himself through email to use them at the PC, for instance, for editing and/or inserting them in documents. The usefulness of dealing with SMS/MMS within the desktop/laptop is different: it frees the user from the need to interact with the smartphone to deal with SMS/MMS, and thus reduces distractions that break productivity [8]. In fact, Windows 10 Your Phone application is listed in the productivity group at Microsoft Store.

In the context of digital forensics, the Your Phone system presents a side-channel access to *i*) the 25 most recent photos/screenshots taken by the smartphone, as well as, *ii*) the SMS/MMS subsystem and, as we shall see later, *iii*) the address book of the smartphone. The SMS/MMS subsystem includes one-month worth of sent/received SMS/MMS and the associated metadata such as sending/receiving phone numbers and timestamps. The address book is the database holding the contacts of individuals/institutions, namely, phone number(s), name, and date/time of the last interaction the individual was contacted. This side-channel method can provide digital forensic examiners an opportunity to access some data of a smartphone, even when the mobile device is unavailable (e.g., missing, destroyed, etc.) or simply inaccessible due, for instance, to strong encryption and the access code is not available.

Despite the emergence of internet-based texting and voice/video instant messaging applications such as WhatsApp, Signal and Facebook Messenger, SMS texting is still a popular mean of communication, even if its usage has dropped significantly since its peak in 2012. Will Smale points out that around 22×10^9 SMS are sent daily [9]. The ubiquity of SMS means that they can be sent from any mobile network, and can be received by any phone device. This is not the case for internet-based instant messaging, where communicating peers must be on the same instant messaging network and to have internet connectivity. Conversely to SMS, MMS have failed to gain wide adoption [10], possibly due to costs and on the size limit imposed to the multimedia part: 300 KiB for MMS standard 1.2 and 600 KiB for standard

1.3, much less than internet-based communication applications, which on top of all have no direct costs.

The main contribution of this work regards the description and analysis of the artifacts present on a Windows 10 PC that has a Your Phone installation, which is or was in the past, connected with an Android smartphone. Specifically, the paper details *i*) the location, format and types of the SMS/MMS and of the most recent photos/screenshots artifacts and *ii*) how the user interactions between the PC(s) and the smartphone affect the artifacts. Another main contribution of the paper is the Your Phone Analyzer (YPA) software module that can extract and export the main SMS/MMS artifacts of the Your Phone system within the Autopsy forensic software. YPA is available under an open source license[1]. To the best of our knowledge, this is the first academic work that focuses on the *Your Phone* system from a digital forensic point of view and the first open source software solution that extracts artifacts from the Your Phone system.

In this paper, unless explicitly stated otherwise, the *SMS* term encompasses both SMS and MMS, while the designation *PC* (personal computer) holds for desktop, laptop and any other device running Windows 10 such as a netbook PC. Finally, the designation *recent photos* encompasses *recent photos and screenshots* of the smartphone.

The remainder of this paper is organized as follows. Section 2 summarizes related work, while section 3 presents the Your Phone system. Section 4 delves into the digital forensic artifacts that can be harvested from the Your Phone system in a Windows 10 PC. Section 5 presents the YPA software and its main functionalities. The limitations of Your Phone are detailed in Section 6, while Section 7 concludes the paper.

2. Related Work

Operating systems with rich interfaces and geared toward positive user experience generate numerous artifacts that can significantly help digital forensic investigations. This is the case for Windows OS, which hosts a wealth of artifacts. The artifacts are sub-products of features of the OS that either aim to benefit users, programmers or both. For instance, Windows Registry first appeared in Windows 95 to allow the OS and applications to store

[1]Give the URL

data, states and configurations. It has been since then a wealthy source of data for digital forensic practitioners [11, 12]. Some of the most interesting Windows registry entries for digital forensics are *User Assist*, *Most Recently Used* (MRU) and *Recent Apps*. *Windows Prefetch* is another Windows OS functionality that stems to enhance the user experience: it records the first 10 seconds of runs of every executed application with the goal to optimize the next launches of the application. In doing so, it creates a *prefetch* file for each application that holds important forensic artifacts such as the total count of executions and the date/time of the last eight executions [13]. Other examples of OS functionalities and services that yield valuable artifacts are the AmCache [14], *thumbcache* files [15], JumpLists [16, 17], Windows Search Indexer [18], Cortana digital assistant [19, 20], and the system resource usage monitor (SRUM) [21], to name just a few. Singh and Singh give a broad overview of the forensic artifacts created whenever Windows OS executes an application [22]. In [20], the same authors analyze the digital forensic artifact left by Microsoft's Cortana application in Windows. They highlight that when Cortana is also activated in the user's Android smartphone, events such as missed calls at the smartphone yield forensic artifacts at Windows machine [20]. Majeed et al. analyze the forensic artifacts left by the usage of the Facebook, Viber and Skype Windows 10's applications [23]. Interestingly, all three instant messaging applications rely heavily on SQLite3 databases, just as Your Phone does for SMS/MMS and the phone address book. Hintea et al. [24] provides a deep comparison between Windows 8.1 and Windows 10 regarding artifacts, although the analysis only covers the initial July 2015 release of Windows 10. Since then, several new features of Windows 10 have provided new forensic artifacts. This is the case for Windows Timeline [7] and also for the Your Phone system.

3. The Your Phone system

The Your Phone software ecosystem comprises two applications: *i*) Your Phone for Windows and *ii*) Your Phone Companion app for Android. The former is available for Windows 10/1803 or above, while the latter requires Android 7 or above. Besides the OS version requirements, Windows 10's Your Phone application also needs the user to have the PC signed in a Microsoft cloud account, such as `outlook.com` or `azure.com` account. The android Your Phone Companion application also needs to be signed in the same Microsoft account. The cloud account is mostly used for authentication and

not for storage, as we shall see later. Note that the Your Phone system allows for a single smartphone device to be linked to multiple Windows 10 PCs. This is the case, for instance, if the user has several Windows 10 devices, such as a desktop at home and a laptop computer for professional usage. Figure 1 represents the relationship between Your Phone Companion and Your Phone for Windows.

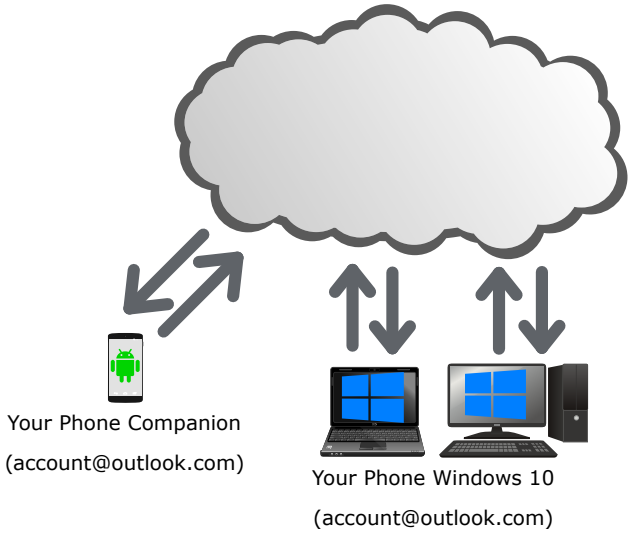


Figure 1: The Your Phone ecosystem

In this work, the experiments were performed with two Android 8.1 smartphones and three PCs: *i*) a laptop running Windows 10 Pro/1803; *ii*) a desktop computer with Windows 10 Pro/1809 and *iii*) another desktop computer running Windows 10 Enterprise/1803. Tests performed at the Windows 10 Enterprise machine resorted to a active directory domain-based account, while experiments at the other two machines relied on local accounts. The version of Windows 10's Your Phone application was 1.0.20453.0, while the Android Companion application was at version 3.4.1.

3.1. Your Phone Android Companion

The Android Companion application is available at the Google Play store[2]. It has, at the time of this writing, more than 5 million downloads.

[2]`https://play.google.com/store/apps/details?id=com.microsoft.appmanager`

The Companion application needs to be running at the smartphone and have internet connectivity so that the most recent photos can be replicated to the PC, and SMS sent/received at the PC application. For Your Phone to work, the smartphone needs to be connected to the internet through WiFi, otherwise synchronization with Windows 10 device(s) fails with an error message displayed in the Windows 10 application. Interestingly, paired Bluetooth connections between the Android device and the Windows 10 PC are not used, despite the Android application requesting the *synchronize with Bluetooth devices* permission during its installation. Although Microsoft's responses to comments existing at the application's Google Play Store section suggest that linking the smartphone with a PC requires that both are connected on the same WiFi network, this diverges from our experiments: we effortlessly linked the Your Phone Android Companion application with several PC that were using different networks to connect to the Internet. In fact, one of the PC was a desktop computer with no wireless card. However, the Your Phone Companion fails to synchronize and thus to function, if a virtual private network (VPN) is being used to route the smartphone internet traffic. The same connectivity failure arises when the Cloudflare's 1.1.1.1 DNS over HTTPS/TLS application is being run on the smartphone, a consequence of 1.1.1.1's functionality somehow relying on a VPN profile.

The companion application participates in the one-time linkage operation of the smartphone with a PC device, with the user being requested to confirm the setup by tapping the Companion application. The other user oriented functionality of the Companion application is the notification that signalizes that the Windows Your Phone has been launched at one of the linked Windows 10 PC. The user can block this connection at anytime by selecting "Terminate" in the companion application. Apart from these two functions, the companion application has no other direct interaction with the user. Contrarily to its limited direct interaction with the user, the Companion application plays an important background role, as it acts as a proxy between the smartphone and the PC application, synchronizing data with the linked PC(s). Rubino points out that data from Your Phone – SMS/MMS, the phone address book, and photos – are not kept in the cloud to comply with the European Union General Data Protection Regulation (GPDR)[8]. To verify that this was indeed the behavior of the system, we measured the volume of data traffic of the Your Phone companion application when linked with three PCs for synchronizing 25 photos totalling around 23.5 MiB. The synchronization operations with the PCs happened in different instants of

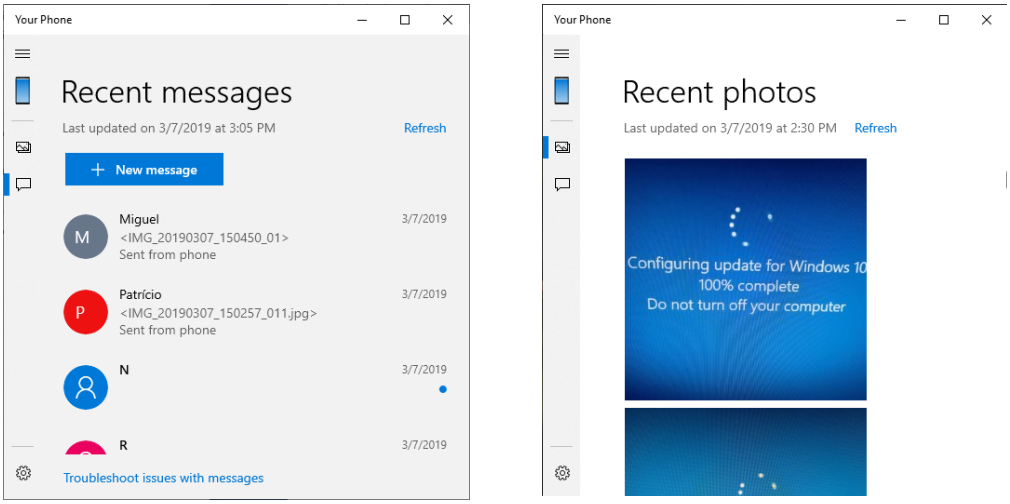
time, so that the synchronization for the 2^{nd} PC was only triggered after the 1^{st} PC has been fully synchronized, and so forth for the 3^{rd} PC. The goal was to observe whether the synchronization operations for the 2^{nd} and 3^{rd} PCs would reflect on data consumption of the Your Phone companion application, or, on the contrary, there would not be significant data increase on Your Phone companion application, an indication that the latter synchronization were done from data stored outside the smartphone. After all synchronizations had finished, data consumption of the Your Phone companion application was roughly 73 MiB, an indication that data for each of the three synchronization operations had came from the companion application. This potentially high data usage appears to be the main reason why Your Phone Companion only works over WiFi connections, refusing any other connection link, namely mobile data.

3.2. Your Phone Windows 10 application

The Your Phone application is an Universal Windows Platform (UWP) application available at Microsoft Store[3]. The executable file is `YourPhone.exe`, located in the following folder:
`C:\Program Files\WindowsApps\Microsoft.YourPhone_1.0.20453.0_X64_ _8wekyb3d8bbwe\`. We assume that system drive, given by the environment variable `%SystemDrive%`, corresponds to `C:`, as it is often the case.

The analyzed version of `YourPhone.exe` has two main screens: one for displaying large thumbnails of the most recent 25 photos and another one, called *recent messages* to interact with SMS and MMS. Both are shown in Figure 2. The messages screen lists SMS/MMS from the most recent to the oldest ones, grouping them in conversations, that is, all messages exchanged between the smartphone and a same remote peer are grouped in a single entry. Regarding MMS, the application displays the text content of the message, but not the media content attached to the MMS (e.g. a photo). Instead, the application shows the name of the multimedia file(s) attached to the MMS. The message screen also allows to send SMS, but not MMS. To send an SMS, the user either writes the destination phone number(s) or, more conveniently, selects it from the address book. As we shall see later, the address book available at the application is a synchronized replica of the smartphone's address book and constitutes one of the main forensic artifact.

[3] `https://www.microsoft.com/en-us/p/your-phone/9nmpj99vjbwv/`



(a) SMS/MMS (b) Photos

Figure 2: Screenshots of Your Phone Windows 10 application

Both screens – photos and messages – displays at the top the date/time of the last update and allows the user to trigger an update for the content of the screen. Finally, a third screen is used for basic settings: *i*) enabling/disabling photos of the smartphone to be displayed at the PC and *ii*) enabling/disabling the SMS synchronization. Besides the user-triggered content update, synchronization between the Your Phone application happens at the application's launch, and also when an SMS is sent or received. Another screen interaction is triggered when a SMS is received: a notification message is displayed by Windows' Notification Center, but only if Windows 10's *toast notifications* are enabled [24].

4. Artifacts of Your Phone in Windows 10

We analyze the main forensic artifacts yielded by the usage of the Your Phone ecosystem. We first describe the location of data in the local file system, then analyze the *most recent photos* feature and finally proceed to the SMS/MMS-based artifacts, that is, the SQLite3 database `phone.db`.

4.1. Location of data

In Windows 10, data related to the Your Phone system are kept under the `Microsoft.YourPhone_8wekyb3d8bbwe` folder, which is a subfolder

Table 1: Location of the main files of Windows 10's Your Phone

Name	Path
YourPhone.exe	%PROGRAMFILES%\WindowsApps\Microsoft.YourPhone_VERSION__8wekyb3d8bbwe\
#BaseDir#	%LocalAppData%\Local\Packages\Microsoft.YourPhone_8wekyb3d8bbwe\
photos/screenshots	#BaseDir#\LocalCache\Indexed\#GUID#\User\PhoneName\Recent Photos\
phone.db	#BaseDir#\LocalCache\Indexed\#GUID#\System\Database\

of %LocalAppData%\Packages\. The %LocalAppData% environment variable points to a path located within the user's home directory (e.g., c:\users\test\AppData\Local\), where test is the name of the account. The Microsoft.YourPhone_8wekyb3d8bbwe folder is used to keep the application data. It stores the most relevant artifacts. This per-user hierarchy of folders means that YourPhone's data and, consequently, the artifacts can be directly linked to an account and thus to a user. This is of great importance for digital forensics. Note that the data hierarchy is the same regardless of the type of account used: local or active directory-based. The hierarchy of folder and files holding Windows 10 Your Phone data is shown in Figure 3. To preserve space, only two of the 25 file photos are shown. From a forensically point of view, the most relevant files are the photos and the SQLite3 database phone.db. The location of the main files and artifacts of Your Phone is given in Table 1. Note that #BaseDir# is a convenience name that we use to designate the base folder where the data of Windows 10's Your Phone is kept. Likewise, the location of the executable is dependent on YourPhone's version. Thus the identifier VERSION used in Table 1 needs to be replaced by Your Phone version, 1.0.20453.0_x64 in our study, where x64 is for a 64-bit application and x86 for the 32-bit version. Finally, #GUID# refers to the global unique identifier used by the local installation of Your Phone (e.g., 2DDCAB42-C88A-518F-BADE-E8F160699121).

4.2. Most recent photos/screenshots

Up to 25 most recent photos/screenshots of the smartphone are kept in the subfolder \LocalCache\Indexed\#GUID#\User\PhoneName\Recent Photos\.

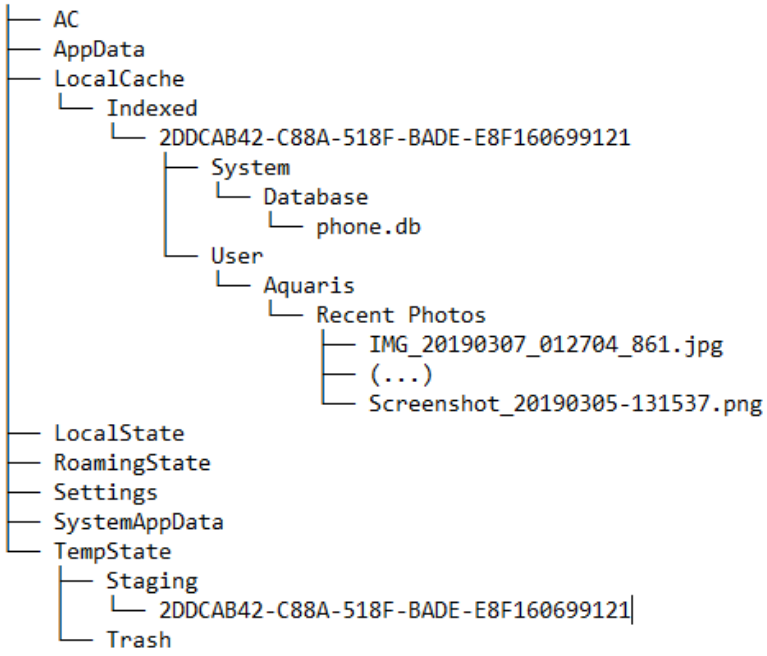


Figure 3: Hierarchy of directories and files holding the Your Phone Windows 10 PC's data

The `#GUID#`[4] is a global unique identifier whose value is dependent on the machine. Indeed, installation of Your Phone across several machines all yielded different GUID. Furthermore *PhoneName* represents the name assigned by the user to her/his own smartphone. By default, and if the user has not customized the name, its corresponds to the brand and model of the smartphone (e.g, `Samsung 7`). Note that the name of the `Recent Photos` folder is localized, and thus solely valid for English-version of Windows. For example, for the Portuguese version of Windows 10, the name of the folder is *Fotografias Recentes*.

The files holding the photos are a bit-by-bit copy of the smartphone's files. They have the same names and, as importantly for forensic purposes, they have the same metadata. The content of the folder where the photos are stored is loosely controlled by the Your Phone application. Indeed, even it is possible for the user to manually add or delete files, for instance through

[4] `https://msdn.microsoft.com/en-us/library/windows/desktop/aa373931(v=vs.85).aspx`

Windows Explorer, Your Phone restores the content of the folder as soon as a refresh operation happens. However, the replacement of files is not detected by the application as long as filenames do not change. This means that Your Phone uses filenames and not hash-based identifiers to synchronize the copy of the most recent photos kept at the PC with the linked smartphone. This might undermine the credibility of the content of the most recent photos folder. However, swapping a photo for another one requires to get not only the right filename but also the correct metadata. Moreover, several techniques exist for camera fingerprinting to assess whether a photo was taken with a given camera/smartphone [25, 26].

When the set of the 25 most recent photos/screenshots is changed at the smartphone, modifications are reflected by the Your Phone application. These changes include not only taking new photos/screenshots with the smartphone, but also deleting ones that include the most 25 recent ones. Therefore, when a change is detected, the Your Phone application updates its local set of the most recent photos. The update process is performed in two stages: first, Your Phone application downloads thumbnails for all the *new* photos, and then, in a second stage, it downloads the full sized files. Each thumbnail file has the name of the corresponding photo file prefixed with the *thumb.* string (e.g., `thumb.IMG_20190209_164026_319.jpg`). The thumbnails are displayed in the photo screen of the Your Phone application (Figure 2) providing visual feedback to the user during the download of the actual larger full-sized photo files. As soon as the download of full-sized photos is completed, the thumbnails are moved to the `\TempState\Trash\`. Periodically, Your Phone deletes all the files of the `Trash` folder.

4.3. The Phone.db database

Data regarding the SMS, MMS and address book are kept in a SQLite3 relational database whose supporting file is *phone.db*. This database file is kept in the subfolder `\LocalCache\Indexed\#GUID#\System\Database\` of `#BaseDir#` (Table 1). The usage of a SQLite3 database departs from many other Microsoft products such as SRUM [21], Cortana [20] and Windows Mail [27] that rely on Microsoft's Extensible Storage Engine (ESE) databases to handle data storage and manipulation. Despite being uncommon in Microsoft desktop products, SQLite3 databases are widely used in applications for mobile and embedded devices, namely in the Android and iOS platforms [28]. This is mostly due to SQLite3 low resources demand, cross

platform availability and support for SQL. In the context of the Your Application ecosystem, which targets both Windows 10 and mobile environments, it makes sense to use SQLite3 databases. Indeed, the volume and update frequency of data is low as it depends on human interaction, which is mostly sending and receiving text messages, and rather infrequently, adding and/or updating address contacts. Note that the desktop Windows Timeline application, which is another Microsoft product for cross-device experience, also resorts to SQLite3 databases [7]. From the forensic point of view, SQLite3 databases are much easier to handle than ESE ones, since the SQLite3 format is simpler, well documented and supported by a plethora of auxiliary tools. Moreover, in the particular case of the `phone.db` database, the forensic analysis is further simplified by the fact that names of tables and names of fields are quite explicit. Although this might seem the norm for databases, some Windows ESE databases such as Windows Mail identify record fields with hexadecimal tags such as 0x00 37 00 1f [27].

4.3.1. Structure of the phone.db database

The phone.db database comprises 10 flat tables. The name of the tables, as well as a brief description of each table is given in Table 2. Next, we focus on the tables that store the most interesting data for digital forensics.

4.3.2. Forensically meaningful data of phone.db

Contact and Address. These two tables hold the address book. Table 3 shows the fields of table **Contact**, while Table 4 displays the fields of the table **Address**. Although, the type *filetime* does not exist in SQLite3 databases, in this paper, we use this designation to identify *integer* fields that actually store a 64-bit FILETIME, that is, the number of 100 nanoseconds elapsed since January 1^{st}, 1601 [29].

The **Contact** table identifies the name of contacts through the fields *display_name* and *alternative_name*, with one record per contact. Both fields hold the same content, although on a slightly different order (e.g, "John Doe" and "Doe, John"). Another field is *last_updated_time*, a 64-bit FILETIME date/time. Despite the name of the field that could indicate that it stores the timestamp of the last update operation of the contact, our analysis did not confirm this behavior. Indeed, although the field reflected the correct timestamp after the creation/update of a contact, the value also changed whenever a SMS/MMS or phone call was exchanged with the contact. More strangely, the value would also change for several contacts without a recog-

Table 2: Tables of the phone.db SQLite3 database

Table name	Description
`Address`	Phone numbers of contacts (address book)
`Contact`	Name of contacts (address book)
`Conversation`	SMS/MMS between two (or more) peers are grouped in a conversation and identified by a unique *thread_id*
`Message`	Text content of all exchanged message (SMS/MMS)
`Message_to_address`	Identifies destination phone number for each sent message (SMS/MMS)
`Mms`	Date/time of sent/received MMS
`Mms_address`	Sender/destination of sent/received MMS
`Mms_part`	multimedia parts of MMS (only filenames and format of multimedia content, not the actual content)
`Sending_message`	Keep records of each failed send attempt
`Sync`	Keep temporary data while synchronizing

nizable pattern. Therefore, we deem this field *unreliable* for forensic usage. The (non-declared) primary key is the numeric field *contact_id*. Finally, two more fields exist: *nicknames* and *last_contacted_time*. Despite their explicit names, in our test cases these fields always had the NULL value.

The **Address** table holds the phone number(s) of each contact. For that purpose, each row identifies the contact with the field *contact_id*, which links with the same name field of the **Contact** table. When a contact has multiple phone numbers, then the **Address** table has, for the contact, one entry per distinct phone number, with all entries of the contact identified with the same *contact_id*. An integer field – *address_type* – classifies the phone number – home, mobile, job, and so on. For example, a phone number labeled as *home* has *address_type*=1, while *address_type*=2 is *mobile phone*. Overall, the value for *address_type* ranges from 1 to 6, although we could not decipher the contact field for type *address_type*=4 as no such value appeared in our tests. Table 5 maps each *address_type* numerical value to its corresponding designation. Note, however, that the accuracy of the *address_type* (home, mobile, work,...) field depends on how the contact's entry was filled. Often, when registering a phone number in the address book, a user does not properly fill the fields, as all that might interest is simply the name of the contact and

Table 3: Fields of table `contact`

Column	Type	Description
`contact_id`	integer	Unique identifier
`display_name`	text	Name of the contact
`alternative_name`	text	Same as `display_name` but in different order
`nicknames`	text	(Always `NULL`)
`last_contacted_time`	filetime	(Always `NULL`)
`last_updated_time`	filetime	Last date/time contact was updated

the phone number(s). Another forensically interesting field of the **Address** table is *last_contacted_time*. This field holds the timestamp, again a 64-bit FILETIME, of the last text contact between the smartphone and any of the phone numbers that are linked to the contact. In our experiments, we observed that this field was always updated when a phone call or a SMS/MMS was exchanged between the smartphone and the contact. We also found out that several other situations triggered the update of the timestamp kept by the *last_contacted_time* field. The situations are *i*) Usage of the WhatsApp platform with an account linked to the phone number of the smartphone and *ii*) Usage of the Signal platform[5]. Situation *i*) – usage of WhatsApp – updates the timestamp field *last_contacted_time*. The same happens with the Signal platform (situation *ii*)), but only if the Signal application is configured as the SMS messaging application of the smartphone. If the Signal application is not acting as the SMS messaging application in Android, then usage of Signal does not trigger the refresh of the *last_contacted_time* field. We hypothesize that the usage of any application that can modify the content of Android's SMS/MMS database, such as WhatsApp and Signal, can trigger the update of the field. Note that none of the above regarding the *last_contacted_time* field applies to contacts that are kept directly into the SIM card. Indeed, for these contacts, the *last_contacted_time* field holds the date/time of the smartphone first boot with the SIM card.

The **Address** table has another field related to contacts: *times_contacted*. This field counts the number of contacts between the smartphone and the other phone number. Similarly to *last_contacted_time*, this field is also af-

[5] `https://www.signal.org/`

Table 4: Fields of table `address`

Column	Type	Description
`contact_id`	integer	Unique identifier of the address
`address`	text	Phone number
`address_type`	integer	1=home phone, ...
`is_primary`	integer	(Always zero)
`times_contacted`	integer	Number of contacted times
`last_contacted_time`	filetime	Date/time of last contact (unreliable)

Table 5: Meaning of `address_type` numerical values

address_type	Description
1	Home phone number
2	Mobile phone number
3	Office phone number
4	Unknown
5	Main phone number
6	Other phone number

Table 6: Fields of table `conversation`

Column	Type	Description
`thread_id`	integer	Unique identifier of the conversation
`recipient_list`	text	Peer(s) who is(are) in this conversation
`timestamp`	filetime	Date/time of last exchanged SMS
`msg_count`	integer	Number of SMS in the conversation
`unread_count`	integer	Number of unread SMS in the conversation
`summary`	text	Text of the most recent SMS of the conversation

Table 7: Fields of Table `message`

Column	Type	Description
`message_id`	integer	Unique identifier of the SMS
`from_address`	text	Phone number of sender
`thread_id`	integer	Identifier of conversation
`status`	integer	1=unread; 2=read
`type`	integer	1=received; 2=sent
`subject`	text	Unknown (always NULL)
`body`	text	Text content of the SMS
`timestamp`	filetime	Date/time of SMS
`pc_status`	integer	Unknown (always 1)

fected by the use of WhatsApp text messaging, with each message counting as one contact. More importantly, the field value increments normally until it reaches 10. Then, the value of the field only increments in chunks of 10 (10, 20, 30 and so on).

Conversation. In `phone.db`, a *conversation* refers to the set of SMS/MMS exchanged between the smartphone's user and the same individual or individuals, if SMS/MMS have multiple destinations. Note that a given conversation can simultaneously hold SMS and MMS, since the only aggregator is the destination list. The fields of the **Conversation** table are shown in Table 6. A conversation is uniquely identified by an integer *thread_id*, which acts as the primary key. As we shall see later, this field is further used in the **Message** (SMS) and **MMS** tables to identify the conversation to which an SMS/MMS belongs to. The *recipient_list* of the **Conversation** table holds

Table 8: Fields of Table `message_to_address`

Column	Type	Description
`message_id`	integer	Unique identifier of the SMS
`address`	text	Destination phone number

the peer's phone number or a text identifier. A text identifier identifies special, mostly business accounts that send, but do not receive SMS. Examples include the SMS sent by companies for setup and verification purposes, such as two-factor authentication (e.g., Internet-based services such as Google and WhatsApp, online banking, etc.) or for informative purposes, such as promotions or invoices to be paid. The other fields of the **conversation** table are self explanatory.

Message and Message_to_address. These two tables handle data exclusively related to SMS. The **Message** table, shown in Table 7, keeps one record per exchanged SMS. Each record is linked to the containing conversation through the *thread_id*. The *from_address* field holds the sender phone number. This field is empty for SMS sent through the phone number associated to Your Phone. *Status* is 1 for unread SMS and 2 for read SMS, while the value of the *type* field distinguishes between received SMS (*type*=1) and sent SMS (*type*=2). The *body* field holds the text content of the SMS, while the 64-bit FILETIME *timestamp* field keeps the date/time of the SMS. In our analysis, we could not ascertain the purpose of the fields *subject* and *pc_status*. All of them had non-meaningful values or were empty.

A separate database table to keep the destination address(es) of an SMS is needed to accommodate the cases when an SMS is sent to multiple destinations. This table is **Message_to_address** (Table 8), with the SMS identified by the *message_id* field which links it to the **Message** table.

MMS, MMS_part and MMS_address. These three tables deal with MMS. The **MMS** table holds the metadata of every sent/received MMS, as shown in Table 9. The *message_id*, which is a integer identifier of the MMS, acts as the primary key. As reported earlier, the *thread_id* field links the MMS to its corresponding conversation, and thus to the database record in the **Conversation** table. The integer fields *status* and *type*, keep, respectively, the read/unread status (unread=1; read=2) and whether the MMS was received or sent (received=1; sent=2). The *timestamp* fields holds the date/time of the MMS. We could not decode the meaning of the fields *sub-*

ject, *charset* and *pc_status*, since they remained constant across all our experiments, as shown in Table 9.

The **MMS_part** table (Table 10) holds, for each MMS, n records, with n corresponding to the number of parts that comprises the MMS, plus an additional part that keeps data using the Synchronized Multimedia Integration Language (SMIL) format. SMIL [30] is a XML-based language to describe how a given set of multimedia objects should be displayed and integrated with the environment. For MMS, it allows to control how the smartphone will notify the user and display the MMS. As such, it appears to have no real forensic value. In the **MMS_part** table, the SMIL record for an MMS has the *sequence_num* set to -1, while the records for the other parts have this field sets to 0. The text *content_id* field holds the filename of the multimedia resource (e.g. `IMG_20190210_180722_1031.jpg` for a photo). An MMS part is an element of the MMS, like for instance a JPG file. For example, an MMS with text and two photos is represented with four records in the **MMS_part** table: one record for the text, another two to hold the name of each photos, plus an additional record for the metadata of the MMS. For each record, the field *part_id* is an integer sequence that acts as the primary key of the table. The *message_id* field links the record to its corresponding MMS message from the **MMS** database table.

The field *content_type* holds the MIME type of the content kept by the record (e.g., `image/jpeg`), while the *text* field has the text content, but only for the part that represents the text (if any) of the MMS. For the other part(s) of the MMS, this field is empty. Likewise, the *charset* is only defined for record with a filled *text* field, while in our experiments, the *blob* field was always NULL.

Table **MMS_address** (Table 11) records the sender/receiver address(es) in the *address* field. It is similar, although more complete, than the SMS-related **Message_to_address**. The field *message_id* links the record to the corresponding entry in the **MMS** table, while the field *type* points out whether it is a received (type=0) or sent (type=1) MMS.

4.4. Properties of the phone.db database

To preserve space, Table 12 only lists the properties of the `phone.db` database that can impact a digital forensics examination, namely the recoverability of deleted records. A SQLite database is organized in pages. As shown in Table 12, `phone.db` uses 4 096-byte pages.

Table 9: Fields of Table `mms`

Column	Type	Description
`message_id`	integer	Unique identifier of the MMS
`thread_id`	integer	Identifies the conversation
`status`	integer	unread=`1`; read=`2`
`type`	integer	received=`1`; sent=`2`
`subscription_id`	integer	Unknown (always `1`)
`subject`	text	Unknown (always empty)
`charset`	integer	Unknown (always `0`)
`timestamp`	integer	Date/time of MMS
`pc_status`	integer	Unknown (always `1`)

Table 10: Fields of Table `mms_part`

Column	Type	Description
`part_id`	integer	Uniquely identifies the record
`message_id`	integer	Identifies the linked MMS
`sequence_num`	integer	`-1` or `0`
`content_id`	text	Content of the MMS
`content_type`	text	MIME type of this part
`text`	text	Content of the part
`name`	text	Name of the part
`charset`	integer	Charset of the part
`blob`	blob	Unknown (always empty)

Table 11: Fields of Table `mms_address`

Column	Type	Description
`message_id`	integer	Identifies the MMS
`contact_id`	integer	Unknown (always 0)
`address`	text	Destination address
`type`	integer	received=`0`; sent=`1`
`charset`	text	Charset of the MMS

Two important properties control how an SQLite3 database handles delete operations: *i*) *secure delete* and *ii*) *auto-vacuum* [31]. The *secure delete* property controls how a record is deleted: if enabled, the whole content of the record to be deleted is overwritten with nullbytes, while a delete operation with *secure delete* deactivated simply marks the record as free space. The Auto-vacuum property controls how SQLite handles a delete operation that leaves one or more database pages empty, that is, with no active records. Specifically, with auto-vacuum enabled, SQLite automatically removes empty pages, effectively compacting the database file, and thus making impossible the recovery, at the database level, of the deleted records that were hosted inside the eliminated pages. Conversely, when auto-vacuum is off, there is no automatic compacting operation of the database, although compacting the database can still be manually ordered with the *vacuum* command. The `phone.db` database is neither set for secure delete, nor for auto-vacuum. This is a plus for digital forensics, since it increases the possibility of recovering deleted records, as shown in Section 5.1. In Your Phone, recovering records might allow to access deleted SMS/MMS and removed contacts of the address book.

In `phone.db`, a delete operation might either be due to user action – a SMS/MMS or a contact(s) is deleted at the smartphone – or the result of the Your Phone application that discards SMS/MMS kept at `phone.db` that are older than 30 days. Note that recovery of SQLite3 deleted text is a non-trivial task due to SQLite3 keeping text fields in variable length cells [32]. Nonetheless, as most of the content of `phone.db` is kept in UTF-8 text, namely *i*) the SMS/MMS, *ii*) the names and *iii*) the phone numbers, even the recovery of non-structured content can provide useful data. The fact that UTF-8 encoding relies on distinctive binary coding patterns also increases the probability of recovering meaningful content.

4.5. Other Artifacts

The execution of Your Phone application in a PC leaves the usual Windows artifacts such as Prefetch, SRUM, Jump Lists, ShimCache, Timeline, to name just a few [22]. Additionally, thumbnails of the photos/screenshots may exist in Windows's `thumbcache.db` [15]. Besides these regular Windows artifacts, the existence and execution of Your Phone also leaves traces in Windows Registry and in Windows Event Log. Next, we describe these traces.

Table 12: SQLite3 properties of `phone.db`

Property	Value
User version	`8`
Page size	`4 096` bytes
Encoding	`UTF-8`
Secure delete	`OFF`
Auto-vacuum	`OFF`

Windows Registry. Most of the footprint of Your Phone application in Windows registry comprises the usual keys of UWP applications. From the forensic point of view, there are three interesting items in Windows registry: *i*) the integer entry `WasEverActivated` from the `HKEY_Class_Root` hive, *ii*) the set of entries under `microsoft.yourphone_8wekyb3d8bbwe-0` in the HKEY_USERS hive and *iii*) the `InstallTime` entry.

The entry `WasEverActivated` is located at `HKU\SID\Software\Classes\Local Settings\Software\Microsoft\Windows\CurrentVersion\AppModel\SystemAppData\Microsoft.YourPhone_8wekyb3d8bbwe`, with `HKU` standing for `HKEY_Users`, and SID for the user's *Security IDentifier* [33]. The entry has a value of 1 to signal that Your Phone application has been activated, but not necessarily that it still maintain this status, since this entry persists after the application has been uninstalled.

The key `microsoft.yourphone_8wekyb3d8bbwe-0` is located under the registry path: `HKU\SID\Software\Microsoft\Windows\CurrentVersion\SettingSync\Namespace\packagestate\microsoft\`. It holds the 64-bit FILETIME entry *LastUploadedTime*, which registers the date/time of the last update of Your Phone data.

For a given user, the install time of Your Phone is kept in a 64-bit FILETIME format by the entry *InstallTime* of the following key: `HKU\SID\Software\Classes\Local Settings\Software\Microsoft\Windows\CurrentVersion\AppModel\Repository\Families\Microsoft\Microsoft.YourPhone_VERSION_8wekyb3d8bbwe`, where `VERSION` corresponds to the version of Your Phone, that is `1.0.20453.0_x64` in this study.

Windows Event Log. The usage of Your Phone does not produce specific entries in Windows Event Log. In our experiments, only `System` log had two types of entries related to Your Phone, with the following ID: *i*)

ID=43 and *ii*) ID=19. Both types of entries signalize the installation/update of the Your Phone application. Specifically, entries with ID=43 document the start of the installation/update, while entries ID=19 are logged when the installation/update operation ends successfully.

4.6. Artifacts after Uninstalling Your Phone

Uninstalling Your Phone practically eliminates all of its artifacts. Indeed, the whole content of the directory `Microsoft.YourPhone_8wekyb3d8bbwe` (Table 1) is removed. This means that all photos, as well as, the `phone.db` database are deleted. Furthermore, and conversely to the install/update process, the uninstall operation of Your Phone is not logged to any of Windows event logs.

The date/time of the uninstall process is kept in the entry `Microsoft.YourPhone_8wekyb3d8bbwe` of the registry under the key `\HKU\SID\Software\Microsoft\UserData\UninstallTimes`. The format of the timestamp entry is again a 64-bit FILETIME. Additionally, and as expected, Windows artifacts created by the execution of Your Phone such as Prefetch and Windows Timeline persist after Your Phone has been removed from the system.

5. Your Phone Analyzer

Your Phone Analyzer (YPA) is a python-based module for the Autopsy software[6]. Autopsy is a well known open source digital forensic software that harbors within a graphical user interface (GUI) many useful tools and functions for digital forensic examinations. The functionality of Autopsy can be extended through three types of modules: *i*) File ingest; *ii*) Datasource ingest; and *iii*) Report. YPA comprises two types of modules: a datasource ingest called `YPA_dataingest.py` (henceforth `YPA_dataingest`) and a report one, named `YPA_report.py` (henceforth `YPA_report`). YPA is available under a GPLv3 license[7]

5.1. YPA_dataingest

`YPA_dataingest` parses the `phone.db` file and inserts four different sets of artifacts into Autopsy: *i*) Contacts; *ii*) SMS; *iii*) MMS; and *iv*) Recovered

[6] `https://www.sleuthkit.org/autopsy/`

[7] `https://github.com/JVictorRS/YourPhoneAnalyzer`

rows, as partially shown in Figure 4. The *Contacts* set shows the contacts existing in `phone.db`, listing for each contact, the name, the last contact date/time, the last updated date/time, the number of contacted times and the phone numbers.

The SMS set displays for each SMS, the timestamp, the phone number and name of the recipient and whether the SMS was sent or received. Similarly, the MMS set displays the sent/received MMS. Finally, *recovered rows* holds the raw text content that was recovered from the freespace of the database. The recovery functionality relies on `undark`, an open source program to recover SQLite's deleted content [34]. Note that, as reported by Nemetez et al. [31], `undark` only correctly extracts ASCII-text, only providing meaningful output for 1-byte UTF-8 text, thus not decoding multi-bytes UTF-8 symbols such as two-byte special characters such as, à, ç, ã and three-byte symbols used for Chinese/Japanese/Korean (CJK) languages. To complement `undark`, YPA also runs DeGrazia's python script `sqlparse_v1.3.py` [35] to recover deleted records. Nemetz et al. report that `sqlparse_v1.3.py` has a high ratio for recovering text fields in deleted records, thus being appropriate for `phone.db` since the most meaningful records are text-based (phone numbers, names, SMS and MMS content). The output of both tools – `undark` and `sqlparse_v13.py` – is made available in raw format within Autopsy, respectively in `Rows Recovered (undark)` and `Rows Recovered (Delete parser)` sets. The module does not attempt to perform any interpretation of the data, only listing it, row by row.

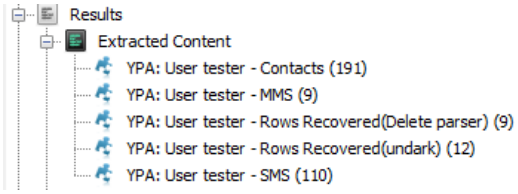


Figure 4: YPA-created artifact tree in Autopsy

5.2. YPA_report

Within the Autopsy module, `YPA_report` produces an HTML report that displays, in separate pages, *i*) the address book and the *ii*) conversations. Each conversation is shown under a layout that mimics SMS/MMS conversation in smartphones to ease the interpretation of conversations. An example

is shown in Figure 5. The address book is shown within an HTML table as depicted in Figure 6.

Figure 5: A conversation shown by YPA-report

ID	Address	Display name	Address type	# contacted (*)	Last contacted (**)	Last updated (***)
322	+3xxxxxxxxxx	João Silva	Main phone number	7	2019-02-25 13:55:11	2019-02-27 18:14:06
321	+3xxxxxxxxxx	Luís Andrade	Main phone number	5	2019-02-25 21:08:38	2019-02-27 18:14:06
208	+3xxxxxxxxxx	Miguel Frade	Mobile phone number	40	2019-02-28 14:00:57	2019-02-28 14:01:51
154	+3xxxxxxxxxx	Patrício Domingues	Mobile phone number	0	--	2019-02-27 18:14:01

(*) Might no be accurate (progresses by blocks of 10 for values larger than 10) and includes contact done via WhatsApp and other instant messaging APP

(**) Affected by WhatsApp and other instant messaging APP that handle SMS at the smartphone

(***) Affected by updates of the smartphone SMS application

Figure 6: Example of an address book rendered by YPA-report

6. Limitations

Your Phone presents several limitations regarding the data that can be harvested for digital forensics. The most relevant one is certainly the fact that it does not come installed by default on Windows 10. Note however, that in its first six months of existence, Your Phone Companion has been downloaded more than five million times, even considering that it requires Android 7+ and Windows 18.03 or above. A further limitation of Your Phone ecosystem that we found experimentally on smartphone with dual SIM: only

one SIM, that is, one phone number is processed by Your Phone, with the other SIM simply being ignored.

Another limitation is that Your Phone only keeps one month worth of SMS/MMS. However, since the SQLite database used to store the system data is not configured for *auto-vacuum* and *auto secure delete*, the probability of recovering some of the SMS/MMS is high. Moreover, as deleted data are kept on pages of the SQLite database, recovery feasibility does not depend on the storage disk technology. More limiting is the fact that multimedia attachments of MMS are not available within Your Phone and thus not available for digital forensic examiners.

The ability to solely access the last 25 photos/screenshots of the smartphone is another limitation, as for forensic purposes, it would be more interesting to access the whole repository of photos of the smartphone devices. Again, previous photos of which Your Phone made a local copy and which were then deleted to make place for more recent photos, may still be recoverable by carving the unallocated space of the PC's storage. Note however, that although recovery of deleted data is common in HDD, its success rate drops substantially when dealing with SSD disks [36].

As reported earlier, another limitation concerns the address book, which only comprises names and the associated phone numbers. It does not include other data such as email, physical address or date of birth. However, for investigative authorities, a phone number is often enough to fully identify an individual or organization, although email addresses can also provide valuable help in some cases.

From the point of view of accuracy, and as reported earlier, the fields *times_contacted* and *last_contacted_time* from the **Address** table have a behavior dependent on external factors such as the use of WhatsApp, Signal, etc. and thus should be taken with care to avoid misinterpretation of data.

7. Conclusion

This work presents a digital forensic approach to the Your Phone ecosystem. To the best of our knowledge, it is the first academic work focusing on forensically exploring Your Phone in a Windows 10 environment. As smartphones have become ubiquitous, they are often relevant in digital forensic examination. In some cases, access to a smartphone might not be possible, either because it cannot be found, or it is encrypted with an unavailable access code and forensic tools that can access the smartphone do not exist or

are simply too expensive for the examination allocated budget. As shown in this paper, in cases involving Android smartphone(s) and Windows 10 PC(s) with Your Phone installed and enabled, the forensic examiner has access through the PC to up to one-month of SMS/MMS, to a phone address book and up to 25 photos/screenshots of the smartphone. This way, exploiting Your Phone data that exist in PC(s) allows the forensic examiner to access some data of the smartphone. Moreover, as Windows' Your Phone application keeps data according to the Windows authenticated user, this opens the possibility to attribute data content to the authenticated individual.

For digital forensic practice, this works contributes with two python scripts wrapped in an Autopsy module to help forensic examiners to detect and leverage data that exists within the SQLite3 database that is central to the Your Phone Windows application. The scripts list and allow to export the flow and content of SMS/MMS and the address book. The scripts also provide for the recovery of deleted SMS/MMS and phone contacts. This way, when deleted SMS/MMS can be recovered, it might be possible to access more than one month worth of SMS/MMS and to recover past contacts that once existed in the address book. The paper also documents the folder structure where Your Phone data are kept, and in particular, the whereabouts of the 25 photos. As future work, we plan to follow the evolution of Your Phone ecosystem and update the YPA scripts to harvest any valuable data from a digital forensic perspective. We also plan to analyze Your Phone Companion application.

Acknowledgment

This work was partially supported by FCT and Instituto de Telecomunicações under project UID-EEA-50008-2013 and by CIIC under project UID/CEC04524/2016.

References

[1] B. X. Chen, Always on: how the iPhone unlocked the anything-anytime-anywhere future–and locked us in, Da Capo Press, 2011.

[2] E. Casey, B. Turnbull, Digital evidence on mobile devices, Academic Press, 3^{rd} edition, 2011, pp. 1–44.

[3] D. Quick, K.-K. R. Choo, Impacts of increasing volume of digital forensic data: A survey and future research challenges, Digital Investigation 11 (2014) 273–294.

[4] P. Crowley, E. Biggers, Adiantum: length-preserving encryption for entry-level processors, IACR Transactions on Symmetric Cryptology 2018 (2018) 39–61.

[5] Operating System Market Share Worldwide, Website (access on 2019-01-12), 2019. http://gs.statcounter.com/os-market-share.

[6] L. Kelion, Microsoft gives up on Windows 10 Mobile, Website (access on 2019-01-12), 2017. https://www.bbc.com/news/technology-41551546.

[7] G. Horsman, A. Caithness, C. Katsavounidis, A Forensic Exploration of the Microsoft Windows 10 Timeline, Journal of Forensic Sciences 64 (2019) 577–586.

[8] D. Rubino, 5 things you need to know about Microsoft's 'Your Phone' for Windows 10, Website (access on 2019-02-26), 2018. https://www.windowscentral.com/5-things-about-microsoft-your-phone.

[9] W. Smale, Why businesses are saving the humble text message, Website (access on 2019-01-19), 2017. https://www.bbc.com/news/business-41666820.

[10] S. K. Samanta, J. Woods, M. Ghanbari, Special delivery: An increase in MMS adoption, IEEE Potentials 28 (2009) 12–16.

[11] H. Carvey, The Windows Registry as a forensic resource, Digital Investigation 2 (2005) 201–205.

[12] B. Dolan-Gavitt, Forensic analysis of the Windows registry in memory, Digital Investigation 5 (2008) S26–S32.

[13] N. K. Shashidhar, D. Novak, Digital forensic analysis on prefetch files, International Journal of Information Security Science 4 (2015) 39–49.

[14] M. Kim, S. Lee, Forensic analysis using amcache.hve, in: Digital Forensics and Cyber Crime: 7th International Conference, ICDF2C 2015,

Seoul, South Korea, October 6-8, 2015. Revised Selected Papers, volume 157, Springer, p. 215.

[15] D. Quick, C. Tassone, K.-K. R. Choo, Forensic analysis of Windows thumbcache files, in: 20^{th} Americas Conference on Information Systems (AMCIS 2014), Savannah, 2014.

[16] B. Singh, U. Singh, A forensic insight into Windows 10 Jump Lists, Digital Investigation 17 (2016) 1–13.

[17] B. Singh, U. Singh, P. Sharma, R. Nath, Recovery of forensic artifacts from deleted jump lists, in: G. Peterson, S. Shenoi (Eds.), Advances in Digital Forensics XIV, Springer International Publishing, Cham, 2018, pp. 51–65.

[18] H. Chivers, C. Hargreaves, Forensic data recovery from the Windows Search Database, Digital Investigation 7 (2011) 114–126.

[19] P. Domingues, M. Frade, Digital forensic artifacts of the Cortana device search cache on Windows 10 Desktop, in: Availability, Reliability and Security (ARES), 2016 11th International Conference on, IEEE, pp. 338–344.

[20] B. Singh, U. Singh, A forensic insight into Windows 10 Cortana search, Computers & Security 66 (2017) 142–154.

[21] Y. Khatri, Forensic implications of System Resource Usage Monitor (SRUM) data in Windows 8, Digital Investigation 12 (2015) 53–65.

[22] B. Singh, U. Singh, Program execution analysis in Windows: A study of data sources, their format and comparison of forensic capability, Computers & Security 74 (2018) 94–114.

[23] A. Majeed, H. Zia, R. Imran, S. Saleem, Forensic analysis of three social media apps in windows 10, in: High-Capacity Optical Networks and Enabling/Emerging Technologies (HONET), 2015 12^{th} International Conference on, IEEE, pp. 1–5.

[24] D. Hintea, R. Bird, M. Green, An investigation into the forensic implications of the Windows 10 operating system: recoverable artefacts and significant changes from Windows 8.1, International Journal of Electronic Security and Digital Forensics 9 (2017) 326–345.

[25] R. C. Pandey, S. K. Singh, K. K. Shukla, Passive forensics in image and video using noise features: a review, Digital Investigation 19 (2016) 1–28.

[26] K. Akshatha, A. Karunakar, H. Anitha, U. Raghavendra, D. Shetty, Digital camera identification using PRNU: A feature based approach, Digital Investigation 19 (2016) 69–77.

[27] H. Chivers, Navigating the Windows Mail database, Digital Investigation 26 (2018) 92–99.

[28] M. H. Goadrich, M. P. Rogers, Smart smartphone development: iOS versus Android, in: Proceedings of the 42nd ACM technical symposium on Computer science education, 2011, ACM, pp. 607–612.

[29] C. Boyd, P. Forster, Time and date issues in forensic computing—a case study, Digital Investigation 1 (2004) 18–23.

[30] L. Rutledge, SMIL 2.0: XML for Web multimedia, IEEE Internet Computing 5 (2001) 78–84.

[31] S. Nemetz, S. Schmitt, F. Freiling, A standardized corpus for SQLite database forensics, Digital Investigation 24 (2018) S121–S130.

[32] S. Jeon, J. Bang, K. Byun, S. Lee, A recovery method of deleted record for SQLite database, Personal and Ubiquitous Computing 16 (2012) 707–715.

[33] H. Xie, K. Jiang, X. Yuan, H. Zeng, Forensic analysis of Windows registry against intrusion, International Journal of Network Security & Its Applications 4 (2012) 121.

[34] P. L. Daniels, Undark - a SQLite deleted and corrupted data recovery tool, Website (access on 2019-02-17), 2019. `http://pldaniels.com/undark/`.

[35] M. DeGrazia, SQLite-Deleted-Records-Parser: recovering deleted entries in SQLite database, Website (access on 2019-02-17), 2019. `https://github.com/mdegrazia/SQLite-Deleted-Records-Parser`.

[36] R. Winter, SSD vs HDD–data recovery and destruction, Network Security 2013 (2013) 12–14.