A Targeted Data Extraction System for Mobile Devices

Sudhir Aggarwal, Gokila Dorai, Umit Karabiyik, Tathagata Mukherjee, Nicholas Guerra, Manuel Hernandez, James Parsons, Khushboo Rathi, Hongmei Chi, Temilola Aderibigbe, et al.

Chapter 5

A TARGETED DATA EXTRACTION SYSTEM FOR MOBILE DEVICES

Sudhir Aggarwal, Gokila Dorai, Umit Karabiyik, Tathagata Mukherjee, Nicholas Guerra, Manuel Hernandez, James Parsons, Khushboo Rathi, Hongmei Chi, Temilola Aderibigbe and Rodney Wilson

Abstract Smartphones contain large amounts of data that are of significant interest in forensic investigations. In many situations, a smartphone owner may be willing to provide a forensic investigator with access to data under a documented consent agreement. However, for privacy or personal reasons, not all the smartphone data may be extracted for analysis. Courts have also opined that only data relevant to the investigation at hand may be extracted.

This chapter describes the design and implementation of a targeted data extraction system for mobile devices. It assumes user consent and implements state-of-the-art filtering using machine learning techniques. The system can be used to identify and extract selected data from smartphones in real time at crime scenes. Experiments conducted with iOS and Android devices demonstrate the utility of the targeted data extraction system.

Keywords: Mobile devices, privacy, targeted data extraction, iOS, Android

1. Introduction

Smartphones contain large amounts of data that are of significant interest in forensic investigations. However, these devices have in essence become personal data repositories and the privacy of their data is a serious concern. A landmark 2014 ruling by the U.S. Supreme Court in Riley v. California and subsequent rulings based on this case suggest that it may not be enough to obtain a warrant to conduct a search of a smartphone, but it may also be required to restrict the search to specific items on the device that relate to the crime being investigated. What is needed

is a forensically-sound system that can perform targeted (selective) data extraction under a documented consent agreement. Commercial tools such as Cellebrite UFED Physical Analyzer have great utility, but they do not support targeted data extraction.

This chapter describes the design and implementation of a prototype software system that supports targeted data extraction from iOS and Android devices in a forensically-sound manner. The system runs on a solid state drive connected to a laptop, which is connected to a mobile device of interest on which the targeted data extraction app is downloaded. Metadata and content filtering rules in the app support targeted data extraction under a consent agreement signed by the device owner. Metadata filtering rules enable data of specific types with relevant creation dates/times and locations to be extracted. Content-based filtering leverages machine learning to exclude non-relevant data and ensure that user data privacy is maintained. Forensic soundness is realized using the eDiscovery Reference Model [19] and dynamic/live analysis techniques drawn from network and cloud forensics [17, 26].

2. Related Work

Several tools support full data acquisition from iOS and Android devices. Commercial tools include Cellebrite UFED Physical Analyzer, Paraben Electronic Evidence Examiner, Oxygen Forensic, AccessData Mobile Phone Examiner Plus, Microsystems XRY, Magnet Acquire and Blackbag Mobilyze. These tools attempt to acquire as much data as possible via logical and physical acquisitions. However, they do not support on-device or off-device selective methods for extracting only the data that is relevant to investigations.

Considerable research has focused on forensic data extraction and analysis. Some of this work deals with the extraction of specific types of artifacts from cloud drives and social networking applications [2, 4, 26]. Other research has been directed at general forensic data extraction techniques for mobile devices [14, 25]. Interested readers are referred to [23] and [28] for detailed discussions about iOS and Android device forensics, respectively.

The concept of "real-time triage" has become increasing important and there has been some work on building such systems [9, 27]. Another important aspect is data privacy in the context of digital forensics in general and mobile forensics in particular [3, 31].

Machine learning (see, e.g., [24]) and its applications have gained considerable attention in recent years. Deep learning (see, e.g., [20]) has been successfully applied in areas ranging from image recognition [18]

to natural language translation [10]. Open-source frameworks such as Caffe [15], Theano [8] and TensorFlow [1] have been developed for implementing deep representational learning using neural networks. State-of-the-art processors in modern smartphones make it feasible to perform image analysis and classification, including facial detection, using deep learning models such as Inception [33], Open NSFW [22] and MobileNet [13].

At this time, mobile device forensic tools are unable to perform on-device targeted data extraction as described in this chapter. In fact, the available tools only extract images of device content and enable the images to be queried and analyzed in an off-device manner. Moreover, these tools do not have the ability to filter data using machine learning techniques.

3. System Overview

The targeted data extraction system (TDES) for mobile devices has three components: (i) data identification system; (ii) data acquisition system; and (iii) data validation system.

- **Data Identification System:** The data identification system is responsible for identifying the relevant files based on metadata and content. Input to the system is broadly driven by a consent form and is fine-tuned by the forensic investigator using a specially-designed user interface.

  Smartphone data comes in a variety of types. The basic categories of smartphone data are photos (images), videos, messages and contact lists. Each category is associated with metadata that describes aspects of the data, such as time (when an image was placed on the device), location (where the image was taken) and sender and receiver (of text and multimedia messages).

  Note that metadata is different from content. For example, a query based on a date range – "photos taken within the past week" – uses metadata about photos. However, a query for photos containing "weapons" would require content-based filtering. The data identification system incorporates state-of-the-art machine learning, natural language processing and data mining algorithms to perform content-based filtering.

- **Data Acquisition System:** The data acquisition system interacts with the data identification system to retrieve targeted files from a smartphone in a forensically-sound manner. Data acqui-

sition corresponds to data collection; therefore, the data that is acquired is the desired evidence.

The data acquisition system incorporates two components: (i) TDES manager; and (ii) TDES app. The TDES manager is a system-on-chip that resides on a portable bootable drive. The manager boots up in Windows 10 when connected to a laptop or workstation. The target smartphone is connected to the same laptop or workstation in order to deploy the TDES app on the target smartphone. The user interface of the TDES app enables an investigator to provide input to the data identification system. Finally, the filtered data from the target phone is transferred to the TDES manager.

- **Data Validation System:** The data validation system, which is integrated with the data identification and data acquisition systems, ensures that data is transferred in a forensically-sound manner. It performs appropriate hashing to insure data integrity. Additionally, it generates a log timeline that documents all the steps taken by the TDES system during "live analysis." Finally, the data validation system produces a report that documents the needs of the investigator (e.g., queries), the data analysis that was performed and the data that was selected.

The data identification and data acquisition systems are described together because their abstractions are closely coupled. Also, because the system only performs logical data extractions, it is assumed that relevant data is not stored in hidden or deleted files. Furthermore, the focus is on rapid targeted data extraction – how to define what data is to be extracted, how to ensure that data extraction is done in a forensically-sound manner and how to perform data extraction very rapidly.

4. Targeted Data Extraction

In order to motivate the development of the model for targeted data extraction, it is instructive to present potential application scenarios. These scenarios, which were suggested by forensic investigators, involve instances where consent is natural and the ability to filter data would be very useful:

- A car accident where a bystander has taken photos or a video of the incident.
- A drug overdose incident where the victim's phone has information about drugs and drug dealers.

- A suicide case where the victim's phone may contain relevant texts, email and photos.

- A domestic violence situation where the victim's phone has photos that document the physical abuse.

- A major incident where several individuals have captured videos and photos of the perpetrators, their weapons and their vehicles.

In several shooting incidents, bystanders and/or companions have recorded the events on their phones [30]. The Boston Marathon bombing case had a massive amount of digital evidence from multiple sources [32]. In these and many other incidents, automated selective data extraction would have been very useful.

Data of value in forensic investigations is classified as follows:

- **User-Created Data:** This includes contacts and address books, SMS messages, MMS messages, calendars, voice memos, notes, photographs, video/audio files, maps and location information, voice mail and stored files.

- **Internet-Related Data:** This includes browsing histories, email and social networking data.

- **Third-Party Application Data:** This includes messaging data (text, voice, video and pictures) from applications such as Facebook, WhatsApp and Skype.

As discussed above, the TDES app, which is deployed on the target device, is responsible for filtering and transferring the data to the TDES manager. This method of data extraction is called "on-device acquisition." In this type of acquisition, only the data that is filtered by the TDES app is transferred from the phone. No other data on the device is ever pushed to the TDES manager.

However, for some iPhone data types, it is not possible to selectively extract relevant data without "jailbreaking" the phone or using the iTunes backup system. Since jailbreaking is not employed in this work, the only option is to use the iTunes backup system. Selective data extraction from iTunes is referred to as "backup acquisition." In backup acquisition, all the available data from the iTunes backup is moved to the TDES manager, which extracts the relevant data and deletes the backup after the extraction is completed.

Table 1. On-device metadata-based extraction.

Data Category	Metadata Type	iOS	Android
Photos	Date and time	Yes	Yes
Photos	Location	Yes	Yes
Photos	Album type	Yes	Yes
Videos	Date and time	Yes	Yes
Videos	Location	Yes	Yes
Contacts	Name	Yes	Yes
Contacts	Number	Yes	Yes
Contacts	Area code	Yes	Yes
Contacts	Email	Yes	Yes
Calendar Events	Date	Yes	Yes
Reminders	Date	Yes	Yes
Photos	Third-party app	No	Yes
Messages/SMS/MMS	Date and time	No	Yes
Messages/SMS/MMS	Contact number	No	Yes
Call Logs	Incoming call	No	Yes
Call Logs	Outgoing call	No	Yes
Call Logs	Missed call	No	Yes
Call Logs	Date and time	No	Yes
Notes	Search string	No	No
Notes	Date and time	No	No
Voice Memos	Date and time	No	No
Web History	Date and time	No	No
Email	Date and time	No	No
Facebook Messages	Date and time	No	No
WhatsApp Messages	Date and time	No	No
LinkedIn Messages	Date and time	No	No
WeChat Messages	Date and time	No	No
Viber Messages	Date and time	No	No

4.1 On-Device Metadata-Based Filtering

Table 1 shows the data that can and cannot be extracted by the TDES app in the on-device mode via metadata filtering. The first part of the table shows the data that can be extracted from iPhones (iOS devices) and Android phones. The second part of the table shows the data that can be extracted from Android phones, but not from iPhones (e.g., photos captured by third-party apps such as Facebook and WhatsApp). The third part of the table shows data that the TDES app currently cannot extract from iPhones and Android phones.

- **iPhones:** System interfaces for iPhones are delivered in the form of packages called frameworks (Figure 1). The TDES app for

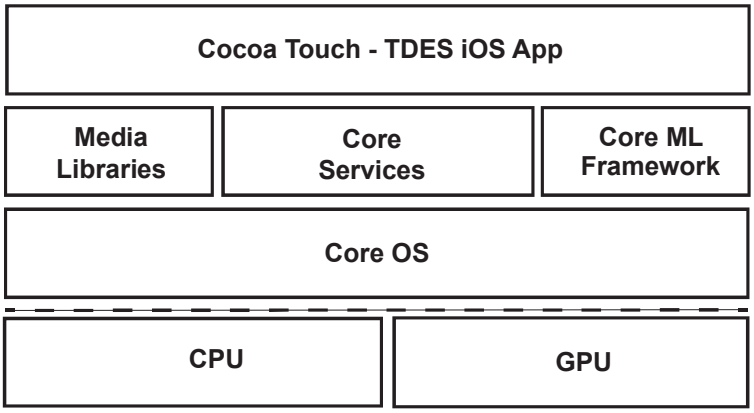


Figure 1. iOS frameworks.

iPhones uses several frameworks in Media Libraries and Core Services. The Photos framework provides direct access to photo and video assets managed by the iPhone Photos app. The AVKit framework provides a high-level interface for playing video content. The CoreLocation framework provides location and orientation information. The EventKit framework provides an interface for accessing calendar events. The Contacts framework provides access to user contacts and functionality for organizing contact information.

- **Android Phones:** Figure 2 shows the Android operating system stack. The TDES Android app, which is deployed in the application layer, leverages services provided by the Application framework, which includes the Content Provider, Activity Manager, Resource Manager and View [12]. Content Provider provides access to a range of data and other services used for design and implementation.

4.2 On-Device Content-Based Filtering

Trained machine learning models are developed using supervised learning techniques, including learning using deep neural nets. A trained model can be incorporated in the iOS or Android TDES app using the appropriate framework. The model can be used directly by retraining the final layer or by using heuristics based on model outputs.

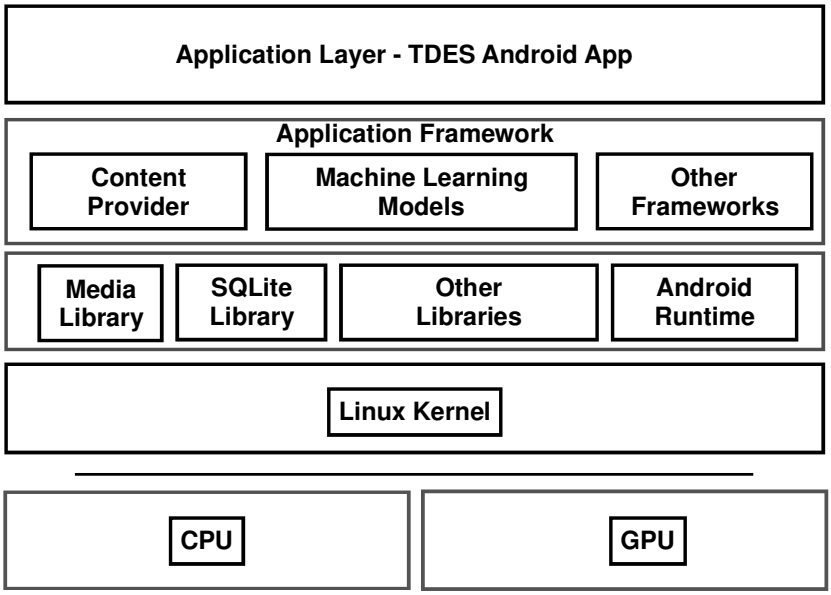


Figure 2. Android operating system stack.

The current versions of the TDES app employ adapted trained models from Inception-v3 [16], MobileNet [13] and Open NSFW [22] to classify photos and videos. Interested readers are referred to the bibliography for details about the accuracy of these models. The TDES apps are able to identify photos containing weapons, people, vehicles, drugs, websites, skin exposure and gadgets. The accuracy of the adapted models is discussed in Section 5.

The Core ML framework [5] is used for on-device content-based filtering on iPhones. Core ML provides support for several machine learning frameworks, including Vision and GameplayKit.

The TensorFlow Lite framework [35] is used for on-device content-based filtering on Android phones. The trained model and related labels are used in conjunction with a shared object file `libtensorflow_inference.so`, which is written in C++. The Java API `libandroid_tensorflow_inference_java.jar` [1, 29] is used to interface with Android platforms.

4.3 Off-Device Backup-Based Filtering

Table 2 shows the data that can and cannot be extracted by the TDES app in the off-device mode via metadata filtering. The first part

Table 2. Off-device metadata-based extraction.

Data Category	Metadata Type	iOS	Android
Photos	Date and time	Yes	Yes
Photos	Location	Yes	Yes
Photos	Album type	Yes	Yes
Videos	Date and time	Yes	Yes
Videos	Location	Yes	Yes
Contacts	Name	Yes	Yes
Contacts	Number	Yes	Yes
Contacts	Area code	Yes	Yes
Contacts	Email	Yes	Yes
Calendar Events	Date	Yes	Yes
Reminders	Date	Yes	Yes
Photos	Third-party apps	Yes	Yes
Messages/SMS/MMS	Date and time	Yes	Yes
Messages/SMS/MMS	Contact number	Yes	Yes
Call Logs	Incoming call	Yes	Yes
Call Logs	Outgoing call	Yes	Yes
Call Logs	Missed calls	Yes	Yes
Call Logs	Date and time	Yes	Yes
Notes	Search string	Yes	No
Notes	Date and time	Yes	No
Voice Memos	Date and time	Yes	No
Web History	Date and time	Yes	No
Email	Date and time	Yes	No
Facebook Messages	Date and time	*	No
WhatsApp Messages	Date and time	Yes	No
LinkedIn Messages	Date and time	*	No
WeChat Messages	Date and time	*	No
Viber Messages	Date and time	*	No

of the table shows the data that can be extracted from iPhones and Android phones. The second part shows the data that can be extracted from Android phones, but not from iPhones. The third part shows data that the TDES app currently cannot extract from iPhones and Android phones. Note that a table entry marked with an asterisk (*) corresponds to an item that was not investigated.

- **iPhones:** Apple iOS security mechanisms do not permit applications that execute on an iPhone to extract certain types of content (second and third sections of Table 1). Therefore, this content is acquired from an iTunes backup. The `idevicebackup2` command supported by the open-source `libimobiledevice` [21] is employed.

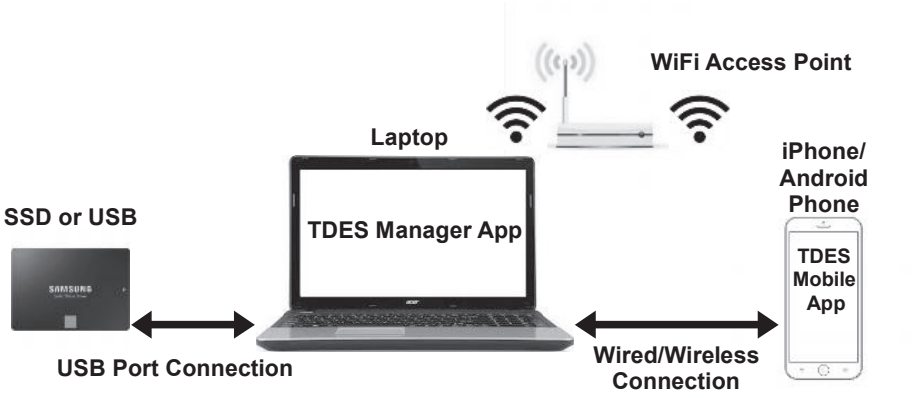


Figure 3. TDES communications paradigm.

Other standard, albeit complex, techniques can also be used to extract data from a backup.

- **Android Phones:** In the case of Android phones, any data that can be extracted off-device can also be extracted on-device; therefore, on-device extraction is employed. However, data from the third-party applications in Table 1 cannot be extracted using on-device acquisition when the phone is not rooted. Experiments with rooted and non-rooted Android phones did not reveal an Android equivalent of the iTunes backup mechanism.

4.4 TDES Communications

Communications between the TDES manager and the TDES app on a target phone is an important component of the TDES system. Figure 3 shows the communications paradigm that is implemented on iPhones and Android phones. The forensic investigator is provided with a portable TDES boot drive (e.g., SSD drive or USB stick) that is preloaded with a bootloader for a Windows 10 machine, TDES manager and the tools necessary to install the TDES app on the target phone. All the extracted data is sent back to the boot drive by the TDES app; reports pertaining to the extracted data also reside on the boot drive.

Any available Windows 10 system can be used to boot into the TDES manager, which runs in an isolated environment on the drive. After booting up, the TDES manager must have Internet access if the target device is an iPhone.

The steps for targeted data extraction are:

- The boot drive containing the TDES manager is inserted into a laptop.
- The Windows 10 operating system boots up and the TDES manager starts its execution.
- A wired connection using a USB cable is established from the laptop to the phone. The TDES app is installed. In the case of an iPhone, a hotspot is needed to connect to Apple in order to sign the code and acknowledge trust in the developer.
- After the app is downloaded, the phone may be disconnected from the laptop.
- A wireless or wired two-way communications channel is set up between the TDES manager and TDES app for data transfer.
- The targeted data extracted by the TDES app is exported to the TDES manager and reports are generated for the extracted data.

Note that no copies of data or residual data from the export process are stored on the phone.

- **TDES App Installation on iPhones:** Only applications from sources approved by Apple can be executed on iPhones that are not jailbroken. Apple iOS requires that all executable code must be signed with a certificate issued by Apple. Third-party apps must have signed certificates to ensure that they do not load any tampered or self-modifying code [6].

  The TDES implementation uses Cydia Impactor [34] to sign the TDES app code. The procedure involves the generation of an iOS App Store Package (IPA) file of the TDES app using the XCode Archive utility. This application archive file stores an iPhone app. In order to sign the code, Impactor logs into the Apple Developer Center and downloads the developer's provisioning profile and iOS development certificate. Logging into the Apple Developer Center requires an Internet connection. Impactor signs the IPA file content in a depth-first manner starting with the deepest folder level. After the signing is done, Impactor installs the TDES app on the iPhone. All these tasks are automated by an AutoHotKey script [7] that executes after the TDES manager boots; thus, no actions are required to be performed by the forensic investigator.

- **TDES App Installation on Android Phones:** The Android operating system permits only signed applications to be installed

on an Android phone. As long as an application is signed and does not attempt to update another application, it can be self-signed – this approach is adopted in the TDES implementation. The output of the compilation is an APK file. Note that no other authentication is necessary.

The TDES app is installed after the APK file is stored on the target phone. For simplicity and ease of use, an Android debug bridge is employed for communications between the host computer and target phone. The Android debug bridge requires the phone to be placed in the USB debugging mode; this mode is turned off after the app is installed.

- **TDES Data Transfer Protocol:** The communications channel between the TDES app and TDES manager must ensure that the extracted data is transmitted with forensic integrity and that all data modifications are detected and documented. Furthermore, data that is modified inadvertently or intentionally during the chain of custody is also identified and documented.

  This is implemented by hashing essentially every file and computing a final hash value, which is exported to the TDES manager. Note that the hashing is done on the phone. If required, the final hash value could be sent to the phone's owner, the forensic investigator or to a third party.

  The iPhone implementation employs a socket-based data transfer protocol. Since the iPhone implementation requires a hotspot in any case, a wireless link is used for communications between the app and the manager.

  The Android implementation uses an Android debug bridge, which supports socket-level communications. Since Android applications are natively written in Java, `ServerSockets` and `Sockets` are employed. A wired connection is used for the Android communications protocol.

4.5 User Interface

The user interface, which runs as part of the app on the target phone, enables a forensic investigator to specify the selection criteria for data extraction. At this time, the interfaces are somewhat different for iPhones and Android phones. An optional PDF consent form is provided by the TDES manager. In the case of an iPhone, after the data extraction criteria are specified using the app, a digital consent form that specifies the data to be extracted can be completed on the app itself. In the case of

an Android phone, a broad consent form is completed on the app first. This consent form ensures that only the relevant subset of data specified using the app is, in fact, extracted.

A useful bookmarking feature is provided by the TDES app. Consider a situation where a dataset has been extracted using a set of filters. The forensic investigator who set up the filters can display the results and do a quick data review on the phone itself before deciding what data to actually export to the TDES manager (i.e., bookmarked data). For example, if the investigator selected a set of images of weapons obtained during a certain time period, then he/she could review the images and select a subset of relevant images by bookmarking the subset.

Discussions with a former prosecutor and a current defense attorney indicated that bookmarking is a useful feature, but it may introduce bias during the evidence collection process. Consequently, the current implementation enables bookmarking to be turned on or off. Alternatively, both options may be selected, producing two versions of the exported data – the bookmarked version and the original version. If needed, an investigator could export all the data that could be examined under the consent and filtering definitions, including possible exculpatory data.

- **iPhone App Interface:** Figure 4 shows the iPhone TDES app interface. The initial choices for a forensic investigator to define are: (i) when (specific date ranges, today, last week, last month, etc.); (ii) where (current location, location within a certain number of miles, location determined by city, state or zip code, etc.); and (iii) what (data types – photos, videos, calendar, call logs, messages and contacts).

  Additional filtering options – generally, content filtering – may be defined. For example, if photos and videos are of interest, then the content filtering options supported are the inclusion or exclusion of weapons, places, vehicles, drugs, websites, gadgets, skin exposure, pornography and favorites. If the exclude skin exposure option is selected, then the app filters the corresponding images, and displays and exports the remaining images.

  The last screen of the interface enables the investigator to display the selected data on the device, export the data, or both. A consent form is displayed before the data is exported to the TDES manager.

- **Android App Interface:** Figure 5 shows the Android TDES app interface. The app first presents a screen for specifying the data categories to be extracted; the same categories of data as the iPhone app are supported. Selecting any of these data types leads

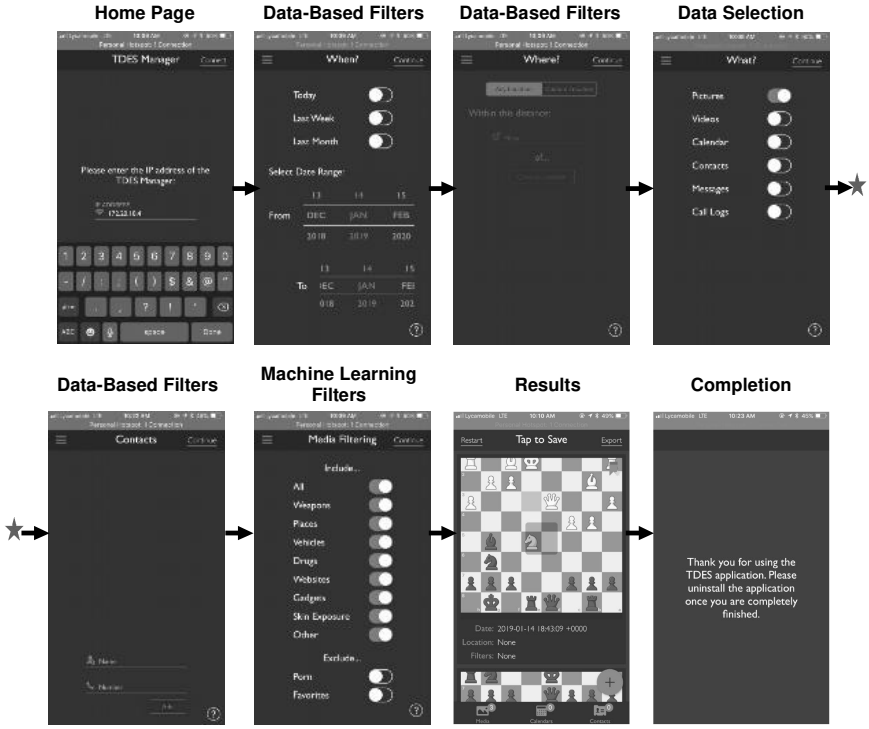
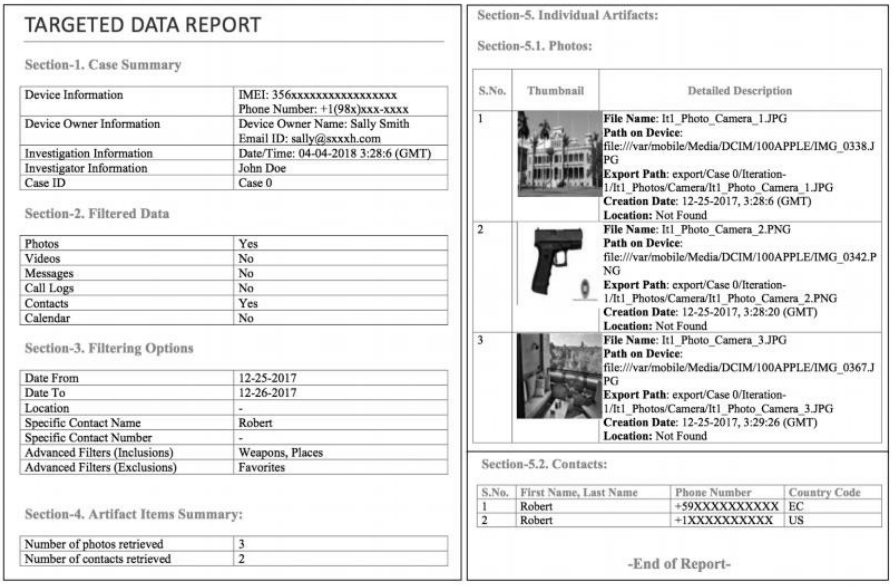


Figure 4. iPhone TDES app user interface.

Figure 5. Android TDES app user interface.



TARGETED DATA REPORT

Section-1. Case Summary

Device Information	IMEI: 356xxxxxxxxxxxxxxx Phone Number: +1(98x)xxx-xxxx
Device Owner Information	Device Owner Name: Sally Smith Email ID: sally@sxxxh.com
Investigation Information	Date/Time: 04-04-2018 3:28:6 (GMT)
Investigator Information	John Doe
Case ID	Case 0

Section-2. Filtered Data

Photos	Yes
Videos	No
Messages	No
Call Logs	No
Contacts	Yes
Calendar	No

Section-3. Filtering Options

Date From	12-25-2017
Date To	12-26-2017
Location	-
Specific Contact Name	Robert
Specific Contact Number	-
Advanced Filters (Inclusions)	Weapons, Places
Advanced Filters (Exclusions)	Favorites

Section-4. Artifact Items Summary:

Number of photos retrieved	3
Number of contacts retrieved	2

Section-5. Individual Artifacts:

Section-5.1. Photos:

S.No.	Thumbnail	Detailed Description
1		**File Name**: It1_Photo_Camera_1.JPG **Path on Device**: file:///var/mobile/Media/DCIM/100APPLE/IMG_0338.JPG **Export Path**: export/Case 0/Iteration-1/It1_Photos/Camera/It1_Photo_Camera_1.JPG **Creation Date**: 12-25-2017, 3:28:6 (GMT) **Location:** Not Found
2		**File Name**: It1_Photo_Camera_2.PNG **Path on Device**: file:///var/mobile/Media/DCIM/100APPLE/IMG_0342.PNG **Export Path**: export/Case 0/Iteration-1/It1_Photos/Camera/It1_Photo_Camera_2.PNG **Creation Date**: 12-25-2017, 3:28:20 (GMT) **Location:** Not Found
3		**File Name**: It1_Photo_Camera_3.JPG **Path on Device**: file:///var/mobile/Media/DCIM/100APPLE/IMG_0367.JPG **Export Path**: export/Case 0/Iteration-1/It1_Photos/Camera/It1_Photo_Camera_3.JPG **Creation Date**: 12-25-2017, 3:29:26 (GMT) **Location:** Not Found

Section-5.2. Contacts:

S.No.	First Name, Last Name	Phone Number	Country Code
1	Robert	+59XXXXXXXXXX	EC
2	Robert	+1XXXXXXXXXX	US

-End of Report-

Figure 6. TDES summary report for an iPhone.

to a new screen with another set of choices providing additional filtering options for metadata and content filtering. The Android app interface also has provisions for first defining a broad consent form that restricts further data selections. It also supports data bookmarking, display and export.

Both versions of the app interface support a fair amount of metadata and content filtering. For example, call logs can be filtered by name and number as well as by date and time. Contacts can be filtered by name and number. Messages can be filtered by name and number as well as by date and time. Videos and photos can be filtered by location, date, time and various implemented content using machine learning models.

4.6 Reporting and Forensic Integrity

A common interface using the JSON object format [11] is implemented for the selected export of data from the iPhone and Android phone apps. The JSON structure facilitates the description of the extracted data as well as hash values and reporting information. For example, a report may need to document when the TDES app began its execution and when the extraction was completed. Although the data transfer is primarily from the app to the manager, some information, such as the forensic investigator's name, phone owner's name and case number, is

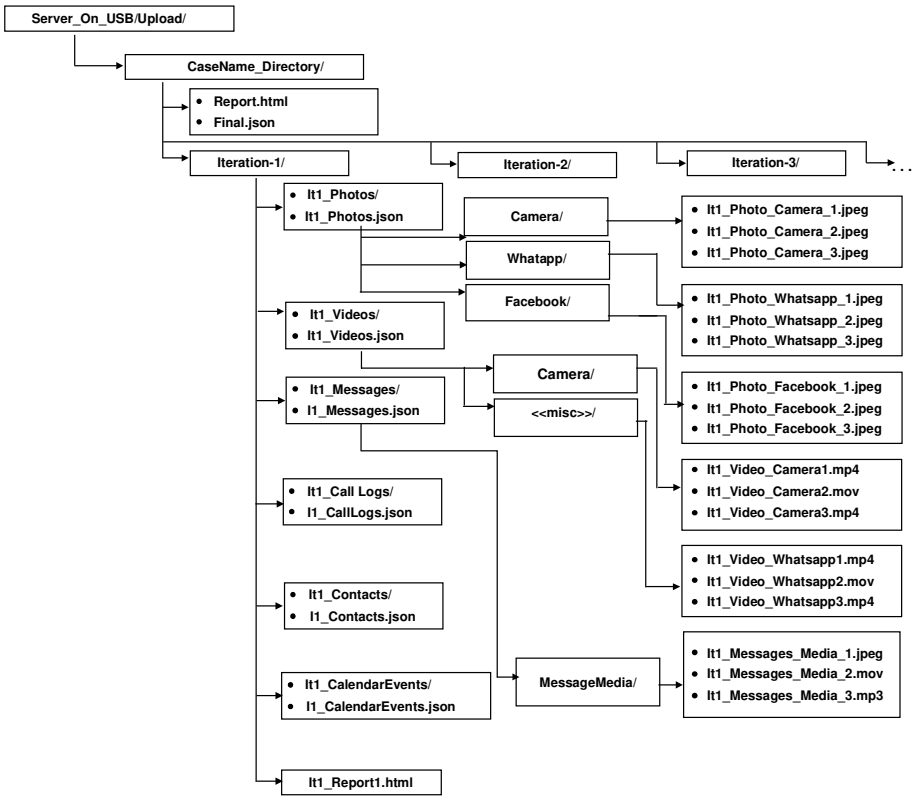


Figure 7. Output file structure.

passed from the manager to the app. The Android TDES app extracts additional information such as the IMEI, phone number and email address associated with the phone. In the case of the iPhone TDES app, this information must be entered in the manager. Figure 6 shows a sample report generated for an iPhone.

- **TDES Directory Structure on the Boot Drive:** Figure 7 shows the directory structure created for storing evidence on the boot drive. The structure is designed to ensure data integrity and support reporting. A directory is created for each case. The complete report is stored as an HTML file in this directory. The JSON files, including `Final.json`, are discussed below. The extracted data is stored as one or more iterations of requests made by the investigator. In each iteration, every data category has a separate directory and a JSON file is associated with the directory.

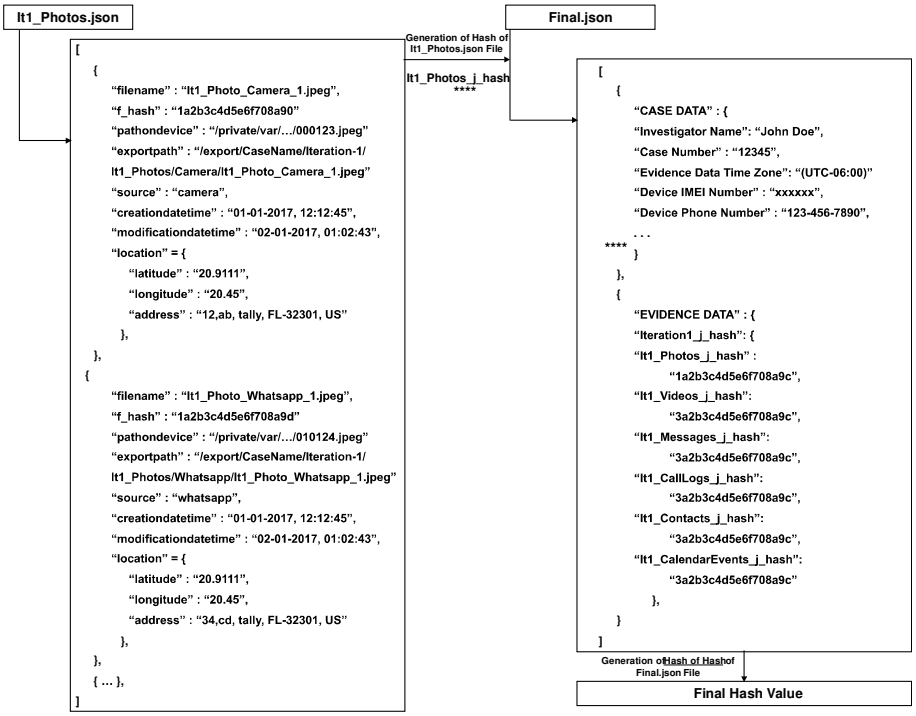


Figure 8. Example JSON files.

- **JSON Format for Data Transfer:** The JSON format is used to describe the structure of the exported data, which is used to create reports in the HTML format. Figure 8 shows example JSON files.

  Assume that a set of photos has been extracted using metadata and content filters. Auxiliary information about each photo is transferred to the TDES manager along with the actual image file. The TDES apps for iPhones and Android phones create this information in the same format. After the information is transferred to the TDES manager, a report manager creates the actual report. Hashes are also transferred as part of the JSON files. As shown in Figure 8, the `It1_Photos.json` file is structured into arrays of arrays containing (key, value) pairs. For example, `creation_date` is a key and its value is the string `01-01-2017`.

  Considerable information is exported in a JSON file. The key `filename` has a `value` string associated with it, which corresponds to the `name` of the actual photo image. The actual image is stored as a separate file as defined by the key `exportpath`. The hash

value of the actual photo file is stored in the JSON structure and is defined by the key `f_hash`.

- **Hashing and Data Integrity:** SHA-1 hashes are used to ensure the integrity of the data transferred to the TDES manager; other hash algorithms may be used if needed. Each file `filename` defined in a JSON file has a hash associated with the file called the `f_hash`.

  Consider the `It1_Photos.json` file shown in Figure 8 and the key `filename` with value `It1_Photo_Camera_1.jpg`. A hash `f_hash` is associated with it (shown in the figure) because the actual file is stored in a separate location. Therefore, any file in the directory that is not a JSON file has a hash value stored in a JSON file.

  Next, every JSON file has a JSON hash `j_hash` associated with the file. For example, the hash value computed for the file `It1_Photos.json` is stored as the key `It1_Photos_j_hash` in file `It1_Hashes.json`. For each iteration `n`, the hash of `Itn_Hashes.json` is stored in the `Final.json` file. The hash of `Final.json` is called `Final_hash`. This hash value ensures that no file in any case directory can be modified without detection.

  The `Final_hash` value computed by an app is sent to the TDES manager and stored in `Report.htm`. The manager can independently compute the `Final_hash` value to check if any changes occurred during the data transfer. Hash values are computed at intermediate points for several reasons, including to facilitate the granular transfer of data and check if the transfer is correct. Checking the extracted files against known files is also simplified. The TDES manager (or app) could also email a copy of the `Final_hash` value to the phone's owner, forensic investigator or third party.

5. Experiments and Results

Several experiments were conducted to evaluate the accuracy and speed of selective data filtering on iPhone and Android phones. The metadata filtering accuracy should be 100% because the Apple and Android frameworks were employed; however, manual checks of metadata filtering were still performed.

The performance of the prototype system was also compared against two commercial tools, Paraben EEE and Magnet AXIOM, which are used by law enforcement. As mentioned above, neither of these tools (nor Cellebrite) can perform selective data extraction as implemented by the prototype system. Note that the Cellebrite commercial tool was not evaluated because this tool (like the others) essentially performs a

Table 3. Devices used in the experiments and device content.

Model/Version	NIC	P	V	M	CL	CO	CA
Device 1 iPhone-8 (iOS v11.2.1)	Lightning port	10,307	178	208	482	1,102	148
Device 2 iPhone-7 (iOS v11.2.5)	Lightning port	2,621	109	5	155	6	46
Device 3 iPhone-6 Plus (iOS v11.2.2)	Lightning port	2,566	102	15,978	714	384	265
Device 4 Samsung Galaxy S7 (v7.0, Nougat)	Micro-USB 2.0	100	6	37	7	20	17
Device 5 Moto G3 (v6.0, Marshmallow)	Micro-USB 2.0	191	7	25,420	429	1,889	780
Device 6 Samsung Galaxy S7 Edge (v7.0, Nougat)	Micro-USB 2.0	249	22	13,362	500	240	337

NIC: Network Interface Card; **P: Photos**; **V: Videos**; **M: Messages**; **CL: Call Logs**; **CO: Contacts**; **CA: Calendar**

physical acquisition of all the phone data and then enables the user to analyze the data off-device.

Three iPhones and three Android phones were used in the experiments. Table 3 provides details about the phones and their contents. Apple Devices 1 and 3, which belong to the authors of this chapter, contained real user data. Apple Device 2 contained synthetic, non-copyrighted data that is available for reuse over the Internet. Similarly, Android Devices 5 and 6 contained real user data and belong to the authors; Apple Device 4 contained synthetic data. Table 3 also shows the total numbers of artifacts of each data category residing in each test device. The TDES boot drive used was a SanDisk Extreme 128 GB stick. A ThinkPad X1 Carbon laptop was used as the boot drive and to connect to the test phones.

iPhone Results. The iPhone experiments employed Devices 1, 2 and 3. Table 4 shows the results for on-device metadata-based filtering for

Table 4. On-device metadata-based filtering for iPhones.

Device 1 Experiments				
Category: Filter	**Artifacts**	**Display Time**	**Export Time**	**Size**
1-Photos: 12/24/17–12/27/17	2/10,307	0.7 s	3.58 s	2.33 MB
2-Photos: Within 10 miles*	418/10,307	1.21 s	42 m, 64 s	822 MB
3-Videos: 09/1/17–01/31/18	34/178	1.20 s	51 m, 11 s	1,038 MB
4-Videos: Within 10 miles*	–	–	–	–
5-Videos: Current location*	4/178	0.2 s	17 m, 2 s	405 MB
6-Contacts: "Puppy"	3/1,102	2.57 s	0.6 ms	–
7-Contacts: "Robert"	–	–	–	–
8-Contacts: (xxx)xxx–xxx	1/1,102	0.12 s	0.8 ms	–
9-Calendar: 01/01/18–01/15/18	19/148	0.14 s	0.6 ms	–
10-Photos: 08/30/17–09/15/17	91/10,307	0.7 s	4 m, 1 s	236 MB
Videos: Any location	1/178			
11-Photos: 08/31/17	9/10,307	0.73 s	1 m, 1 s	51 MB
Videos: Within 50 miles	1/178			
12-Videos: Last week Within 10 miles	3/178	0.4 s	1 m, 2 s	47 MB

Device 1. Each experiment (row) focuses on a specific data category and filter. For each experiment, the total number of artifacts selected out of the total number of artifacts on the device is shown (e.g., in the case of the 1-Photos experiment, 2/10,307 means that two photos out of 10,307 photos on the device were extracted). The metadata filtering was 100% accurate based on manual checking (e.g., a phone feature such as Photos Album count). The table also shows the times required to display data on the target device and to export data to the TDES manager (via a wired connection). The recorded times show that TDES is feasible for in-field targeted data extraction. The amounts of exported data are also shown. Note that a table entry marked with an asterisk (*) corresponds to an item whose location depends on the physical location of the phone.

Table 5 shows the results of experiments for off-device backup-based metadata filtering for Device 3. The results for messages and call logs are shown. As discussed earlier, the backup-based procedure involved the TDES manager acquiring a complete backup from iTunes; thus, there is no export time. Note, however, that the forensic investigator must still specify the filtering that must be performed by the TDES app. The

Table 5. Off-device backup-based filtering for iPhones.

Device 3 Experiments Category: Filter	Artifacts	Display Time
1-Messages: None	15,978/15,978	1.95 s
2-Messages: 10/03/17–12/30/17	510/15,978	0.33 s
3-Messages: (***)***–***	1,016/15,978	0.29 s
4-Call Logs: None	683/683	0.29 s
5-Call Logs: 01/14/17–08/14/17	297/683	0.27 s
6-Call Logs: (***)***–***	40/683	0.27 s
7-Messages: 01/14/17–08/14/17 Call Logs: (***)***–***	738/15,978 35/683	0.32 s
8-Messages: (***)***–*** Call Logs:	1,016/15,978 40/683	0.28 s

accuracy of metadata filtering is always 100% based on manual analysis using iTunes.

Table 6. On-device metadata and content filtering for iPhones (Inception-v3).

Device 2 Experiments Category: Filter	Content Filter	Display Time	Export Time	Accuracy (%)
1-Photos: 12/25/17	Weapons	9.82 s	4.69 s	97.22
2-Photos: Within 10 miles	Weapons	20.31 s	–	–
3-Photos: 12/25/17–12/29/17	Weapons	19.05 s	4.34 s	94.50
4-Photos: 12/25/17–12/29/17	Places	17.73 s	9.93 s	88.07
5-Photos: 12/25/17–12/29/17	Vehicles	17.06 s	0.72 s	100.00
6-Photos: 12/25/17–12/29/17	Drugs	16.26 s	0.33 s	96.33
7-Photos: 12/25/17–12/29/17	Websites	16.66 s	7.03 s	99.08
8-Photos: 12/25/17–12/29/17	Gadgets	17.33 s	7.88 s	89.91
9-Photos: 12/25/17–12/29/17	Skin exposure	16.23 s	8.68 s	100.00

Table 6 shows the results of the experiments using Device 2 photos for various combinations of metadata and content filtering. The test iPhone had 2,621 photos with 109 photos in the date range 12/25/17 to 12/29/17, and 72 of these photos were taken on 12/25/17. The Inception-v3 model was used for content filtering. Rows 1–3 of the table focus on filtering for "weapons." In the case of Row 2, content

Table 7. On-device metadata-based filtering for Android phones.

Device 5 Experiments				
Category: Filter	**Artifacts**	**Display Time**	**Export Time**	**Size**
1-Photos: 02/03/18–02/05/18	2/191	0.31 s	2.32 s	6.56 MB
2-Photos: Current location	1/191	0.63 s	10.58 s	46.1 MB
3-Videos: 12/19/17–02/03/18	3/7	0.89 s	3.91 s	16.6 MB
4-Videos: Current location	7/7	0.90 s	13.09 s	190 MB
5-Calendar: 05/29/17–05/30/17	85/780	1.03 s	2.40 s	13 KB
6-Messages: "aaabb"	32/25,420	1.23 s	14.23 s	7 KB
7-Messages: (***)***–***	5/25,420	0.92 s	1.25 s	4 KB
8-Call Logs: "aaabb"	9/429	0.49 s	6.59 s	5 KB
9-Call Logs: (***)***–***	11/429	0.89 s	11.2 s	6 KB
10-Messages: (***)***–***	100/25,420	1.25 s	14.08 s	199.1 MB
Photos: 01/28/18–02/05/18	6/191			
Videos: Current location	7/7			
11-Messages: 12/12/17–02/05/18	1,000/25,420	1.02 s	3.89 s	258 KB
Call Logs: (***)***–***	8/429			
12-Messages: 09/12/17–09/29/17	300/25,420	1.65 s	18.2 s	236.1 MB
Calendar: 09/12/17–09/29/17	5/780			
Photos: Current location	1/191			
Videos: Current location	7/7			

filtering was not applied because none of the weapons photos were taken within 10 miles. Rows 4–9 focus on content filters that would be relevant to law enforcement. The times required for display and export are shown for each experiment. The accuracy measure expresses how well the Inception-v3 model performs content filtering. The accuracy computations involved the creation of a confusion matrix for each experiment, following which the accuracy was computed as:

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \times 100 \quad (1)$$

where TP denotes true positive; TN denotes true negative; FP denotes false positive; and FN denotes false negative.

Android Phone Results. The Android phone experiments employed Devices 4, 5 and 6. Table 7 shows the results for on-device metadata-based filtering for Device 5. Each experiment (row) focuses on a specific

Table 8. On-device metadata and content filtering for Android phones (MobileNet).

Device 4 Experiments				
Category: Filter	**Content Filter**	**Display Time**	**Export Time**	**Accuracy (%)**
1-Photos: 11/12/17–02/02/18	Weapons	35 s	1.5 s	75.68
2-Photos: Current location	Weapons	1.3 s	0.81 s	100.00
3-Photos: 10/12/17–12/02/17	Vehicles	37.4 s	1.56 s	25.00
4-Photos: Current location	Vehicles	1.4 s	1.3 s	100.00
5-Photos: 12/01/17–01/13/18	Drugs	34.69 s	1.2 s	92.06
6-Photos: Current location	Drugs	1.2 s	0.0 s	71.43
7-Photos: 08/11/17–12/31/17	Skin exposure	33.08 s	2.48 s	92.21

data category and filter. Note that the display and export times are very good. For example, in the case of the 12-Messages experiment, exporting 236 MB of device artifacts required only 18.2 seconds.

Table 8 shows the results of seven experiments using Device 4 photos for various combinations of metadata and content filtering. The MobileNet model from TensorFlowLite was used for metadata and content filtering. The display and export time results are excellent. The accuracy measure, computed using Equation (1), expresses how well the MobileNet model performs content filtering. The results are modest; better machine learning models will have to be developed to improve the accuracy of content filtering.

Comparison with Commercial Tools. Several experiments were conducted to compare the data export times for TDES against the times required by two commercial tools, Paraben and Magnet AXIOM. iPhone Device 2 and Android Device 4 were used in the experiments. Table 9 shows the experimental results – the iPhone comparisons are in the top half of the table and the Android comparisons are in the bottom half of the table. The app installation time (AIT) is the time period from the instant the target device was connected to the laptop to the time when a data selection can be made (in the case of TDES, this is when a data selection can be made on the target device; in the case of Paraben and Magnet AXIOM, this is when a data selection choice can be made on the laptop). The backup acquisition time (BAT) is the time taken for backup-based acquisition. Note that, in the case of TDES, the exported data was stored on a flash drive whereas, in the case of Paraben and Magnet AXIOM, the exported data was stored on the laptop hard drive.

Table 9. Export time comparisons for iPhone Device 2 and Android Device 6.

Item	TDES	Paraben	Magnet AXIOM
Device 2 (iPhone)			
AIT	52 s	10 m	9 m
BAT	26 m (2 GB)	20 m	38 m, 54 s (4.1 GB)
Call Logs	(BAT) 15 ms	0.1 s	0.4 s
Messages	(BAT) 16 ms	0.1 s	0.3 s
Contacts	1.8 ms	0.2 s	0.3 s
Calendar	2 ms	0.2 s	0.3 s
Photos	39 m, 3 s (2,621 files)	–	80 m (29,488 files, 2.30 GB)
Videos	30 m, 15 s (109 files)	–	4 m (438 files, 1.73 GB)
All Media	Not needed	32 m	93 m (48,701 files, 2.69 GB)
Device 6 (Android)			
AIT	14 s	5 s	NA
BAT	NA	NA	29 m
Call Logs	1 s	40 s	1 m, 17 s
Messages	4 m, 9 s	17 m, 3 s	1 m, 21 s
Contacts	1 s	2 m, 11 s	1 m, 11 s
Calendar	6 s	1 m, 5 s	1 m, 14 s
Photos	42 s (249 files)	–	14 m, 41 s (13,711 files)
Videos	14 s (22 files)	–	1 m, 38 s (62 files)
All Media	NA	43 s	NA

- **iPhone Comparison:** The installation time of the TDES app on the iPhone was 52 seconds. Paraben and Magnet AXIOM had to first create a backup of the iPhone data. In the case of Paraben, backup creation (20 minutes) occurs in conjunction with application initialization (10 minutes) whereas Magnet AXIOM has a separate backup creation step of 38 minutes and 54 seconds after 9 minutes of application initialization. Note that TDES has a backup acquisition time only when extracting call logs and messages.

- **Android Phone Comparison:** The installation time of the TDES app on the Android phone was 14 seconds. Since Magnet AXIOM uses backup-based acquisition, a backup must be created before extracting any artifacts. For example, when extracting call logs, Magnet AXIOM created a backup that took 29 minutes followed by call log extraction that took one minute and 17 seconds. The TDES app required 14 seconds for app installation and one second

for data export. In contrast, Paraben required five seconds for initialization and 40 seconds for data export.

In the case of Paraben and Magnet AXIOM, the only choices available for acquisition are the broad categories shown in Table 9. Paraben does not extract photos and videos separately; it provides one option for all media artifacts. However, experiments revealed that selecting this option resulted in the extraction of metadata associated with media artifacts, not the artifacts themselves.

6. Conclusions

The targeted data extraction system described in this chapter supports the acquisition of relevant data from iOS and Android devices in a forensically-sound manner. It implements state-of-the-art metadata and content filtering functionality based on machine learning techniques. Forensic soundness is realized using the eDiscovery Reference Model [19] and dynamic/live analysis techniques drawn from network and cloud forensics [17, 26]. The design assumes that a phone is voluntarily provided to law enforcement under a documented consent agreement. However, it is equally applicable to situations where a court orders that a smartphone passcode must be provided for evidence recovery or where a smartphone memory dump (e.g., from a cloud backup) with an intact filesystem is available. The targeted data extraction system is currently being provided to law enforcement for testing and feedback, with the goal of incorporating additional features and capabilities.

Acknowledgements

This research was supported in part by the National Institute of Justice, Office of Justice Programs, U.S. Department of Justice under Award No. 2016-MU-CX-K003. The opinions, findings and conclusions or recommendations expressed in this chapter are those of the authors and do not necessarily reflect the opinions of the U.S. Department of Justice.

References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu and X. Zheng, Tensorflow: A system for large-scale machine learning, *Proceedings of the Twelfth USENIX Symposium on Operating Systems Design and Implementation*, pp. 265–283, 2016.

[2] N. Al Mutawa, I. Baggili and A. Marrington, Forensic analysis of social networking applications on mobile devices, *Digital Investigation*, vol. 9(S), pp. S24–S33, 2012.

[3] A. Aminnezhad, A. Dehghantanha and M. Abdullah, A survey of privacy issues in digital forensics, *International Journal of Cyber-Security and Digital Forensics*, vol. 1(4), pp. 311–323, 2012.

[4] C. Anglano, Forensic analysis of WhatsApp Messenger on Android smartphones, *Digital Investigation*, vol. 11(3), pp. 201–213, 2014.

[5] Apple, Core ML, Cupertino, California (`developer.apple.com/documentation/coreml`), 2017.

[6] Apple, iOS Security, iOS 12.3, Cupertino, California (`www.apple.com/business/docs/iOS_Security_Guide.pdf`), 2019.

[7] AutoHotkey Foundation, AutoHotkey (`autohotkey.com`), 2019.

[8] J. Bergstra, F. Bastien, O. Breuleux, P. Lamblin, R. Pascanu, O. Delalleau, G. Desjardins, D. Warde-Farley, I. Goodfellow, A. Bergeron and Y. Bengio, Theano: Deep learning on GPUs with Python, *Proceedings of the BigLearning Workshop*, vol. 3, 2011.

[9] G. Cantrell, D. Dampier, Y. Dandass, N. Niu and C. Bogen, Research toward a partially-automated and crime-specific digital triage process model, *Computer and Information Science*, vol. 5(2), pp. 29–38, 2012.

[10] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk and Y. Bengio, Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, arXiv:1406.1078 (`arxiv.org/abs/1406.1078`), 2014.

[11] D. Crockford, The application/json Media Type for JavaScript Object Notation (JSON), RFC 4627, 2006.

[12] Google, Android Developer Manual, Mountain View, California, 2017.

[13] A. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam, MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, arXiv:1704.04861 (`arxiv.org/abs/1704.04861`), 2017.

[14] M. Husain, I. Baggili and R. Sridhar, A simple cost-effective framework for iPhone forensic analysis, *Proceedings of the International Conference on Digital Forensics and Cyber Crime*, pp. 27–37, 2010.

[15] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama and T. Darrell, Caffe: Convolutional architecture for fast feature embedding, *Proceedings of the Twenty-Second ACM International Conference on Multimedia*, pp. 675–678, 2014.

[16] Keras Team, Keras: Deep Learning for Humans, GitHub (`github.com/keras-team/keras`), 2019.

[17] S. Khan, A. Gani, A. Abdul Wahab, M. Shiraz and I. Ahmad, Network forensics: Review, taxonomy and open challenges, *Journal of Network and Computer Applications*, pp. 214-235, 2016.

[18] A. Krizhevsky, I. Sutskever and G. Hinton, ImageNet classification with deep convolutional neural networks, in *Communications of the ACM*, vol. 60(6), pp. 84–90, 2017.

[19] D. Lawton, R. Stacey and G. Dodd, E-Discovery in Digital Forensic Investigations, CAST Publication Number 32/14, Centre for Applied Science and Technology, Home Office, London, United Kingdom, 2014.

[20] Y. LeCun, Y. Bengio and G. Hinton, Deep learning, *Nature*, vol. 521(7553), pp. 436–444, 2015.

[21] `libimobile.org`, `libimobiledevice`: A cross-platform software protocol library and tools to communicate with iOS devices natively (`www.libimobiledevice.org`), 2019.

[22] J. Mahadeokar, Open NSFW Model, GitHub (`github.com/yahoo/open_nsfw`), 2017.

[23] S. Morrissey and T. Campbell, *iOS Forensic Analysis: For iPhone, iPad and iPod touch*, Apress, New York, 2010.

[24] K. Murphy, *Machine Learning: A Probabilistic Perspective*, MIT Press, Cambridge Massachusetts, 2012.

[25] D. Quick and M. Alzaabi, Forensic analysis of the Android filesystem YAFFS2, *Proceedings of the Ninth Australian Digital Forensics Conference*, pp. 100–109, 2011.

[26] V. Roussev, A. Barreto and I. Ahmed, Forensic Acquisition of Cloud Drives, arXiv:1603.06542 (`arxiv.org/abs/1603.06542`), 2016.

[27] V. Roussev, C. Quates and R. Martell, Real-time digital forensics and triage, *Digital Investigation*, vol. 10(2), pp. 158–167, 2013.

[28] N. Scrivens and X. Lin, Android digital forensics: Data, extraction and analysis, *Proceedings of the ACM Turing 50th Celebration Conference – China*, article no. 26, 2017.

[29] A. Shekhar, Android TensorFlow Machine Learning Example, *MindOrks Blog* (`blog.mindorks.com/android-tensorflow-machine-learning-example-ff0e9b2654cc`) March 6, 2017.

[30] Y. Steinbuch and J. Tacopino, Woman records horrific scene after boyfriend is fatally shot by police, *New York Post*, July 7, 2016.

[31] P. Stirparo and I. Kounelis, The Mobileak Project: Forensic methodology for mobile application privacy assessment, *Proceedings of the International Conference on Internet Technology and Secured Transactions*, pp. 297–303, 2012.

[32] M. Stroud, In Boston bombing, flood of digital evidence is a blessing and a curse, *CNN*, April 18, 2013.

[33] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, Rethinking the inception architecture for computer vision, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

[34] Team Cydia, Cydia Impactor (`cydia-app.com/cydia-impactor`), 2019.

[35] TensorFlow, Introduction to TensorFlow Lite (`www.tensorflow.org/mobile/lite`, 2018.