Article

AutoPoD-Mobile—Semi-Automated Data Population Using Case-like Scenarios for Training and Validation in Mobile Forensics

Margaux Michel [1], Dirk Pawlaszczyk [2,*] and Ralf Zimmermann [1]

[1] Central Office for Information Technology in the Security Sector (ZITiS), Zamdorfer Str. 88, 81677 Munich, Germany; margaux.michel@zitis.bund.de (M.M.); ralf.zimmermann@zitis.bund.de (R.Z.)
[2] Faculty of Computer Sciences, Hochschule Mittweida—University of Applied Sciences, Technikumplatz 17, 09648 Mittweida, Germany
* Correspondence: pawlaszc@hs-mittweida.de

Abstract: The complexity and constant changes in mobile forensics require special training of investigators with datasets that are as realistic as possible. Even today, the generation of training data is almost exclusively done manually. This paper presents a novel open-source framework called AutoPoD-Mobile. The framework supports the creation of case-based scenarios. Even more, the semi-automated provision of datasets for mobile forensics is enabled. Thus, the behavior of suspects interacting with each other can be simulated. The result combines mobile device data from normal device usage and case-related information. This way helps validate mobile forensic tools, test new techniques, and create realistic training datasets in the mobile forensics domain. The results of a proof-of-concept trial in a realistic deployment environment will also be presented. The paper concludes with a discussion of the results and identifies options for future improvements.

Keywords: mobile device digital forensics; digital forensics analysis; digital evidence; device population; training; forensics dataset; behavioral simulation

check for updates

Citation: Michel, M.; Pawlaszczyk, D.; Zimmermann, R. AutoPoD-Mobile—Semi-Automated Data Population Using Case-like Scenarios for Training and Validation in Mobile Forensics. *Forensic Sci.* **2022**, *2*, 302–320. https://doi.org/10.3390/forensicsci2020023

Academic Editors: Mario Antunes and Patricio Domingues

Received: 30 January 2022
Accepted: 22 March 2022
Published: 25 March 2022

1. Introduction

In recent decades, the amount of digital evidence has been constantly increasing. With the technical development of physical devices, the amount of information has also changed drastically. In the context of mobile forensics, the focus of the forensic targets and the corresponding techniques had to be strongly adapted [1,2]. As a result of these changes, the forensic software we use in our examinations receives continuous updates. Forensic experts must consider state-of-the-art techniques applicable to solve new technical challenges. To address this issue, we require datasets as close to reality as possible for education, training, and the validation of forensic tools and our workflows. Even though incriminating data from real criminal cases are the most accurate source, these data are not suitable for training [3–5]. Legal issues and data privacy aspects prevent widespread use. According to Grajeda et al. [4], only 3.8% of the newly created datasets were published. As a probable reason, the authors state that many researchers do not like to share their datasets.

For this reason, datasets used for training consist preferably of synthetically generated data. With these restrictions in mind, providing sharable datasets, which are as realistic as possible, is one of the most important prerequisites. In so doing, we ensure an appropriate level of quality both for the scientific research community and for forensic experts' training [2].

In order to provide realistic data for training situations, each institution usually creates its own emulated "incriminating" digital data source for investigation. Especially in mobile forensics, the generation and deployment of such models are mainly made manually.

It usually takes several days for an expert to create and deploy a single case, and the quality highly depends on previous experience in placing synthetic artifacts on mobile devices [3,5,6].

In particular, the realistic generation of communication data for messenger services, such as WhatsApp or Signal, is particularly time-consuming. Simultaneously, the generated sample data must meet several requirements at once. For example, they must cover normal wear-and-tear, white noise, and the actual digital traces covered during the investigation [3].

As the expected data on a mobile device changes drastically with the behavior and habits of the user, the sample cases should contain different user behavior profiles.

The support of age-related user behavior and gender-specific characteristics in the use of cell phones are just two examples of this. Instead of just providing devices with a certain amount of evidence to find, the training should simulate a crime case to be as realistic as possible. The consideration of aberrant time profiles and certain geo-position tracks are further aspects that play an essential role in storytelling and data generation for realistic training sets. Finally, we must consider different regional or cultural aspects in data generation. The most publicly available examples use English, which does not reflect the situation in many countries [7].

In the past, government organizations already addressed the problem of generating case data for mobile devices [8]. There are guidelines from CTFF [9] and NIST [10] that provide specific recommendations and guidance on creating a forensic image for training purposes or tool testing. The results are primarily catalogs of requirements and sample datasets, which are suitable for certain tasks, such as testing the performance of particular forensic tools (see Table 1). However, in the context of training mobile forensic experts, these provided datasets do not include complex cases with realistic conditions.

Table 1. Properties of public datasets & repositories for mobile forensics.

	CFReDS/NIST	**CSAFE**	**Digital Corpora**	**DFRWS**
Dataset Type	Tool Testing & Training	Mobile app usage & Geo Locations	Extraction Images	Challenge
Date	2018	2020	2011–2020	2011–2022
Available Datasets	Chip-Off Device Images	Databases	Android 2.2 physical Image & IOs Backups	Different Smartphone extraction Images
Case-Driven	no	no	no	yes
Language	English	English	English	English
Wear and Tear	no	partly	yes	yes
White Noise	no	no	no	partly
Age and Gender profiles	no	no	no	no
Geo Tracking Information	no	yes	no	partly

As with computer forensics, where forensic images are frequently used for the training of analysts, the idea of providing image files was mapped to the mobile forensics context. Considering the very short launch cycles of new cell phone generations with major technical changes both in hardware and the operating system—a considerable difference compared to classic computer forensics—these datasets have a limited lifetime.

Aside from the mobile forensic images provided by NIST, several datasets, including disk images, mobile phone dumps, and network captures, are available in a digital forensic corpora [11]. Nevertheless, most of the images provided for mobile devices are more than eight years old in the majority and only a limited number of complete scenarios are available. Another popular evidence repository is offered by the Center for Statistics and Applications in Forensic Evidence (CSAFE) (https://forensicstats.org/data/, accessed on the 31 January 2022). However, the data provided does not look at a closed criminal case or forensic image file. Instead, dataset and statistics are provided for research purposes.

In addition to the guidelines and test repositories, several recent scientific publications address the problem of forensics dataset creation and provision [4,12,13]. For example, within the TraceGen framework [14], the emulation and automatic creation of specific user activities is addressed (see Table 2). Unfortunately, the article mainly points out Windows-based image creation. It is not transferable to mobile devices. In another contribution, the authors discuss a model consisting of the maintenance of base hard drive images and a methodology for creating, storing, categorizing, clustering, and indexing evidence packages [15]. The project, called EviPlant, focuses specifically on the deployment or rollout process of forensics datasets and attempts to automate it. Another current example is the hystck framework [16]. This project deals with the generation of synthetic datasets and the focus lies on communication traffic in the computer network that is as realistic as possible.

Table 2. Frameworks for creation and provision of synthetic forensics datasets.

	TraceGen	**EviPlant**	**Hystck**	**3LSPG**
Framework Type	realistic network traffic & app behavior on windows	generation of synthetic datasets for computer networks	reation and distribution hard drive images	creation of content based on discrete time Markov chains
References	[14]	[15]	[16]	[17]
Date	2020	2021	2018	2014

All the frameworks presented so far address specific aspects of training data generation coming from the domain of computer forensics. Unfortunately, we cannot fully transfer the techniques to the context of mobile forensics. User behavior, software update cycles, hardware evolutions, and security restrictions create unique challenges in the mobile domain [18]. A realistic training model should consider all of these circumstances. Even the public datasets discussed in the beginning are of limited value for training purposes. Some of them are outdated. Other datasets only have a specific scope, which is not comparable to a real criminal case. Today, there is a significant gap between the datasets offered and the data needed for training purposes. Currently, no suitable publicly available solution supports at least a partially automated generation of realistic scenarios. Even today, this task must be solved entirely by hand.

In this contribution, we present AutoPoD-Mobile—Automatic Population of Data for Mobile Forensics—a proof-of-concept implementation of an extensible framework to automatically populate mobile devices with synthetic data. It takes a case scenario and the behavior of the acting persons into account and helps to populate mobile devices with data from normal device usage as well as case-related information; compared to completely manual generation of training data, this results in a considerably shorter lead time. The scripts provided facilitate the consideration of regional peculiarities, age- and gender-specific aspects, the creation of white noise, as well as realistic, logically sequenced chats. This significantly reduces the gap between real and artificially generated data. The aim is to improve the quality of training data in this way even if the dataset is synthetic. In doing so, AutoPoD-Mobile helps with the validation of mobile forensic tools, testing new techniques, and the creation of realistic training datasets in the field of mobile forensics.

In Section 2, we first describe a methodology and a framework for generating a realistic dataset. As we evaluate the system in a practical experiment leading to a ring trial in mobile forensics, we provide insights on the setup and the fictive case. In Section 3, we outline the result of this work. We further discuss the potential and limitations of the current approach, fundamental challenges when creating artificial training data in mobile forensics, and identify future research questions. The contribution ends with a short conclusion.

2. Materials and Methods

The following section describes all elements that led to and are needed to reproduce mobile devices' automated generation of data. It was part of the FORMOBILE project with the incentive of creating a ring trial for digital forensics experts. The ring trial required test data with specific properties, i.e., being realistic, complex, temporal, of a certain amount, and forensically sound. To achieve this, we generated deterministic data over multiple weeks, following a fictive case scenario involving multiple devices and different data types.

For the purpose of the ring trial and creating a comparable challenge for the forensic experts, the generation and population steps required certain degree of automation, which led to several advantages compared to manual population: the population of multiple devices in parallel with (almost) identical forensic artifacts, increased complexity of the data, less manual labor, and reduced chance of human errors, e.g., mistakes with the content or recipient of a message and the simulation of complex behavior patterns. To validate the results and ensure that the population process was successful, we analyzed the mobile devices, extracting the information and inspecting the artifacts with forensic tools.

2.1. Devices

The scripts developed in this context were tested with various mobile devices. As there are many mobile devices on the market and the field is not homogeneous, we selected different manufacturers as a cross-section of standard devices or devices with common properties or hardware features. Please keep in mind that while Android provides an AOSP core, every manufacturer is allowed to change parameters and adjust the firmware, including their applications and provide special launchers for the customers.

Table 3 provides the list of the devices we used and includes the most common operating systems—different versions of Android and iOS—that covered the majority [19] of the European market share as of December 2021. Furthermore, we included mobile phones from the three most common manufacturers in Europe: Apple, Samsung, and Huawei, which represent a market share of 36%, 31%, and 11%, respectively [20], choosing models from low-end to high-end devices. Please note that we concentrated our efforts on Android devices and only developed rudimentary support for iOS.

Table 3. List of devices used for the development and validation of the automated scripts.

Manu-Facturer	Model	Model Number	Release Year	Price at Release	CPU/Architecture	OS Version	Device Type
Alcatel	1SE	5030D	2020	120 €	Octa-core (4 × 1.6 GHz Cortex-A55 & 4 × 1.2 GHz Cortex-A55); 3 GB RAM	Android 10.0	low end
Apple	iPhone X		2017	1.000 €	Hexa-core 2.39 GHz (2× Monsoon + 4× Mistral); 3 GB RAM		high end
Huawei	Mate 20 Lite	SNE-LX1	2018	300 €	Octa-core (4 × 2.2 GHz Cortex-A73 & 4 × 1.7 GHz Cortex-A53); 4 GB RAM	Android 8.1 (Oreo); EMUI 8.2	high end
Samsung	Galaxy A40	SM-A405F	2019	220 €	Octa-core (2 × 1.77 GHz Cortex-A73 & 6 × 1.59 GHz Cortex-A53)	Android 9.0 (Pie)	mid end
Samsung	Galaxy A50	SM-A505F	2019	250 €	Dual SIM; Octa-core (4 × 2.3 GHz Cortex-A73 & 4 × 1.7 GHz Cortex-A53); 4 GB RAM	Android 9.0 (Pie)	mid end
Samsung	Galaxy S9	SM-G960F	2018	490 €	Octa-core (4 × 2.7 GHz Mongoose M3 & 4 × 1.8 GHz Cortex-A55); 4 GB RAM	Android 8.0 (Oreo)	high end
XGody	S20 Lite	-	2017	70 €	MTK6580 Quad Core 1.3 GHz ARM Cortex A7; 1 GB RAM	Android 10.0 [1]	low end

[1] As reported by the manufacturer, the OS version is in fact Android 6.0.

In addition to regular devices, we also included a so-called clone phone from the vendor XGody in the selection. Such phones look precisely like their counterfeited models, although they may be completely different from the hardware and software, e.g., an iPhone look-alike using an Android system. As there is a market for such devices, we need to target them in light of mobile forensics.

2.2. Tools

As mobile devices follow breakneck development cycles both with software (apps and operation system) updates and the development tools changing rapidly, we chose to work only using open source tools in our setup. This allows us to benefit from similar update cycles and makes reproduction and sharing of our scripts easier.

Furthermore, we chose not to reverse engineer any Android apps manually, limiting our scripts to publicly available APIs and protocols. Because of this, we can maintain a higher degree of stability, as these interfaces should work identically (or only with minor changes) over multiple releases.

The scripts in this paper were developed using Python 3.8. For the communication and interaction with the devices, we used the following tools: Chrome driver, Google Cloud Platform API, Google Cloud Platform (for calendar and email tokens), WhatsApp Web Interface, Reddit automatic downloader, Android Debug Bridge (ADB), and Appium Settings App (for GPS spoofing).

2.3. Datasets

Automatically generating data for mobile devices implies the need for datasets containing conversations, emails, pictures, and pre-generated files to store on the devices. We also looked into GPT-2 [21] or GPT-3 [22] generated datasets for the conversations and emails, but this seemed to go beyond our requirements at this stage of the framework development. We, therefore, made use of open-source datasets that are publicly available. The datasets that we used in the current framework are freely exchangeable to fit the users' needs and serve here as proof of concept:

Conversation between two persons on WhatsApp: We first started with two-person conversations on WhatsApp and needed a well-annotated and simple open-source dataset. We used the Topical-Chat dataset [23] from the Amazon Alexa AI team. It is annotated in eight subjects, i.e., fashion, politics, books, sports, or music, and contains 235,000 messages. Each message is also annotated with a sentiment, i.e., angry, disgusted, fearful, sad, or happy, that allows us to include emojis.

Group conversation in WhatsApp and emails: For group conversations, we could not use the Topical Chat dataset mentioned above since it only includes two agents. In the context of email messages, the dataset did not provide longer text blocks. Therefore, we looked at different sources for conversations matching these unique requirements and finally chose the Reddit platform (see https://reddit.com (accessed on 29 January 2022)). Here, we used different subreddits, i.e., the SeriousConversation for emails and CasualConversation for WhatsApp group conversations. To prevent potential data privacy issues, a script downloads, filters, concatenates, re-formats, and anonymizes the messages. As only the number of participants per conversation and an anonymized sender ID per message is needed to (re)generate conversations, we can easily map them to fictitious persons in our case scenario.

Pictures for the camera roll and the gallery application: In addition to conversations, we required open-source pictures to fill the media gallery of the mobile devices. We used the published Visual Genome dataset [24] from Stanford University to this extent. It contains more than 108,000 pictures with different content, e.g., persons, landscapes, animals, or inanimate objects. Please note that we did not use the annotations of the images at this point.

2.4. Simulation of Human Behavior with Smartphones

As we intended to create realistic data on the mobile devices, we had to adjust both the quantity and quality of the information stored on the devices compared to a manual approach. While the quantity aspect was solved using the available datasets and scripts that automatically populated the devices, improving the quality is a more significant challenge.

First, we needed to adjust the model of the device owner from merely playing a particular part in the overall case to a human being with daily life. As such, every person reacts to events in their life, their surroundings, or events triggered by the mobile device. This leads to follow-up actions, e.g., reading a message, taking a picture, or opening an application, and thus, the automated scripts needed to simulate a person's behavior.

The second aspect of improving the quality considers the timeline and the chain of events, which is essential for the forensics analysis: At which time was a person at an exact location, when did this person receive a message, when was a particular picture taken? Events are chronological, and so is the data accumulation on mobile devices. Automated scripts should mimic this behavior, and thus, the timestamps should be realistic.

Another critical aspect to consider is the degree of personalization, which also reflects on the data on a mobile device. This means that two distinct persons will not collect and store the same information on their mobile devices. The reasons may include age, personality and interests, the chosen usage of the mobile device (work/private), culture, location, or specific habits.

All these aspects should be considered to generate realistic data for the test devices and improve the overall quality. This leads to a balance between behavior simulation and computation effort. In this case, we used a simplified approach using transition probabilities to switch between actions and define different events and time frames per person, e.g., working hours.

2.5. Data Extraction and Analysis

To validate the results, we analyzed the artifacts on the mobile devices. First, we extracted the information using physical or logical extraction (depending on the model) using MSAB XRY 9.6.1. The Android extraction options "Extract additional metadata" and "App-Downgrade" for all the apps were active during logical extraction. Afterwards, we performed the analysis using MSAB XAMN 6.2 and compared automatically generated data to the analysis results of manually created samples.

Another important aspect we had to consider was the analysis of covered personal information of the scientists or unwanted location data. When manually populating devices, it is challenging to prevent such information from appearing on the device, e.g., carrying around the device and forgetting to turn off WiFi at home may expose private addresses. Thus, we explicitly searched for traces that did not belong to the case scenario.

2.6. Software

The source-code as well as the documentation of the framework is provided in an open-source repository via a GitHub repository belonging to FORMOBILE (https://github.com/FORMOBILE, (accessed on 2 March 2022)).

3. Results

This section introduces the context leading to AutoPoD-Mobile and presents more detailed information on the implementation. We used it to generate artifacts on multiple devices according to a fictive case scenario. We outline the case scenario and results from the experiment.

At the beginning of the FORMOBILE project in 2020, our team manually created a first ring trial to challenge forensic experts and assess the current state of mobile forensics. Aiming for a realistic case scenario, we populated seven sets of three devices, each with innocent background data and case-relevant information. In the end, we internally required several people to help us, carrying the mobile devices and sending messages at an exact point in time from a given location, resulting in very labor-intensive work and mobile devices with less irrelevant data than we planned. The amount of data was not sufficient to cover all forensic questions. The relevant information of the case scenario was easy to find. It did not challenge forensic experts compared to their cases within their Law Enforcement Agencies (LEAs). From this experience, we concluded that manually setting up multiple devices in a realistic way for training, testing tools and challenges in ring trials is not a suitable option.

After researching state-of-the-art methods to generate mobile forensic training datasets and reviewing the available training materials, we started working on small scripts to automate repetitive tasks derived from the first ring trial generation process. In the end, we refactored many components. We started combining the modules into AutoPoD-Mobile as a framework for us to prepare the mobile devices for a second ring trial in the FORMOBILE project in 2021. This enabled us to place far more background data as irrelevant noise and address forensic challenges, such as dealing with big data on modern mobile devices and identifying and assessing relevant information.

Before we review the details of the sample case, we will cover the different types of information placed on the mobile devices in the experiment. Table 4 provides the different types of data and information on the implementation and resulting artifacts. We chose these types based on typical smartphone usage of a real person, covering contact information, calls, messages, photos, emails, calendar entries, browser usage, third party applications, and position information.

Table 4. Data types generated in this study.

Data Type	Main Scripts	Tools Needed	Data Input	Data Output	Result
Contacts	`contacts_main.py`; `send_contacts.py`; `v_card.py`; `delete_contact.py`	contact app on the device, selenium, ADB	vCard file	vCard 2.1; vCard 3.0	Contacts in the contact app
Calls	`phonecall_main.py`	SIM card, ADB	Phone number	Outgoing phone call	Phone call in call log
WhatsApp	`whatsapp_FSM.py`; `main_FSM`; `whatsapp_setup.py`	WhatsApp Web Client, SIM card	Conversation dataset	Text messages, voice messages, pictures	Chats and group chats
Pictures	`main_pictures.py`; `remove_metadata.py`	ADB	Picture dataset	Renamed pictures	Pictures in the camera roll with corrected timestamp
Apps	`apps_main.py`; `install_apks.py`	ADB	APK file	Apps	Installed apps
Browser	`chrome_main.py`; `populate.py`	ADB	URL list	-	Browser history
Location	`spoofing_main.py`	Appium Settings app, Google Maps app	Location file	-	Logged location in Google Maps
Email	`gmail_main.py`; `setup_mail.py`; `email_FSM.py`	Google account	Conversation dataset	-	Emails
Calendar	`calendar_main.py`; `cal_setup.py`	Google account	Event file	-	Calendar Events

Different input and output formats might be available with each base class of information. This covers different image formats, such as JPEG or RAW images or multiple versions of the vCard format. After reviewing the fictive case used to test and validate our approach, we will cover these implementation aspects.

3.1. Case Scenario Definition

The case scenario defines the role of different persons, behaving in a defined way depending on their personality and life situation. They act as part of the main plot leading to a criminal case and the subsequent investigation of the mobile devices in their possession. As we think of the scenario as a play, we assume the role of the stage director.

As a first step, we defined the type of crime in the scenario. In this case, we chose to include conversations and other artifacts relating to drug and arms trafficking, as well as homicide. We refer to the individuals as persons and provide fictional information, such as name, surname, birthday, home address, and work address in the scenario.

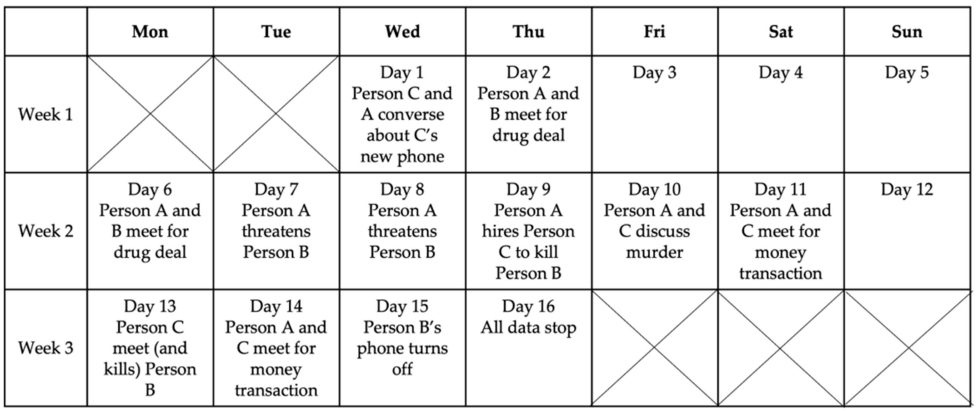
In addition, all the persons receive individual Gmail accounts. Some of them were also equipped with a PCloud account. Hobbies and interests, contacts and calendar events, GPS data for the course of the case, subscribed Telegram channels, visited websites, and some scripted conversations between each other, i.e., to provide background information/evidence, were also added. Details of the data generated per person can be seen in Table 5.

Table 5. Data generated for each person in the case scenario.

Data Type	Person A	Person B	Person C
Device	Alcatel 1SE	Samsung A40	XGody S20 Lite
SIM card	O2	Telekom	Vodafone
Occupation	Drug dealer	Store employee	Arms dealer
Interests	drugs, money, cars	gambling, marihuana, dealing with debts	arms, cryptocurrencies
Accounts	Google	Google	Google, PCloud
WiFi connections	17	13	5
Call log entries	0	>15	0
Newsletters subscribed	>20	>20	>20
URL list	>50	>50	>50
Contacts	>20	>20	>20
Locations	8	8	8
Location changes	>30	>25	>25
Subscribed Telegram channels	>20	>20	>20
WhatsApp group chats	>50	>50	>50
Active time	10:00–03:00 (17 h)	07:00–23:00 (16 h)	10:00–01:00 (15 h)
Calendar events	>10	>10	>10
Apps	>10	>10	>10
Pictures	on device >21,000	on device >9000	on Pcloud: >8000
Videos	>5	>5	>5

In our experiment, the runtime of the case scenario covered 16 days in total and was deliberately kept simple. It contains only three main actors: Person A is a drug dealer selling drugs to Person B on two occasions. They exchange messages, plan the transactions, and physically meet. Nevertheless, Person B refrains from paying for the drugs and is threatened by Person A, who then decides to hire his contact—Person C—to kill Person B when the threats seem ineffective as a motivation to pay the debts. Person C is an acquaintance of Person A, is involved in arms trafficking, and finally commits the murder of Person B.

Figure 1 provides an overview and the timeline of the case-related events and interactions between the different persons. Aside from the physical meetings between Person A and Person B on days 2 and 6, another physical meeting takes place: on day 13, Person C visits the workplace of Person B, intending to (and succeeding to) murder Person B. Person A and C also meet before and after the murder, on day 11 and 13, where Person A pays Person C for the murder. After the murder, all forms of active communication/interactions on Person's B phone stop. Finally, the phone runs out of battery the next day, on day 15.



	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Week 1			Day 1 Person C and A converse about C's new phone	Day 2 Person A and B meet for drug deal	Day 3	Day 4	Day 5
Week 2	Day 6 Person A and B meet for drug deal	Day 7 Person A threatens Person B	Day 8 Person A threatens Person B	Day 9 Person A hires Person C to kill Person B	Day 10 Person A and C discuss murder	Day 11 Person A and C meet for money transaction	Day 12
Week 3	Day 13 Person C meet (and kills) Person B	Day 14 Person A and C meet for money transaction	Day 15 Person B's phone turns off	Day 16 All data stop			

Figure 1. Timeline and relevant events in the case scenario.

3.2. *Setup of the Automated Data Population Process*

This part describes the setup for our automated data population, which combines the developed software framework and the physical devices. The process focuses on a deterministic way of creating multiple (almost) identical sets containing several mobile devices in parallel.

To understand the need for multiple sets, we recall the initial goal—enabling us to create one set of devices, which seem consistent within a given fictional case during the preparation phase of a European ring trial. As the methods used for the forensic extraction are also part of the ring trial (compared to sending out forensic images to analyze), we need to provide multiple (close to identical) sets.

In our case, we prepared six sets, each consisting of three devices, as each of the three persons A, B, and C uses exactly one device, as shown in Table 3. All of the 18 devices are grouped and denoted as Ai, Bi, and Ci with i ∈ {1, 2, . . . , 6} as the index of the set, e.g., device B4 refers to the Samsung A40 smartphone in the fourth set.

Figure 2 shows the setup of the early-stage experiment in the lab: for each person in the case scenario, we used a single computer, connecting all of the devices within the respective set via a (powered) USB hub. This created a clean physical setup and distinction between the sets, making device management very simple in practice. In addition, we used a dedicated computer controlling the finite state machine (FSM). This device featured a USB sound card, which we use to generate voice messages in WhatsApp using a playback loop to simulate the recording process.

Please note that at this time, the FSM did control only email and WhatsApp correspondence for all persons using online resources (web applications/cloud services). The idea was that we could use the dedicated machine to orchestrate the overall scenario. During the further development and the addition of device-specific modules for browser activity, photo/gallery, and location spoofing, the computer controlling each person's devices now also runs the FSM directly.

All devices were powered on for the full 16 days of the case scenario experiment to allow the device interactions. While some scripts initialize and prepare the devices and are thus executed only once initially, other modules require a constant connection between the computer and the mobile device, e.g., to simulate browsing patterns.

Switching from the hardware to the software setup, Figure 3 represents the high-level architecture of the different scripts. To fulfill our role as stage director, we need to provide the base configuration (configuration file), defining device- and person-specific information used within the scenario. The configuration file defines personal traits for each person regarding probabilities for each state transition, active/inactive time, and typing/reading speed. The configuration file also includes the likelihood of sending audio messages with whom a person communicates, group chats participation, and number of pictures and applications on the device.

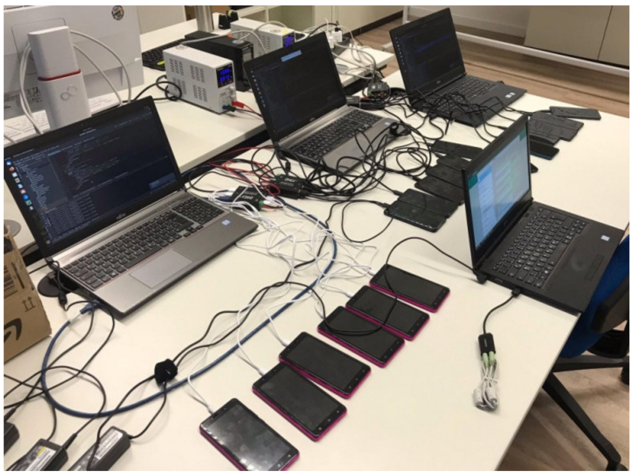


Figure 2. Case scenario hardware setup in the first experiment. Each device set (A, B, and C) consists of six devices (1–6) and each device set is controlled by a dedicated laptop (from left to right: Person C, XGody S20 Lite; Person A, Alcatel 1SE; and Person B, Samsung Galaxy A40). One additional laptop (in front) controls the FSM and generates voice messages. The dedicated power supply (in the back) powers the USB hubs and thus charges the devices.

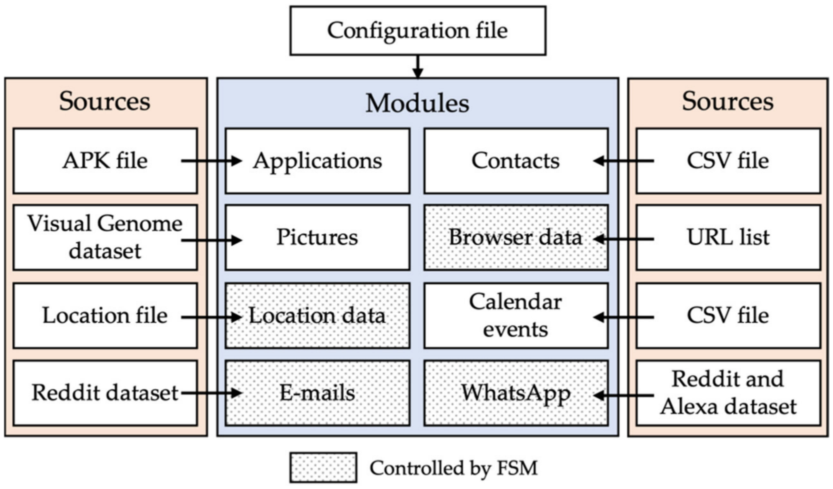


Figure 3. Schematic representation of the Python scripts created for the automated data population process. Some modules are controlled by a finite state machine (FSM; dotted background).

We created a single module for each data type included in the scenario, which manages the interaction with the device(s) or specific applications. They require a data source, which in our experiment may be a publicly available dataset, pre-generated files to deploy, or tables to look up. Now we had to combine all the information and run the scenario. Therefore, an FSM script orchestrates the timing, threading of processes, and probabilities of the different actions within our fictive case.

Currently, the scripts focus on Android devices and were tested using the following models: Alcatel 1SE, Samsung A40, Samsung A50, Samsung S9, Huawei Mate 20 Lite, and XGody Mate Lite 20. This provided many insights into the difficulties of automating across multiple vendors and Android versions. Looking at iOS, we integrated basic iOS support and tested it using an iPhone X. The iOS scripts fully support WhatsApp, emails, and calendar events, while contacts, pictures, browser data, location data, and custom applications are partially supported.

Setting up custom applications: This module aims to control custom applications installed on the mobile device, which can be used to initialize the device based on a certain personality or install an application within the scenario, e.g., installing a messenger app after receiving an SMS talking about it.

The application script uses APK files of various applications from the APK mirror (see https://www.apkmirror.com (accessed on 24 January 2022))—representing a free mirror outside Google Play—and installs the files through the ADB interface on mobile devices. As an example, we installed Zoom, Disney+, Instagram, Delivery Heroes, Reddit, Signal Messenger, Snapchat, Microsoft Teams, Twitter, eBay, Telegram, WhatsApp, etc. The user can define a set of apps shared between all devices and applications specific to some persons. This is useful when modeling the different persons: a teenager will have different applications installed compared to a middle-aged person.

Setting up the phone contact list: this module feeds the contact application with contact information, including fields such as the first name, last name, phone number (work or private), email address, address, birthday, and picture.

On the Android models, contacts can be transferred to the mobile device as vCard 2.1 using ADB. A CSV file defines these values and can be changed according to the requirements of the case scenario. The data are subsequently converted to the vCard format. As iOS devices do not support a service similar to ADB, we send the vCard via email to the device. Opening the vCard will import the contacts into the contact application of the iOS device.

Simulating "taking photos": the picture script intends to simulate camera usage and fills the gallery application of a mobile device with images and corrected metadata.

A study in 2019 [25] showed that Android users store on average 952 pictures on their mobile devices. This number varies depending on certain aspects, such as age, gender, or country of residence. We used the Stanford Visual Genome dataset for our experiment, though images corresponding to the fictive case can be used later. First, we divided the dataset into multiple distinct subsets so that different devices would have different pictures. We attributed random timestamps to the picture, covering the time between the provided start and end date specified in the configuration file. As the script takes care to refrain from modifying the timestamp during transfer to the mobile device, this leads to a continuous and chronological stream of pictures on the mobile device.

Simulating surfing the web: analyzing the browser history (especially search history) is crucial for mobile forensics. The Browser module simulates different web activities. The script opens a web browser and different websites to create the respective artefacts on the device. These sites can be defined in an input file specific to each person in the case scenario. Furthermore, the browsing frequency and randomization of the URL may be changed. This script will also take WhatsApp messages as input and look up words present in the message using a Google search.

Providing location data: as the devices do not physically leave the lab, this module uses a straightforward method of storing location information on the device. The location data script will provide location data spoofed with an application (e.g., Appium Settings) that is opened regularly and receives longitude and latitude data as input, controlled by a configuration file. It may contain different locations so that the fictive location is automatically updated at different defined points in time to match the case scenario.

Setting updates and reminders: calendar apps are available on all smartphones, but they vary as each vendor may supply their app and store events differently (locally or remotely using different cloud services). Using a vendor-independent method that all (Android) phones support, this module interacts with the Google Calendar, setting up events or series of events.

The calendar events script uses the Google account and Google Developer Console. As a preparation step, we need to enable specific settings in each person's Google account manually: allowing third-party apps and granting permission to see, edit, and share calendars. The calendar module requires an input file that contains the different events to be placed on the mobile device. The events must contain a start and end time/date and additional information, such as the title, location, or invitees. They can also be part of an event series.

Corresponding via emails: while most daily conversations shifted from emails to instant messengers, business conversations often rely on this medium. As with the calendar events, the mail application varies between different vendors, and we chose the vendor-independent method using Google Mail. The email module uses the Google account and similar settings required by the calendar module. As we need to produce larger text fragments, we use anonymized messages from a sub-thread of the Reddit platform as an input dataset. As in our experiment, all the people use Google Mail as an email platform. They can read and reply to messages automatically via the FSM.

Participating in conversations: instant messaging is crucial for daily mobile device usage. We focused on WhatsApp messages representing instant message services, as they generate more data traffic today than SMS messages [26]. This module enables messaging between two or more persons and creates a massive amount of communication data.

The WhatsApp script sends different messages, i.e., text, pictures, documents, and voice messages, either in direct communication between two persons or within a group chat. In terms of automation, we automatically select different input datasets and utilize the WhatsApp Web client as a simple method to support cross-platform usage, and thus support Android and iOS devices independent of the make and model.

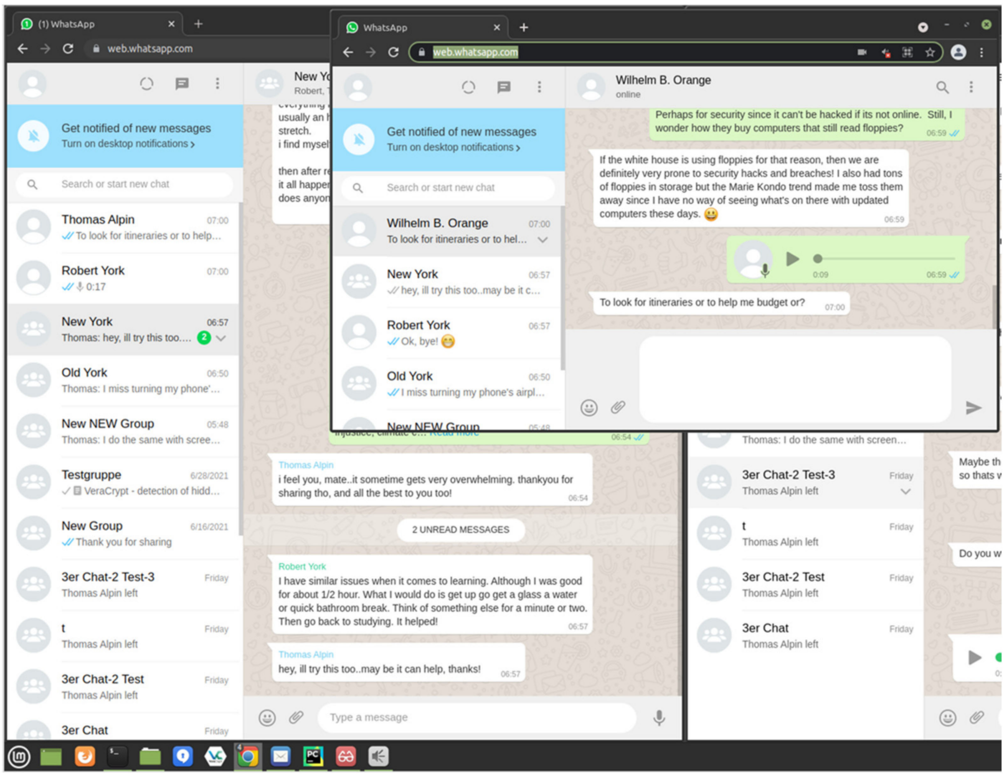
To set up and initialize the module, we install the WhatsApp application on the mobile device. Afterwards, we connect it to the internet and link the device to the WhatsApp Web Client in a browser manually. The Web client creates a QR code, which the mobile device scans; this process generates a security token stored for all further sessions. From this point on, sending and receiving messages is automated, requires no interaction, and does not need visible Web clients anymore.

Please note that the WhatsApp Web client receives continuous updates, which may break the automation. To ease the potential maintenance requirements, we implemented an observer mode: all the WhatsApp Web clients are visible when observing. The interaction between the clients can be watched and potential problems can be traced. Figure 4 shows the observer mode for three clients with text and voice messages.

The WhatsApp script requires the conversations as input datasets. We used different datasets depending on the number of people within a conversation and added support for emojis. The artificial audio messages are currently generated using text-to-speech conversion.

Case-unrelated background noise: in a realistic setting, either all contacts belong to a case setting, i.e., when analyzing particular "business" or burner phones, or a small group of contacts is case relevant. To increase the number of unrelated contacts generating only case-irrelevant communication, we use the devices from the other device sets.

As each person and the respective device was duplicated six times, i.e., the devices A_1 to A_6 all belong to Person A. We can use these devices as case-irrelevant contacts. The forensic analyst in the ring trial does not receive the other five devices. Figure 5 illustrates the communication paths for the datasets index $i = 1$. Please note that all devices use active SIM cards and are connected to networks and the internet. This allows us to use the technique with all communication channels.



(1) WhatsApp

web.whatsapp.com

Get notified of new messages
Turn on desktop notifications ›

Search or start new chat

Thomas Alpin 07:00
To look for itineraries or to help...

Robert York 07:00
0:17

New York 06:57
Thomas: hey, ill try this too.... 2

Old York 06:50
Thomas: I miss turning my phone'...

New NEW Group 05:48
Thomas: I do the same with scree...

Testgruppe 6/28/2021
VeraCrypt - detection of hidd...

New Group 6/16/2021
Thank you for sharing

3er Chat-2 Test-3 Friday
Thomas Alpin left

t Friday
Thomas Alpin left

3er Chat-2 Test Friday
Thomas Alpin left

3er Chat Friday

New Y
Robert, T

usually an h
stretch.
i find mysel

then after re
it all happen
does anyon

Thomas Alpin
i feel you, mate..it sometime gets very overwhelming. thankyou for sharing tho, and all the best to you too! 06:54

2 UNREAD MESSAGES

Robert York
I have similar issues when it comes to learning. Although I was good for about 1/2 hour. What I would do is get up go get a glass a water or quick bathroom break. Think of something else for a minute or two. Then go back to studying. It helped! 06:57

Thomas Alpin
hey, ill try this too..may be it can help, thanks! 06:57

Type a message

WhatsApp

web.whatsapp.com

Get notified of new messages
Turn on desktop notifications ›

Search or start new chat

Wilhelm B. Orange 07:00
To look for itineraries or to hel...

New York 06:57
hey, ill try this too..may be it c...

Robert York 06:57
Ok, bye! 😁

Old York 06:50
I miss turning my phone's airpl...

New NEW Group 05:48

Wilhelm B. Orange
online

Perhaps for security since it can't be hacked if its not online. Still, I wonder how they buy computers that still read floppies? 06:59

If the white house is using floppies for that reason, then we are definitely very prone to security hacks and breaches! I also had tons of floppies in storage but the Marie Kondo trend made me toss them away since I have no way of seeing what's on there with updated computers these days. 😃 06:59

0:09 06:59

To look for itineraries or to help me budget or? 07:00

Thomas: I do the same with screen...

3er Chat-2 Test-3 Friday
Thomas Alpin left

t Friday
Thomas Alpin left

3er Chat-2 Test Friday
Thomas Alpin left

3er Chat Friday
Thomas Alpin left

Maybe th
so thats v

Do you w

Figure 4. Screenshot of the WhatsApp module in observer mode using three WhatsApp Web clients on the workstation. The windows represent different persons who independently write messages to each other. At the time of the screenshot, the simulated person in the middle window types a message, whereas another person simulated just received an audio message.

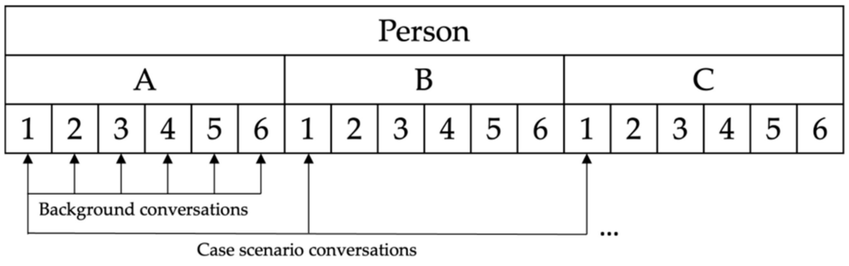


Figure 5. Schematic representation of the conversation matrix between persons, exemplary for Person A in device set 1 (Device A_1). The six different device sets (1–6) were used to generate background conversation for each person. Conversation between A, B, and C were only permitted within the same device set number, e.g., A_1 may converse with B_1 but not with B_2.

3.3. Modeling Realistic Behavior

The significant advantage of an automated approach for data population is the vast amount of information placed on the devices. Compared to the manual approach, where every message is entered and sent by hand, automation reduces the time and effort, allowing complex and realistic mobile forensics datasets.

A straightforward implementation is using the different modules independently, e.g., by first setting up new contacts, then running the WhatsApp module for several hours, and finally switching to e-mail conversations manually. Unfortunately, this does not create realistic multi-party interactions.

In practice, mobile device usage is highly dynamic, as the user receives outside triggers, i.e., a “new message” indicator pops up and can instantly switch apps back and forth. With this in mind, we designed an FSM based on a Markov Chain Model, where transitions are probability-driven. A similar approach has been implemented in the 3LSPG tool [17].

Figure 6 shows an instance of the FSM that controls the WhatsApp, Browser data, email, location data, and picture modules. It starts in the `idle` state on the far left side. This is the state where the person's mobile device is unlocked, but yet inactive. In this instance, the person opens the WhatsApp application with a probability of 60%, moving to the state `whatsapp_idle`. From this point, with a probability of 70%, the person starts reading unread messages, changing to `whatsapp_read_message`.

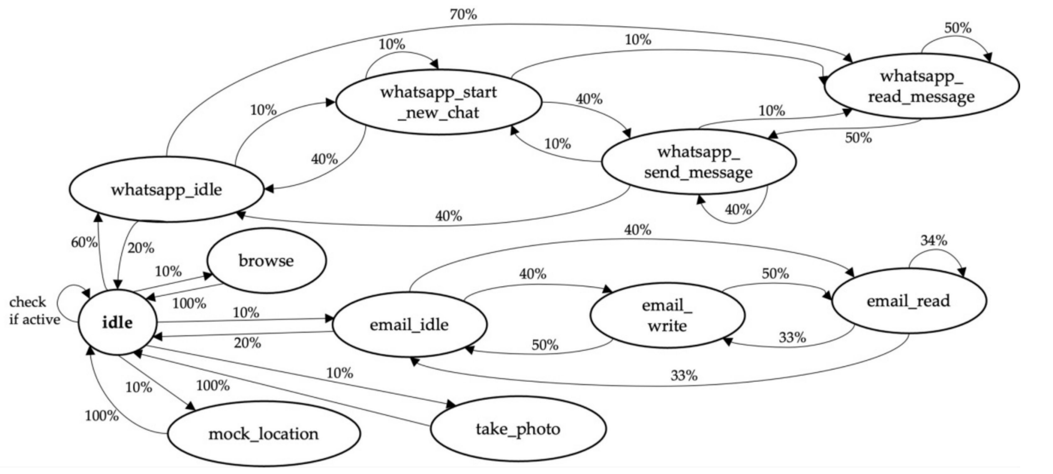


Figure 6. FSM defining the different states of WhatsApp, browsing, emails, pictures, and GPS spoofing. Each circle represents a state, each arrow represents the transition to another state and each percentage represents the probability with which the transition occurs.

As the FSM models the habits of each person, the transition probabilities are unique for each person. Modeling the case scenario leads to multiple independent FSMs in parallel with different active states at any given point in time, which creates a more realistic chain of events and covers "offline" behavior, such as sleeping or working hours.

Adjusting the probabilities for each person helps reflecting on different personality traits. For example, one person might rarely write emails, another person might read messages on WhatsApp, but rarely reply. The current FSM may be extended to suit other scenarios and it is possible to add more states representing other actions, e.g., initiating, accepting, or rejecting phone calls, or opening a social media app to create more complete and realistic datasets in the future.

3.4. Experimental Conclusions

The current state of the scripts allows for the successful creation of training datasets, trying to mimic the behavior of different persons using mobile devices. It was developed using Python 3.8 and can be extended in the future.

We used the current setup for 16 days to create six sets of three mobile devices each, containing artificially generated data in a sufficient amount to be realistic in a training or ring trial. A more than two week time frame allowed us to create enough data and scatter "relevant evidence" of our case scenario over multiple days.

We validated the results by extracting and analyzing the generated data and distributed the remaining device sets as a still-ongoing ring trial within the EU H2020 project FORMOBILE. Our analysis of the automated data population process showed that most of the generated data could not be distinguished from manually generated data directly on the device. The only exception is that WhatsApp messages are tagged in the forensic tool as emanating from a PC client instead of a mobile device as we use the WhatsApp Web client. Please note that the client's IP is not reported and does not reveal that the data were synthetically generated compared to the normal use of the Web Client to converse on WhatsApp.

4. Discussion

Mobile forensics is a modern and fast-paced field of digital forensics. We face yearly new major OS releases with new features and security changes, monthly releases of new mobile devices with significant hardware and firmware differences, and daily releases of new third-party apps or updates to them. Every change creates a direct impact for the forensic experts, e.g., requiring an adapted forensic data extraction process, evaluating the app support of existing forensic tools, or requiring a new tool for the analysis steps.

To address the impact on mobile forensic analysts and ensure high quality in their analyses, regular training, test devices, and ring trials are mandatory to keep the experts on the verge of technical advances. Therefore, access to complex and realistic datasets is already relevant and will further increase in the future.

Please note that the available methods for forensic extraction in mobile forensics depend on many factors, such as the device make and model, the firmware itself, and thus the operating system version and security patch level.

For example, even if the same information were present on two devices, the challenges of extracting the forensic data may lead to a different depth of the analysis and the results, i.e., when comparing unencrypted physical images with a logical image from a running device. Keeping this in mind, creating training images and sharing them neglects a crucial part of the expert training: handling the mobile device itself.

AutoPoD-Mobile was designed to address this need and help with the automated creation of artificial (yet realistic) data on mobile devices, following a fictive case scenario. We successfully used it to create multiple sets of devices. As we still consider it a proof of concept, we will use the remainder of this section to discuss our findings, potential pitfalls, mismatches between artificial and genuine data, and open questions for future work.

An important aspect when generating training data on mobile devices is the process of cloning such devices. Once a case scenario involving multiple mobile devices has been created, including the relevant as well as non-case related data, multiple copies of these sets may be required: not only for ring trials, but also to share the sets among the trainees in a mobile forensics training. Depending on the operating system and device, cloning mechanisms may work. Nevertheless, this process usually requires unlocking the bootloader and custom recovery or boot images on the Android devices. As such modifications are not typically encountered in a real case (or are the minority), it is currently impossible to clone devices reliably for training purposes. The extent of such modifications and means to hide it for training purposes should be investigated in future work, including using the OS backup solutions, e.g., Google One, and the impact on the quality of the remaining artifacts on the clone.

If we think further about the requirements for mobile devices, SIM cards are essential to consider already when preparing the devices. In our manual creation process, we used SIM cards from the same network provider and used SIM cards with consecutive IDs. As the careful analysis of the hardware and the SIM card is central in an investigation, we need to take care of such anomalies (unless we explicitly need the analyst to conclude our case scenario).

A registered SIM card is usually mandatory for all mobile devices for calling or sending text messages and when generating instant messaging data. We did not consider the case of WiFi-only tablets using a workaround to register non-supporting messenger apps and did not store additional data on the SIM card itself.

This should be considered in the future, together with an evaluation of potential pitfalls using eSIMs or dual SIM devices.

Another pitfall of the setup to keep in mind concerns the smartphones' battery. In our setup, we kept all devices charged all the time. We noticed that this is an anomaly, as the battery monitor data does not match a natural person's suggested location changes and normal behavior. For future experiments, we require a hardware module that simulates the different persons' charging behavior.

Before discussing the different modules of the software implementation and potential improvements, we need to point out that we focused on Android devices. This means that functions developed for iOS are fundamental in the current setup and all modules relying on ADB for interaction with Android require an iOS alternative. For example, when importing contacts in iOS, we send vCard per email to the device to test the import functionality for very basic functionality. This will most likely create email fragments on the iOS device and thus should be changed in the future.

Please note that all framework modules based on an external API or web client are already usable with iOS. While this provides great flexibility and cross-platform portability, it also comes with a drawback: These modules need maintenance and updates in case of significant API or GUI changes.

As we considered the apps for the case scenario, we decided not to use third-party apps that replace standard applications. An example would be the calendar application, where hundreds of replacements exist in different app stores. As our approach relies on simulating the regular use of an app, we need to analyze each application manually before supporting it, as we may require a reliable way to generate, transfer, and store information. Please note that this approach is more rewarding than inserting artificial data into SQL databases. Using the app itself may leave different artifacts in the file system (even in unrelated parts due to the OS interaction) present in a real device.

We tried to use standard formats to supply information whenever possible. Still, the format or version of the standard differed in some cases between Android and iOS or even between different models of Android devices. An example is contact information, which is usually imported and exported using the vCard format. There exist different versions, e.g., 2.1 or 3.0, and some devices may require a specific version and, thus, further adjustments in the future.

While the data on the device passed our validation checks, we identified many ways to improve the artificially generated data on the devices. This may include pre-or post-processing or different approaches to work on.

Enhancing (meta)data of pictures and video: in the current approach, we remove all the metadata present in the picture file and adjust the critical information about the date, time, and the file's name. This leads to consistent display in the gallery apps and the mobile forensic software suites. To create more realistic artificial data, the camera (app) of the target device could be automatically analyzed beforehand, creating a profile for the camera/device combination. Using this profile, we can correct the image itself and the metadata, e.g., resolution, camera name, aperture, exposure time, focal length, ISO, and add location data and significantly increase the credibility of the image and video data.

Application management: as of now, applications are installed by the application module using APK files, and we used this feature to initialize the devices. During the runtime of the case scenario, more applications should be downloaded using the Google PlayStore or alternative stores, such as F-Droid, and uninstalled if necessary. This requires UI automation and frequent updates if the store applications change their interface, but it would allow a more realistic simulation. For example, a person might receive a message inviting the person to use Signal instead of WhatsApp to continue the conversations and asking the person to delete WhatsApp for security reasons.

Working with location information: nowadays, location information, especially position tracking, is omnipresent in modern devices. This information contains important clues for forensic analysts in the investigations. While spoofing the location information according to the case scenario is mandatory to make the dataset realistic, this is non-trivial (as the devices never leave the lab).

Currently, our location module uses the Appium Settings application that should update the location information in the mobile device. With our first setup, the forensic analysis showed that the presence of the location data was device-dependent and not always available. The workaround we exploit in the current setup consists of regularly opening the Google Maps application to log the location information there, as most forensic tools report this information.

Improved (context-based) messaging: the current setup uses WhatsApp's instant messaging platform. The chats are created and follow the correct conversation path. The module sends messages within those chats and can keep track of all chats simultaneously for an unlimited amount of time, usually the duration of the experiment, to generate as much background noise as possible. While useful, the behavior should be further refined: some chats could run during the first days, others may be kept back and start later and finish early. This also removes the anomaly that a person is simultaneously active in 50 or more chats.

We implemented the group chats to keep to a certain topic by using dedicated input datasets (based on subreddits about the topic). In the future, the input should change from fixed datasets to on-the-fly creation using open-source artificial intelligence, such as the language models GPT-2 [11] or GPT-3 [12]. Such a change is non-trivial but will add more flexibility in the end.

Text to speech: to create basic support for voice messages, we chose a Python library using text-to-speech. One of the reasons was to preserve the person's privacy setting up the experiments. While this worked fine, it still leaves room for improvements, such as realistic text-to-speech conversion or different voice profiles (one per person in the scenario).

Adding more messaging services/breaking up communications: finally, we currently support WhatsApp as an example of a messaging application. Many different messengers are frequently used, such as Signal, Threema, or Telegram. Commonly, mobile device users install and use more than one application—sometimes within one conversation, switching channels as they see fit. Such breaks in a conversation by switching communication channels is particularly challenging for LEAs. We included such a switch between SMS, email, and WhatsApp in our case scenario. Still, adding support for more communication channels is important.

Generation Footprint considerations: in this project, our goal was to keep the noticeable footprint of the population automation process as low as possible. Regarding location data, instead of using the Appium Settings application, we could also use a framework requiring root permissions and install a location spoofing service. Still, this leaves more traces on the device and implies that the device was (potentially) artificially populated.

We may need to think about shifting the balance in favor of functionality and accept a higher Generation Footprint in the future. With such a change, we could implement a population service, which automates the full simulation at exactly the correct time and performs many actions without delay. This allows full control over the artifacts created on the device—in case of root permissions—and we need to investigate the limitations of a user-land process depending on the different systems.

Aside from considering features to enhance the framework, the generation of realistic mobile forensic training datasets still has many open research questions, out of which we would like to highlight two:

Translating behavior profiles to simulations: the FSM framework used to simulate the behavior is a first step toward realistic device usage simulation. We limited ourselves to certain basic device features we estimated the different persons might use, as adding all of the features of modern smartphones manually is close to impossible.

Still, we need much more research on how to create realistic behavior artificially and derive the device features from the behavior. Therefore, we need to link together different disciplines. We start with language and behavior profiling to generate realistic conversations and general behavior. We further need to consider criminal profiling for realistic case-related behavior and computer science or—in some cases, specialized topics, such as game design—for models derivation and the automated creation of complete control FSMs to automate the device usage.

Dynamic movement simulating: currently, location changes are defined manually as a location at a single point in time. A much more advanced method includes the complete simulation of movement patterns for each person depending on the habits, personal and business contacts, calendar events, and relevant aspects of the case scenario. In such a simulation, the current location per person should be dynamically computed and embedded into all related artifacts. As this type of simulation has a major impact on the overall simulation, this is non-trivial for future research.

5. Conclusions

Datasets play an essential role in educational and academic work in mobile forensics. However, these groups often do not have access to use datasets from real criminal cases. As shown, privacy issues and legal problems limit the usage and distribution. In this contribution, we first review available digital forensic datasets as well as forensics frameworks. Based on this survey, we introduce a methodology and a ready-to-use framework to populate smartphones with datasets in a semi-automated way. The proposed solution is designed to address this need. Following a fictive case scenario, we can create artificial (yet realistic) data on mobile devices. To our knowledge, there is no comparable solution for the field of mobile forensics so far. All scripts discussed in this contribution have been published as open-source. Hence, other researchers can directly use the results presented in this article to generate realistic datasets more quickly in the future.

With the solution described, we can already create datasets close to reality. Many aspects are automated and show great potential for the mobile forensics community. In a practical experiment, we evaluated the framework, successfully preparing a ring trial with six sets of three devices within the FORMOBILE EU Project. We discussed the results and possible improvements and outlined topics for future work to modulate, complement, and enhance the presented work.

Author Contributions: Conceptualization, M.M. and R.Z.; methodology, M.M. and R.Z.; software, M.M. and R.Z.; validation, M.M., D.P. and R.Z.; formal analysis, M.M.; investigation, M.M.; resources, M.M. and R.Z.; data curation, M.M.; writing—original draft preparation, M.M., D.P. and R.Z.; writing—review and editing, M.M., D.P. and R.Z.; visualization, M.M.; supervision, R.Z.; project administration, R.Z.; funding acquisition, D.P. and R.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the European Union's Horizon 2020—Research and Innovation Framework Programme, H2020-SU-SEC-2018, under grant agreement no. 832800.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors thank the European Union's Horizon 2020—Research and Innovation Framework Programme for funding this research. The authors also acknowledge the important help of Alexander Raßhofer (HM Hochschule München University of Applied Sciences) and Dennis Tóth (Deggendorf Institute of Technology) to this project.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Pawlaszczyk, D. Mobile Forensics—The End of a Golden Age? *J. Forensic Sci. Crim. Investig.* **2022**, *15*, 1–4. [CrossRef]
2. Montasari, R.; Hill, R. Next-Generation Digital Forensics: Challenges and Future Paradigms. In Proceedings of the 2019 IEEE 12th International Conference on Global Security, Safety and Sustainability (ICGS3), London, UK, 16–18 January 2019; pp. 205–212.
3. Humphries, G.; Nordvik, R.; Manifavas, H.; Cobley, P.; Sorell, M. Law enforcement educational challenges for mobile forensics. *Forensic Sci. Int. Digit. Investig.* **2021**, *38*, 301129. [CrossRef]
4. Grajeda, C.; Breitinger, F.; Baggili, I. Availability of datasets for digital forensics—And what is missing. *Digit. Investig.* **2017**, *22*, S94–S105. [CrossRef]
5. Horsman, G.; Lyle, J.R. Dataset construction challenges for digital forensics. *Forensic Sci. Int. Digit. Investig.* **2021**, *38*, 301264. [CrossRef]

6. Jahankhani, H.; Hosseinian-far, A. Chapter 8—Digital Forensics Education, Training and Awareness. In *Cyber Crime and Cyber Terrorism Investigator's Handbook*; Akhgar, B., Staniforth, A., Bosco, F., Eds.; Syngress: Rockland, MA, USA, 2014; pp. 91–100. ISBN 9780128007433. [CrossRef]
7. McCullough, S.; Abudu, S.; Onwubuariri, E.; Baggil, I. Another brick in the wall: An exploratory analysis of digital forensics programs in the United States. *Forensic Sci. Int. Digit. Investig.* **2021**, *37*, 301187. [CrossRef]
8. Horsman, G. Tool testing and reliability issues in the field of digital forensics. *Digit. Investig.* **2019**, *28*, 163–175. [CrossRef]
9. National Institut of Standards and Technology. *Mobile Device Data Population Setup Guide*; Version 2.0; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2016.
10. Ayers, R.; Livelsberger, B.; Guttman, B. *Quick Start Guide for Populating Mobile Test Devices*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2018; pp. 1–43, NIST SP 800-202.
11. Yannikos, Y.; Graner, L.; Steinebach, M.; Winter, C. Data Corpora for Digital Forensics Education and Research. In *Proceedings of the Advances in Digital Forensics X. DigitalForensics 2014. IFIP Advances in Information and Communication Technology, Vienna, Austria, 8–10 January 2014*; Peterson, G., Shenoi, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2014; pp. 309–325.
12. Garfinkel, S.; Farrell, P.; Roussev, V.; Dinolt, G. Bringing Science to Digital Forensics with Standardized Forensic Corpora. *Digit. Investig.* **2009**, *6*, S2–S11. [CrossRef]
13. Brueckner, S.; Guaspari, D.; Adelstein, F.; Weeks, J. Automated computer forensics training in a virtualized environment. *Digit. Investig.* **2008**, *5*, S105–S111. [CrossRef]
14. Du, X.; Hargreaves, C.; Sheppard, J.; Scanlon, M. TraceGen: User Activity Emulation for Digital Forensic Test Image Generation. *Forensic Sci. Int. Digit. Investig.* **2021**, *38*, 301133. [CrossRef]
15. Göbel, T.; Schäfer, T.; Hachenberger, J.; Türr, J.; Baier, H. A Novel Approach for Generating Synthetic Datasets for Digital Forensics. In Proceedings of the Advances in Digital Forensics XVI. DigitalForensics 2020. IFIP Advances in Information and Communication Technology, New Delhi, India, 6–8 January 2020; Peterson, G., Shenoi, S., Eds.; Springer International Publishing: Cham, Switerland, 2020; pp. 73–93.
16. Scanlon, M.; Du, X.; Lillis, D. EviPlant: An Efficient Digital Forensic Challenge Creation, Manipulation and Distribution Solution. *Digit. Investig.* **2017**, *20*, S29–S36. [CrossRef]
17. Yannikos, Y.; Franke, F.; Winter, C.; Schneider, M. 3LSPG: Forensic Tool Evaluation by Three Layer Stochastic Process-Based Generation of Data. In Proceedings of the Computational Forensics. IWCF 2010. Lecture Notes in Computer Science, Tokyo, Japan, 11–12 November 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 200–211.
18. Hummert, C.; Pawlaszczyk, D. *Mobile Forensics—Common File Formats and File Systems Used in Mobile Devices. The File Format Handbook*; Springer: Berlin/Heidelberg, Germany, 2022. [CrossRef]
19. Mobile Operating System Market Share Europ, October 2020. Available online: https://gs.statcounter.com/os-market-share/mobile/europ,%20October%202020 (accessed on 24 January 2022).
20. Mobile Vendor Market Share Europe. Available online: https://gs.statcounter.com/vendor-market-share/mobile/europe (accessed on 24 January 2022).
21. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language Models Are Unsupervised Multitask Learners. *OpenAI Blog* **2019**, *1*, 9.
22. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models Are Few-Shot Learners. *arXiv* **2020**, arXiv:200514165 Cs.
23. Gopalakrishnan, K.; Hedayatnia, B.; Chen, Q.; Gottardi, A.; Kwatra, S.; Venkatesh, A.; Gabriel, R.; Hakkani-Tür, D. Topical-Chat: Towards Knowledge-Grounded Open-Domain Conversations. In Proceedings of the Interspeech 2019, Graz, Austria, 15–19 September 2019; pp. 1891–1895.
24. Krishna, R.; Zhu, Y.; Groth, O.; Johnson, J.; Hata, K.; Kravitz, J.; Chen, S.; Kalantidis, Y.; Li, L.-J.; Shamma, D.A.; et al. Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations. *Int. J. Comput. Vis.* **2017**, *123*, 32–73. [CrossRef]
25. Snap Happy: Avast Reveals Which Countries Store the Most Photos. Available online: https://blog.avast.com/which-countries-store-the-most-photos (accessed on 24 January 2022).
26. SMS und WhatsApp—Nachrichten pro Tag Deutschland bis 2015. Available online: https://de.statista.com/statistik/daten/studie/3624/umfrage/entwicklung-der-anzahl-gesendeter-sms--mms-nachrichten-seit-1999/ (accessed on 24 January 2022).