

# Univerza v Ljubljani Fakulteta za računalništvo in informatiko

Izzivi

Študij

Ustvarjanje novih svetov

Raziskovanje

FRI

# UNIVERSITY OF LJUBLJANA FACULTY OF COMPUTER AND INFORMATION SCIENCE

Zbornik

# DIGITALNA FORENZIKA Seminarske naloge, 2023/2024

Ljubljana, 2024

# Zbornik

Digitalna forenzika, Seminarske naloge 2023/2024

Editors: Andrej Brodnik, Nika Molan, Nikolina Petrovikj, študenti

Template author: David Klemenc

Ljubljana : Univerza v Ljubljani, Fakulteta za računalništvo in informatiko 2024.

©These proceedings are for internal purposes and under copyright of University of Ljubljana, Faculty of Computer and Information Science. Any redistribution of the contents in any form is prohibited. All rights reserved.

# Kazalo / Contents

1	Uvod / Introduction	<b>2</b>
2	Povzetki / Summaries         2.1       Come fire come snow, these scientists will get in your phone	$\begin{array}{c} 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 4 \\ 4 \\ 4 \\ 4$
3	Mobilna forenzika / Mobile Forensics         3.1       Come fire come snow these scientists will get in your phone         3.2       Smartphone Local Backup         3.3       Mobile Forensics	<b>7</b> 7 11 15
4	<ul> <li>Dnevniki in forenzika / Logs and Forensics</li> <li>4.1 Every step you take, I'll be tracking you: Forensic analysis of the tile tracker application</li></ul>	<b>21</b> 21 26 32
5	<ul> <li>Forenzika gesel in šifriranje / Password Forensics and Cyphering</li> <li>5.1 Forenzična analiza dvofaktorske avtentikacije v nekaterih aplikacijah</li></ul>	<b>38</b> 38 44 51
6	Forenzika brskalnikov / Browser Forensics6.1Forensic Browser Memory Analysis6.2Cross platform cloud data access through browser credential migration	<b>58</b> 58 61
7	Forenzična preiskava / Forensic Investigation         7.1 Smernice za forenzično preiskavo avtomobilskih kamer         7.2 A Bayesian evaluation of captured device location         7.3 Can ChatGPT Unlock Potential in Digital Forensics?         7.4 Evidence in the fog – Triage in fog computing systems	<b>67</b> 67 73 79 86

# 1 Uvod / Introduction

Digital forensics is a pivotal branch of forensic science dedicated to the recovery and investigation of materials found in digital devices. Originally synonymous with computer forensics, the field of digital forensics has expanded significantly to encompass the examination of all digital data storage devices. Its roots can be traced back to the personal computer revolution of the late 1970s and early 1980s, evolving through the 1990s without formal organization until the emergence of national guidelines in the 21st century.

Digital forensic investigations serve diverse purposes, primarily to substantiate or challenge hypotheses in both criminal and civil court proceedings. In criminal cases, these investigations support allegations of offenses such as murder, theft, and personal assault. In civil cases, they safeguard individuals' rights and property, often addressing disputes ranging from family matters to contractual conflicts among economic entities. The field also extends into electronic discovery, crucial in modern litigation.

This conference gathers the seminar papers of master's students at the Faculty of Computer Science and Information, University of Ljubljana 2023/2024. It explores a variety of topics including mobile forensics, logs, password forensics and cyphering, browser forensics and forensic investigation. Each paper contributes to the ongoing development and application of digital forensic techniques, enhancing our ability to navigate the complexities of today's digital landscape.

In mobile forensics, three seminar papers provide valuable insights. The first paper addresses data recovery from damaged devices, highlighting challenges in accessing data without owner-provided access. The second paper evaluates smartphone backup integrity, focusing on iPhone and Android systems. The third paper examines MOBILedit Forensic Express for retrieving digital evidence, emphasizing complexities in accessing messaging app data amid evolving security measures.

In logs and forensics, seminar papers cover varied topics. One explores forensic analysis of the Tile tracker app, stressing tools like Tile Artifact Parser (TAP) for geolocation data and spoofing detection. Another examines Windows 10 and 11 logging, extracting diagnostic data (USB, Wi-Fi, apps) with advanced methods. Lastly, a paper addresses data recovery challenges, leveraging MSSQL transaction log structure knowledge and memory analysis to identify and partially restore deleted entries, proving effective in database forensic investigations.

In password forensics and cyphering, one paper examines two-factor authentication (2FA), specifically Time-Based One-Time Passwords (TOTP). It reviews studies on secret key discovery, secure 2FA implementations, and analyzes different 2FA applications. Another paper focuses on decrypting TPM-protected BitLocker volumes, critical for digital forensics. It explores methods like Cold Boot attacks, TPM sniffing, DMA bug exploitation, and processor debugging to recover encryption keys. The third paper enhances password cracking with context-based dictionaries optimized by NLP and sociological insights, highlighting their efficacy over generic approaches.

In browser forensics, research explores two key areas: memory analysis for recovering browsing history despite private mode limitations, and methods for cross-platform access to cloud-stored data through browser credentials. These studies address critical challenges in digital investigations related to web browser artifacts and cloud data security.

In forensic investigation, seminar papers cover several key areas. One focuses on guidelines for analyzing automotive cameras like dashcams, highlighting their role in legal and insurance contexts for traffic incidents. Another introduces a Bayesian approach to enhance the accuracy of device geolocation data evaluation. Additionally, research explores the potential of ChatGPT in digital forensics, examining its applications and limitations. Lastly, a paper discusses strategies for handling evidence in fog computing systems, addressing challenges in data access across dynamic network structures.

# 2 Povzetki / Summaries

## 2.1 Come fire come snow, these scientists will get in your phone

With mobile devices playing a crucial part in evidence collecting, the ability to fix damaged phones shows its importance. This report first explains why such practices are so important. We then delve into different types of damages, from water to impact and bio-hazardous damage. Based on the literature found, these damages are either described in general or through specific cases that forensic scientists worked on. We also explain common problems when dealing with damaged devices and solutions to these. In further chapters we then talk about possible processes, which can fix the device or at least make it possible to extract the device data. Again, we mention benefits and dangers of such procedures. Lastly, we briefly describe possible work, when the owner of the phone doesn't provide their information, such as passwords or other encryption keys. We mention what extracted data from device might look. Finally, we mention other methods, which were ommited in the essay and other possible types of phone damages.

Ključne besede / Keywords: Digital forensics, Data recovery

### 2.2 Smartphone Local Backup

In this paper, we first present the general blueprint described in [7] for evaluating the data integrity of mobile device local backup mechanisms. Then we provide a general overview of smartphone backup mechanisms with an emphasis on iPhone and Android mechanisms. Lastly, we provide a more detailed description of the general process on an iPhone 7 mobile device and a Samsung Galaxy S8 mobile device, along with a brief analysis of the acquired data.

### 2.3 Mobile Forensics

This study explores mobile forensics and focuses on MOBILedit Forensic Express's (MFE) effectiveness in locating digital evidence on cellphones, especially in Western messaging applications. We evaluate the advantages, disadvantages, and moral implications of MFE through empirical study. We also discuss the differences between government-only and commercial technologies and argue for fair use of forensic capabilities. Our results show how difficult it is to get around device security and how mobile forensics is developing. Through our analysis of messaging app data accessibility, we clarify the intricacies of this subject. In the end, this study aims to advance knowledge about how mobile forensics can fight cybercrime while protecting people's right to privacy.

## 2.4 Every step you take, I'll be tracking you: Forensic analysis of the tile tracker application

The surge in popularity of personal Bluetooth trackers has sparked a demand for forensic tools that assist law enforcement in retrieving artifacts. With reported sales of 40 million Tile devices to date, Tile trackers stand as one of the most favored personal Bluetooth trackers. However, this growth has brought about challenges, as incidents of malicious use of Bluetooth trackers have also risen. We present a paper that examines the Tile ecosystem and its associated application across iOS, Android, and Windows platforms. The data extracted from the application contains geolocation coordinates spanning the previous 30 days. The authors present an open-source tool, named Tile Artifact Parser (TAP), capable of parsing these forensic artifacts from the Tile application and detecting potentially spoofed data.

Ključne besede / Keywords: Tile, Memory forensics, Application forensics, Bluetooth tracker

## 2.5 Analysis of Windows 10 & 11 Logging Mechanisms for Use in Digital Forensics

Due to Windows' high market share [5], forensic analysis of Windows 10 and 11 is a high-value target for researchers. In this paper we review existing methods for analysing Windows' diagnostic data, explain the structure behind the EventTranscript.db database, and re-implement our own version of Park and Lee's DiagAnalyzer tool [2] in order to obtain valuable forensic evidence from a snapshot of a Windows machine. Alongside Park et al.'s research into USB storage, Wi-Fi connection and Edge browser usage events, our paper proposes two new types of diagnostic data that is of high forensic value: laptop battery level, and detailed application activity.

Ključne besede / Keywords: Digital Forensics, Logging, Diagnostic Data, Windows, Application Activity

# 2.6 Forenzična analiza transakcijskih dnevniških zapisov SQL strežnika na nedodeljenem delu datotečnega sistema

Podatkovne baze so ključni del računalniških sistemov, ki hranijo pomembne informacije o poslovnih okoljih. Njihova uporaba se povečuje, s tem pa tudi število incidentov kot so poškodba podatkov, vdori in nenamerno razkritje podatkov. Ker so danes informacije bolj pomembne kot strojna oprema, je digitalna forenzika ključnega pomena. Za obnovo izgubljenih podatkov so pomembni predvsem dnevniški zapisi o transakcijah. Ti zapisi se običajno shranjujejo v dnevniških datotekah transakcij, ki se jih redno varnostno kopira (kopije se periodično brišejo zaradi prihranka prostora). Obravnava se problem pridobivanja izbrisanih dnevniških zapisov iz nedodeljenega pomnilnika. Z uporabo znanja o strukturi dnevnika transakcij MSSQL ter analize nedodeljenega pomnilnika je mogoče identificirati in delno obnoviti izbrisane zapise. Eksperimenti na izmišljenih podatkih in realnih podatkih so pokazali uspešnost pristopa, ki se je izkazal kot obetaven za forenzične preiskave na področju podatkovnih baz.

Ključne besede / Keywords: Digitalna forenzika, transakcijski dnevniški zapisi SQL, nedodeljeni del datotečnega sistema, rekonstrukcija podatkov

## 2.7 Forenzična analiza dvofaktorske avtentikacije v nekaterih aplikacijah

V seminarski nalogi predstavimo koncept dvofaktorske avtentikacije in njeno uporabo. Naštejemo različne načine njene implementacije in se poglobimo v dvofaktorsko avtentikacijo (2FA) s TOTP. Opišemo rezultate predhodnih člankov na temo iskanja tajnih ključev, varnejše implementacije 2FA in analize različnih 2FA aplikacij. Osredotočimo se na delo v članku o analizi 2FA aplikacij. Predstavimo njihove rezultate in sklepe. Več različnih 2FA aplikacij ima enak algoritem za izračun TOTP, kar je potencialno koristno za delo digitalnih forenzikov. Na koncu izvedemo še lasten test obhoda TOTP in dokumentiramo svoja opažanja. Ugotovimo, da se ujemajo z opažanji izhodiščnega članka.

Ključne besede / Keywords: Dvofaktorska avtentikacija, digitalna forenzika, artefakti, OTP, tajni ključi

## 2.8 Decrypting TPM-Protected BitLocker Volumes

TPM-protected BitLocker volumes can pose significant challenges in digital forensic investigations. This seminal work researches different methods for encryption key recovery including Cold Boot attacks, TPM sniffing, DMA bug exploitation, and processor debugging. Our focus is primarily on the replication and evaluation of the TPM sniffing method.

Ključne besede / Keywords: Trusted Platform Module, TPM, Cold boot attack, TPM sniffing, Encryption key recovery, BitLocker

# 2.9 Enhanced Password Cracking: Optimizing Context-Based Dictionaries with NLP and Sociological Insights

The increasing frequency of password usage in the digital age demands advanced techniques for password cracking. In this paper, we present sociological insights into password creation and common password patterns while considering the vulnerabilities commonly exploited by attackers. We discuss a few known password cracking techniques such as bruteforce attacks, Rainbow tables and Markov-based models, with a special focus on password cracking by using contextual dictionaries, which is presented in the main article. We present tools like DBpedia and natural language processing to explain how authors made and improved special contextual dictionaries that match certain sets of information. Their methodology involved the extraction of thematic concepts from Wikipedia articles and the ranking of keywords based on their thematic distance from seed words. The results of their research are presented, highlighting the performance of customized contextual dictionaries compared to generic dictionaries like Ignis-10M.

Ključne besede / Keywords: Password cracking, contextual dictionary, natural language processing

## 2.10 Forensic Browser Memory Analysis

Internet browsing activity generates vital evidence for forensic investigations. The browsing history is particularly significant in this context. However, private browsing modes prevent the creation of cache and history records, which complicates finding evidence substantially. This report reviews methods used in forensic analysis to address this challenge.

Ključne besede / Keywords: Digital forensics, Volatile data, Memory forensics, Web browser, User activity, Physical memory

## 2.11 Cross-platform cloud data access through browser credential

Browser credential migration has become a pivotal focus in digital forensic investigations, driven by the widespread adoption of cloud services and the critical role of web browsers in accessing sensitive information. This article offers a comprehensive examination of browser data migration methodologies, challenges, and practical solutions for transferring user information across different operating systems.

Ključne besede / Keywords: Browser, Cookies, Firefox, Chrome, Chromium, DPAPI, keychain

# 2.12 Smernice za forenzično preiskavo avtomobilskih kamer

V sodobnem prometnem okolju se vse pogosteje srečujemo z različnimi vrstami kamer, ki so postale ključni del nadzornih sistemov na cestah. Te kamere niso samo pomemben element za vzdrževanje reda in zagotavljanje varnosti na cestah, temveč postajajo tudi nepogrešljiv del forenzičnih preiskav. Namen te seminarske naloge je osvetliti različne vidike uporabe avtomobilskih kamer, zlasti tistih, ki so nameščene v vozilih (t.i. dashcams), in raziskati, kako lahko te naprave služijo kot ključni viri dokazov v pravnih in zavarovalniških primerih.

Ključne besede / Keywords: Smernice za preiskavo, forenzična preiskava avtokamer, prometne nesreče, dashcam

# 2.13 A Bayesian evaluation of captured device location

This paper presents an improved procedure for assessing geolocation evidence quality in digital forensics. Building on previous research, we introduce a probabilistic approach that considers directional errors in geolocation data. By using prior knowledge on GPS error margins, our method offers a more detailed evaluation framework. We outline the problem, our procedure, how we use data, and compare our findings. Our study improves the accuracy and reliability of geolocation evidence evaluation, pushing forward digital forensic investigations.

Ključne besede / Keywords: Device geolocation, MCMC, digital forensics

# 2.14 Can ChatGPT Unlock Potential in Digital Forensics?

In this report we build upon the research presented in the paper "ChatGPT for Digital Forensic Investigation: The Good, the Bad, and the Unknown." [6] We start by exploring related work in digital forensic investigation. Then, we dive into a detailed examination of the original paper, focusing on its experiments and findings. We also add our own examples as validation of the original findings. Our goal is to understand the strengths, weaknesses, and areas yet to be explored in using ChatGPT for digital forensic tasks.

Ključne besede / Keywords: ChatGPT, GPT-4, digital forensics, Artificial Intelligence, Large Language Models

## 2.15 Evidence in the fog – Triage in fog computing systems

With the increase use of cloud services the need for fog networks as an intermediate layer grows. At the same time the number of crimes involving at least one digital device is also growing. This paper will present how to approach (digital) evidence in the fog. The most important part is understanding how to network operates, and where to look for data - with some probability. There are quite some metrics we need to take into account if we want to give a good estimation, but we can never be 100% certain to say which node will have the data, because the structure of fog networks is very dynamic. We will also compare some measures that can be used for estimating the probability of a node containing the data.

Ključne besede / Keywords: Fog computing systems, triage, forensic, evidence, data volatility, YAFS, dynamic networks, analytical model



3 - Mobilna forenzika / Mobile Forensics

# Come fire come snow, these scientists will get in your phone

Matek, Veronika Fakulteta za računalništvo in informatiko Večna pot 113 Ljubljana, Slovenia vm0012@student.uni-lj.si

#### ABSTRACT

With mobile devices playing a crucial part in evidence collecting, the ability to fix damaged phones shows its importance. This report first explains why such practices are so important. We then delve into different types of damages, from water to impact and bio-hazardous damage. Based on the literature found, these damages are either described in general or through specific cases that forensic scientists worked on. We also explain common problems when dealing with damaged devices and solutions to these. In further chapters we then talk about possible processes, which can fix the device or at least make it possible to extract the device data. Again, we mention benefits and dangers of such procedures. Lastly, we briefly describe possible work, when the owner of the phone doesn't provide their information, such as passwords or other encryption keys. We mention what extracted data from device might look. Finally, we mention other methods, which were ommitted in the essay and other possible types of phone damages.

#### **General Terms**

Digital forensics, Data recovery

#### Keywords

Digital forensics, Phone damage, Phone recovery, Data preservation

#### 1. INTRODUCTION

The mobile device is in digital forensics becoming a crucial concern. The share of mobile devices keeps increasing, edging out the traditional computer systems, which were once the go-to devices for work and entertainment. With most people in first world countries having their phone on their body at all times, a variety of information is recorded, some of which might play vital roles in court, when trying to come to an accurate verdict. It is now, more than ever, important for digital forensics scientists to be able to extract data from these devices. Klančar, Matej Fakulteta za računalništvo in informatiko Večna pot 113 Ljubljana, Slovenia mk2529@student.uni-lj.si

Sadly, small devices, which are taken everywhere, have a much higher chance of being damaged than traditional computer systems. Additionally, wrongdoers might have incentive to try and destroy their device and the information stored on it. Thus, digital forensics scientists have been working hard on methods, which try to save as much evidence from possible phones as possible. As will be seen, their work is impressive, with devices one might call a *lost cause* successfully fixed and data extracted.

# TYPES OF DAMAGE Water damage

One way of destroying a mobile device includes water damage which causes elecrochemical reactions within the device causing metal corrosion. The damage done to the device is affected by multiple factors including but not limited to time of exposure, battery life and salt content.

We differentiate between three different types of corrosions:

- electrochemical migration,
- galvanic corrosion,
- conductive anodic filament.

Electrochemical migration (ECM) is caused because of existing potential bias between electrodes that are closer and closer due to the constant shrinking of electronic components. Potential bias that can be almost 20 V can be as close as 0.4 mm apart. By introducing water, the potential bias drives electrons from the anode to the cathode. The deposited metals start to grow toward the anode and upon touching it creates a short curcuit.

The speed of ECM depends on the voltage and distance of components, as well as the chemicals in the water. Chlorides and salt speed up this process.

Galvanic corrosion can occur with no voltage needed as the water that is introduced in the equation serves as a channel for electron movement. To maintain electron neutrality, electrons flow from less noble metals (like aluminum) to noble metals (like copper). This happens because noble metals have higher electrode potentials.

Conductive anodic filament (CAF) is the hardest to identify as the are no visual signs of it. The reason for this is because corrosion happens between internal layers inside the printed curcuit board (PCB) and the components on the surface. Similar to ECM, once deposited salts reach the cathode a short curcuit is created.

Fukami and Nishimura [5] tested the affect of 3 day water submersion of 2 different brands of phones - Samsung and LG. For each brand they submerged a turned off and an operating device and tested the damage the water made after completely drying the phones after submersion.

Between Samsung and LG it was noticed that different types of contaminants were deposited on parts. Samsung having mostly copper, nickel and tin which are used for PCBs and solder which makes us assume that elecrochemical reactions happened on PCB. Differently the LG deviced had mostly aluminum contaminants which we can conclude was present because of galvanic corrosion. All these deposits were removed before attempting to be connected to a power source.

Both Samsung devices had severe damage to parts directly connected to battery, but the turned on device had more deposits. Neither of the devices turned on after being connected to a power sources. They were able to repair both devices by replacing the parts impossible to repair and reballing the solder lost in the submersion process.

LG devices turned on, the turned off device even booting and operating normally. The reason why LG devices were less damaged than Samsung ones can be accounted to power managment chips being protected by an epoxy glue named underfill. This protected the solder balls from the water. The other device did not display anything, the cause for which was corrosion on pins of the display. It turned on as well after repairing the connections between the display and the PCB.

The damage done by water depends on many factors. One of them is how much charge the battery has left which fuels the corrosion process. Having the device on also causes power supply to the components that need it. This is why it is essential the battery is removed as fast as possible when trying to successfully extract data from a submerged device.

#### 2.2 Impact damage

Impact damage is another common approach criminals take to destroying their device. This type is especially attractive to wrongdoers who wish to get rid of their data in a hurry. The article [7] describes a case where the owner of a Samsung phone threw it against the wall in order to destroy evidence. This is just one example of a possible impact damage, with other popular methods such as destroying devices using tools e.g. hammers also being noted.

Article reports the team thoroughly examined the phone, first simply by observing it and then examining it with a microscope. The forensic team was not able to extract data directly from memory as it was encrypted. When all damaged parts were replaced the device still wouldn't boot, thus forcing the forensics team to look closely at the printed circuit board, as they suspected a short circuit might be to blame. They confirmed their theory by checking multi-meter readings and then detecting the short circuit spot by creating a layer of soldering paste through which a current was passed. The short circuited spot had blackened. When the screen for the device was replaced, the device successfully turned on.

While this is just one example of how forensics team can access data of impact damaged phones and the challenges they encounter, such processes are common. In the extreme cases, cause of damage might be gunshots. Even then, depending on the location of the impact, restoration might be possible.

Authors of [2] report phones with larger batteries have better chance of being recoverable, due to the chance of bullet only hitting the battery increasing. Such phones typically have smaller logic boards, which, when damaged, quickly decrease the possibility of data recovery. Thus, shots aimed at the center of the phone have smaller odds of ruining evidence than if the bullet hit the phone in the outer regions. Of course, ammunition type plays a huge role as well.

#### 2.3 Bio-hazardous Damage

Digital forensics scientist Steve Watson reported in his presentation [9] his experience from cases he worked on, where the device was covered in bio-hazardous material, more accurately human blood. His testing consisted of covering various devices in blood, soaking them for different periods of times, from 24 hours, to a week, months, and a year, as can be seen in figure 1. He reports of a good success rate of recoverable data, being able to access it in each case so far. Only circuit boards were soaked in blood, as other parts of the phone could be replaced if needed, thus their recoverability not being of crucial concern. One of the devices had a failure of the digitizer, that is, the part of the mobile phone which converts touch into digital signal. The part could be replaced and the data was still recoverable.

Device Number	Manufacturer	Model Name	Model Number	Duration	Retrieval Date
1	LG	G Stylo	MS631	Control	2/27/2019
2	LG	Leon LTE	LG-M5345	24 hours	3/1/2019
3	Samsung	Galaxy on5	SM-G550TI	1 week	3/7/2019
4	LG	G4	L5991	1 month	3/28/2019
5	LG	Volt 2	L5751	3 month	5/28/2019
6	Samsung	Galaxy on5	SM-G550TI	6 month	8/28/2019
7	LG	Tribute HD	LS676	9 month	11/28/2019
8	Samsung	Galaxy Avant	G386TI	12 month	3/1/2020

Figure 1: Table showcasing different device models and blood soaking times in the study [9].

Article [4] reports of dangers to the forensic examiners themselves, as bio-hazardous material could infect the person studying the device, if proper precautions are not taken. Several viruses and bacteria are able to survive without a host for several years. For example the bacteria Salmonella typhimurium has a lifespan of up to several years. Common solutions, like bleach, are in this case unusable, as they pose a threat do the device. Article [4] proposes the use of ultrasonic cleaning, which uses sound waves. Sound waves cause cavitation, a process which creates micro-sized vacuum bubbles, which, when imploded, can destroy both bacteria and



Figure 2: Before and after of a phone, covered in blood for 48 hours, then cleaned with the ultrasonic cleaner and Conflikt, a disinfectant spray produced by Decon Labs [4].

viruses. An example of such cleaning with combination of a disinfectant spray Conflikt can be seen on Figure 2.

Wrong use of such cleaners can cause damage to fragile wire bonds. Frequencies of 35 kilohertz or higher must be used, while still omitting "hot spot" frequencies, which cause resonance in the device materials. A restriction of the cleaning cycle times is also put in place.

#### 2.4 Microwave damage

When a phone is put in a microwave, it looks a lot like fire damage from the outside, as the heat causes deformation of the plastic enclosure that a phone can have. A korean study by Song, Sa, Nam, Kim, Choi and Oh [6] tested the effects of 2.45GHz microwave radiation of an older flip phone using a 700W microwave. They tested fully charged phones that were turned off and on for different periods of time.

The noticed that 10 to 20 seconds of microwaving did almost no damage in terms of deformation, as well as no big temperature change was recorded. More damage was recorded when the exposure time was more than 30 seconds. They noted severe damage to the folder hinge and the antenna, but this is due to this parts of the phone being metal. The reason for this is because metal concentrates the electromagnetic waves of the microwave and creates localized heat which causes the damage. In these localized places the temperature rose as high as 260°C. The internal damage can be seen in Figure 3.

They also noted that the internal components itself were not damaged, regardless of the phone being turned off or not. The batteries were okay, as long as the exposure was not longer than 60 seconds. There were also no noted defects to the inner board. The damage can occur to internal components, that are connected to metal parts where there is localized heat. The difference with fire damage is that the



Figure 3: Internal damage of a flip phone after 50 seconds of microwave exposure [6].

battery swells and can lead to a fire.

#### **3. FIXING THE DAMAGE**

Exposing the phone to different types of damages which causes damage to any of its components can prevent its functioning. This damage can stop at circuit damage but be so severe it causes a fire or an explosion. How we reconstruct the device depends on many factors, but there are standard procedures in forensic laboratories on how to deal with this [7].

All devices are tagged for identification upon arrival. The damage is inspected with a microscope and assessed. Any sediments that can or had caused short circuits are removed, which in cases of light damage can restore the device. When it comes to worse damage, we must perform chip transplantations or chip-off recovery. This in the case of transplantation means that we must own other healthy components we can use to replace the damaged ones in hopes of restoring the device [5, 7].

On the other hand when dealing with chip-off recovery, individual chips from the damaged device are moved to a working logic board. All of the chips, created by machines, need to be individually desoldered by humans. The chip pins have to be cleaned of the solder before the chip can be carefully moved to the donor board or a chip reader [1]. This is a very delicate process as the chips can easily overheat or break, losing the valuable data on them, which can be of key importance to an investigation. This process is getting harder and harder as the components are getting smaller. [2]. The needed heat also poses a danger to the chip itself, as it can damage the chip, thus ruining the wanted evidence. Article [1] also warns of legislations requiring the return of the device in its original state, which this method doesn't allow.

An alternative to the desoldering method is the so called *hot air* method, which melts the edges of the TSOP chip, making it possible to remove the chip using a vacuum air gripper. Pins still have to be to be cleaned of the solder, but the amount of solder is reduced in comparison to the conventional method [3].

Other times the expert also need to do processes such as resoldering the solder lost in the damage process or repairing any lost wiring and retracing the connections between components [5].

# 4. EXTRACTING THE DATA

A lot of times criminals do not want to give up their information on how to get into the phone to law enforcement. In some cases the data can be accessed regardless if the device has USB Debugging enabled. Unfortunately for law enforcement this is disabled by default. When this is enabled the pin code or pattern can be decoded using information from the gesture.key file which we can find in /data/system [2].

The US court requires of some phone companies, like Apple, to help the law enforcement with accessing the data on a locked device on the court's subpoena, but the newer phone updates no longer allow this [2].

Included in re	port	Total	
50	-	50	
200		200	
2291	(311 Deleted)	2291	(311 Deleted
2620	(125 Deleted)	2620	(126 Deleted
588	(98 Deleted)	588	(98 Deleted)
588	(98 Deleted)	588	(98 Deleted)
52		52	
52		52	
1520	(27 Deleted)	1520	(27 Deleted)
1520	(27 Deleted)	1520	(27 Deleted)
459	(1 Deleted)	459	(1 Deleted)
459	(1 Deleted)	459	(1 Deleted)
14619	(96 Deleted)	14619	(96 Deleted)
5021	(5 Deleted)	5021	(5 Deleted)
5		5	
3		3	
2		2	
5		5	
9		9	
309	(1 Deleted)	309	(1 Deleted)
471		471	
574	(74 Deleted)	574	(74 Deleted)
393	(1 Deleted)	393	(1 Deleted)
4		4	
458		458	
42		42	
32		32	
	Roubert in           60           20           2261           200           200           300           300           301           400           401           401           402           403           403           404           405           407           408           409           401           502           6           7           8           9           9           301           401           502           603           604           704           705           704           705           704           705           704           705           704           705           705           706           707           708           709           700           700           701           702	Databel Insport           50           200           201         (11 Desker)           202         (20 Desker)           50         (20 Desker)           50         (20 Desker)           50         (20 Desker)           52         (20 Desker)           53         (20 Desker)           40         (1 Desker)           409         (1 Desker)           54         (2 Desker)           55         (2 Desker)           5         (2 Desker)           5         (2 Desker)           5         (2 Desker)           5         (2 Desker)           6         (2 Desker)           70         (1 Desker)           40         (1 Desker)           41         (2 Desker)           53         (1 Desker)           44         (4 Desker)           45         (1 Desker)           46         42	Induct inspir         1124           60         00           200         201           200         011 Collegit           500         020           500         020           500         020           500         020 Collegit           501         020 Collegit           502         020 Collegit           403         (1 Collegit)           403         (1 Collegit)           501         500 Collegit           502         20           503         20           504         500 Collegit           505         300 Collegit           6         9           9         0           702         (1 Collegit)           703         (1 Collegit)           704         (1 Collegit)           704         (1 Collegit)           703         (1 Collegit)           704         (1 Collegit)           703

# Figure 4: Excerpt of Mobile Device in Report from Physical Analyzer [2].

In some cases people of course willingly give up their code pins or patterns to law enforcement. Special devices are then used to extract data, such as Universal Forensic Extraction Device Touch-2 (UFED). The extracted data contains information such as call logs, messages, images, documents, the browsing history, application data from social media account, etc. All of this information is provided to the authorities to be used in the given case. An excerpt of this information can be seen in Figure 4 [2].

#### 5. CONCLUSION

As was shown through the essay, the ability to fix damaged devices is quite remarkable. In our research, we were surprised time and time again with all cases which were salvageable through right steps. This holds for both the presented cases and the ones omitted, such as thermal or heat damage. The biggest challenge reported here is the ability to open the device up [8]. The other extreme, phones being frozen, again leaves the possibility of recovery quite high [9].

We hope the essay showed how ingenuity of forensic scientist is an important personality trait in their line of work, and creates another layer of appreciation for their everyday work.

#### 6. **REFERENCES**

- K. A. Alghafli, A. Jones, and T. A. Martin. Forensics data acquisition methods for mobile phones. In 2012 International Conference for Internet Technology and Secured Transactions, pages 265–269, 2012.
- [2] D. Blackman. Mobile device damage and the challenges to the modern investigator. 2015.
- [3] M. Breeuwsma, M. Jongh, C. Klaver, R. Knijff, and M. Roeloffs. Forensic data recovery from flash memory. *Small Scale Digital Device Forensics Journal*, 1, 01 2007.
- [4] K. Dudeck, T. Brennan, and D. Embury. Decontamination of blood soaked electronic devices using ultrasonic technology. *Forensic science international*, 214:88–95, 08 2011.
- [5] A. Fukami and K. Nishimura. Forensic analysis of water damaged mobile devices. *Digital Investigation*, 29:S71–S79, 2019.
- [6] S. Jae-Yong, S. Seung-Hun, N. Jung-Woo, and O. Bu-Yeol. Analysis on damage patterns of a folder type mobile phone caused by microwave-irradiation. *Journal of Korean Institute of Fire Science and Engineering*, 26(2):11–16, 2012.
- [7] A. Kumar, B. Ghode, K. Maniar, and D. Jain. Forensic analysis of broken and damaged mobile phone - a crime case study. *International Journal of Scientific Research* in Computer Science, Engineering and Information Technology, pages 481–487, 06 2021.
- [8] S. Watson. Damaged device forensics. Digital Forensics Research Conference, 2018.
- [9] S. Watson. Extracting evidence from damaged devices. Forensic Focus, 08 2021.

# **Smartphone Local Backup**

Jan Novak Domijan Don Mohar UL FRI

#### ABSTRACT

In this paper, we first present the general blueprint described in [7] for evaluating the data integrity of mobile device local backup mechanisms. Then we provide a general overview of smartphone backup mechanisms with an emphasis on iPhone and Android mechanisms. Lastly, we provide a more detailed description of the general process on an iPhone 7 mobile device and a Samsung Galaxy S8 mobile device, along with a brief analysis of the acquired data.

#### **1** INTRODUCTION

In the realm of digital forensics, the integrity of data acquisition methods is critical in proving the authenticity of digital evidence. Despite the push from both Apple and Google towards remote backup mechanisms, local backup mechanisms are still used extensively for data acquisition on mobile devices and should be scrutinized to provide validity to such data acquisition methods.

To evaluate data integrity it is imperative to first have an overview of different local backup mechanism as some considerations must be made with regards to the quantity and quality of data included in backups. It is also crucial that there exists a way of obtaining reference data that is as close to the original as possible. The data integrity model presented gives us a better understanding of how much data is changed during backup and which data is changed. This is done by analysing the data acquired for mismatch between the data collected before the backup, during the backup and after the backup. Of particular interest are the files that are present before and during the backup but their value differs.

#### 2 LITERATURE REVIEW

Upon review we did not find any recent work directly addressing local backup integrity. The lack of literature on the subject can be explained by a shift towards remote backups as the mechanism of choice. Both Google and Apple encourage users to use Google Drive and iCloud for their backup needs. While Apple still actively supports local backup through iTunes, ADB backup has been marked as deprecated and is no longer actively supported [5].

We were able to find information on how local backup mechanisms can be used to obtain data but it was scattered across various blogs and forums and was in certain cases heavily device and firmware specific.

There are also some vendor-specific data migration services and backup mechanisms for Android, such as Samsung Switch, XiaoMi Mi Mover and Huawei Phone Clone. For a comprehensive review of data acquisition using vendor-specific solutions consult [6].

#### 3 BACKUP MECHANISM OVERVIEW

In [7] the authors gave a categorization of backup methods in regards to:

- Location of backup, local or remote.
- Scope of backup, full or selective.
- Type of backup, file-based or content-based.

#### 3.1 Local and Remote Backups

Both Android and iOS offer options for local and remote backups. For remote backups iOS uses iCloud, whereas Android uses Google Drive by default. For local backups iOS uses iTunes and Android uses ADB. Local backups are valuable in forensics as they provide a quick and non-destructive way to obtain at least some data for off-device analysis.

#### 3.2 Full and Selective Backups

The authors consider a backup that only contains some applications as a selective backup and a backup that contains all applications as a full backup. They further give the ADB full backup as an example of a full backup mechanism and Apple iTunes backup as an example of a selective backup.

We find this distinction between a full backup and a selective backup misleading, as the ADB full backup is intended for debugging applications and does not necessarily backup all application data. Additionally, deciding which parts of the system software could be considered an application may not be straightforward.

A related concept to the full backup is the concept of a full system image, which is the basis for the general data integrity evaluation model. A full system image is a copy of all directories of a mobile device obtained with root privileges.

#### 3.3 File-based and Content-based Backups

The choice of which and how many files are included in a backup is left to each individual application. Some applications backup entire files which is the more straightforward way of backing up information but may use more space. Others choose to backup only relevant information from a single file as opposed to the entire file. The former is called a file-based backup and the latter is called a content-based backup.

Android offers two backup mechanisms [9], *Android Backup Service* backup and *Android Auto Backup*. Android Auto Backup is the default backup mechanism and is file-based with the option to exclude specific files. It has a limit of 25MB per application and requires no additional implementation from developers. Android Backup Service is content-based in the form key-value pairs. It has a limit of 5MB per application and requires the developer to implement a backup agent. On the other hand, iOS backups are file-based with the option of excluding certain files from backups [2]. The exception is the iPhone key-chain [1] which is encrypted.

#### 4 EVALUATION MODEL

The authors propose a general data integrity evaluation model comprised of three stages.

- Pre backup: Create a full system image using elevated privileges.
- (2) Backup: Create a backup using the local backup mechanism of choice.
- (3) Post backup: Create another full system image.

The contents of each backup are considered as key-value pairs for consistent analysis. In the case of file-based backups, the key is the name of the file and the value its complete bit-string, and in the case of content-based backups, each key-value pair contained in the file is a separate entry. For data analysis the stages are denoted as *Pre, Backup* and *Post* respectively.

#### 4.1 Data Evaluation

The number of and content of the key-value pairs is compared between the three stages for mismatch.

To conduct the analysis the following sets and values are collected from the data. First we list the sets and values pertaining to *Pre* and *Backup*.

- *E*: the set of all names contained in either *Pre*, *Backup* or *Post*.
- *N*<sub>over</sub>: the set of all names that are contained in *Pre* but not in *Backup*.
- *N*<sub>new</sub>: the set of names that are not in *Pre* but are in *Backup*.
- *N*<sub>both</sub>: the set of all names that are both in *Pre* and *Backup*.
- *V*<sub>eq</sub>: the set of all names that are the same for *Pre* and *Backup*.
- V<sub>ch</sub>: the set of all names of key-value pairs that are contained in both *Pre* and *Backup* but their values are different.

Then the set of all key-value pairs contained in  $V_{ch}$  is compared to *Post* and split into the following sets:

- $P_{\text{mis}}$ : the set of names that are in  $V_{\text{ch}}$  that aren't in *Post*.
- *P*<sub>mback</sub>: the set of all names in *V*<sub>ch</sub> in *Post* where the value is the same both in *Backup* and in *Post*.
- *P*<sub>mpre</sub>: the set of names in *V*<sub>ch</sub> where the value is the same in both *Post* and *Pre* but is different in *Backup*.
- *P*<sub>nom</sub>: the set of names that are contained in *Pre*, *Backup* and *Post* and their values differ in each stage.

We are especially interested in the set  $V_{ch}$  which is exactly the set of files with conflicting value between *Pre* and *Backup* and may give a false positive or a false negative. To provide a measure of uncertainty process and analysis are repeated as many times as deemed necessary.

#### **5 IPHONE BACKUP PROCESS**

The process for preparing the iPhone device uses a jailbreak to obtain superuser privileges. The authors of the original paper used the checkra1n jailbreak, but the device we acquired uses iOS version 15.8.2 which is not yet officially supported by checkra1n [4]. We instead used the palera1n jailbreak [11], which uses the same

checkm8 exploit as checkra1n and supports iOS version 15.8.2. The procedure goes as follows:

- The palera1n jailbreak is applied to the device. With the jailbreak comes the Sileo package manager which allows installation of third-party applications.
- (2) Sileo is used to install OpenSSH and NewTerm3. OpenSSH allows connection to the device via SSH and NewTerm3 gives a command line utility on the phone.
- (3) The command *sudo passwd root* is used to enable the root user, which was otherwise disabled.
- (4) Then *idevicepair* from *libimobiledevice* is used with the command *pair* to enable a trusted connection between the computer and the device.
- (5) *iproxy* from *libimobiledevice* is used to route the device port to the local device port.
- (6) Once the connection is established the files can be obtained using the secure file transfer protocol.

The backup is created using *idevicebackup2* from *libimobiledevice*. The backup must then be reconstructed to restore the original file sytem.

In Table 1 we list the iPhone 7 device specification and the exact versions of the tools used for the iPhone backup process.

IPHONE 7 DEVICE INFORMATION			
Device	iPhone 7		
Model Number	MN932ZD/A		
Storage	128 GB		
iOS Version	15.8.2		
SOFTWARE INFORMATION			
libimobiledevice	1.3.0		
palera1n/palen1x	v1.1.4		
Newterm	3.0 beta1		

 Table 1: iPhone device hardware and software details and software used for data acquisition.

#### 5.1 Data analysis

Data evaluation was performed in Python using the Sqlite3, hashlib and OS libraries. The OS library was used to iterate through Pre, Post and Backup directories to create name-value pairs for each file, the hashlib library was used to calculate hash-sums for the value of each file and the Sqlite3 library was used to read manifest.db to access file names. The results are presented in Table 2.

Our results are consistent with the original authors results. We found that no files included in the backup were changed during the process. Additionally, the only files that were different in value from *Pre*, were database files. The different value of these files can be attributed to Write-Ahead Logging used by the SQLite database.

#### 6 ANDROID BACKUP PROCESS

On android phones, the process to prepare a phone for evaluation is a bit more involved. It requires flashing a modified system image Smartphone Local Backup

SET	SIZE
Pre	341485
Backup	437
Post	341561
E	341485
Nover	341048
N <sub>new</sub>	0
$N_{ m both}$	437
$V_{ m eq}$	386
V <sub>ch</sub>	32
N <sub>mis</sub>	0
N <sub>mback</sub>	0
N <sub>mpre</sub>	32
V <sub>nom</sub>	0

Table 2: Evalution results for the iPhone device.

onto the device and it also requires additional steps specific to Samsung devices.

- Samsung firmware is downloaded using the *Frija* tool [13]. The device model, region and IMEI is required.
- (2) The downloaded firmware is modified using the *Magisk* tool, which modifies an image to allow apps to gain superuser privileges,
- (3) The modified firmware is flashed onto the mobile device using the *Odin3* tool [12], flashing the firmware is destructive and may not preserve certain data.

The modified firmware enables a shell session to obtain superuser privileges, but it does not allow the ADB client to run as root. The root user is disabled by default on production builds of the firmware. To bypass the restriction the following options are available:

- (4) (a) Start a shell session as root to obtain write permission to the /system folder, then navigate to the system/build.prop file and add the line ro.debuggable=1, which tricks the ADB server into believing it is not running on a production build.
  - (b) Install the application *adbd-insecure* [3], which is designed as a quick way to accomplish what is described in point (1).
  - (c) Instead of obtaining root permissions, you can copy the files onto a removable medium, and extract files from there. This option may be less desirable as it requires connecting a new media, but it does not involve changing any system files or installing any applications.

Once the phone is prepared, the evaluation proceeds as dictated by the general evaluation model. In the article, the authors used ADB as their local backup method of choice along with app downgrading to access otherwise unavailable data. We were unable to replicate their results due to issues with the app downgrade procedure. The idea is to use ADB to install an older version of an application that still permits local backup through ADB. But this procedure is not an intended feature and has been disabled on newer versions of Android.

Google Play Store requires new applications and updates to target an Android API level within one year from the latest major Android OS version [10]. In practice this means that any application that has been updated recently is set to disallow debugging by default [8]. One option to bypass these restrictions would be to modify the applications to set the debuggable property to true, but this is beyond the scope of this paper and would go against the spirit of the procedure.

In Table 3 we list the Samsung Galaxy S8 device specification and the exact versions of the tools used for the Android backup process.

SAMSUNG GALAXY S8 DEVICE INFORMATION			
Device Samsung Galaxy S8			
Model	SM-G950F		
Region	SEE		
Storage	64 GB		
Android	9		
Build number	PPR1.180610.011		
SOFTW	ARE INFORMATION		
adb	28.0.2-debian		
Magisk	27.0		
Frija	v2.0.23364.3		
Odin	3.13.1		
adb-insecure	v2.0		

Table 3: Android device hardware and software details along with the software used for data acquisition.

#### 7 CONCLUSION

The general model of evaluation appears as a vital addition to the mobile forensics toolkit. It allows a better understanding of which data is changed upon backup and how it is changed and it also provides a framework for evaluating these changes in a scientific manner. The same process could also be used to access any physical data acquisition methods.

But it does have some downsides, the process is quite time intensive and can differ heavily across different devices and operating systems, it is hampered by the apparent lack of support for local backup options and it can't discern if changes are made by the applications used to obtain data or by the custom firmware or jailbreak used to obtain elevated privileges.

#### REFERENCES

- [1] Apple. 2024. Keychain Services. https://developer.apple.com/documentation/ security/keychain\_services
- [2] Apple. 2024. Optimizing Your App's Data for iCloud Backup. https://developer.apple.com/documentation/foundation/optimizing\_your\_ app\_s\_data\_for\_icloud\_backup/
- [3] Chainfire. 2012. Abdb insecure v2.0. https://xdaforums.com/t/2014-11-10-rootadbd-insecure-v2-00.1687590/
- [4] checkra1n. 2020. The state of checkra1n on iOS 14. https://checkra.in/news/ 2020/09/iOS-14-announcement
- [5] Joe Fedewa. 2019. Google warns that ADB backup and restore may be removed in a future Android release. https://www.xda-developers.com/adb-backup-andrestore-depreciated/
- [6] Peijun Feng, Qingbao Li, Ping Zhang, and Zhifeng Chen. 2018. Logical acquisition method based on data migration for Android mobile devices. *Digital Investigation*

(2018). https://doi.org/10.1016/j.diin.2018.05.003

- Julian Geus, Jenny Ottmann, and Felix Freiling. 2024. Systematic Evaluation of Forensic Data Acquisition using Smartphone Local Backup. https://doi.org/10. 48550/arXiv.2404.12808
- [8] Google. 2024. Behavior changes: Apps targeting Android 12. https://developer. android.com/about/versions/12/behavior-changes-12#adb-backup-restrictions
- [9] Google. 2024. Data backup overview. https://developer.android.com/guide/ topics/data/backup
- [10] Google. 2024. Target API level requirements for Google Play apps. https: //support.google.com/googleplay/android-developer/answer/11926878
- [11] palera1n. 2024. palera1n. https://github.com/palera1n/palera1n
- [12] Samsung. 2024. Samsung Odin 3.13.1 with Android ROM Flashing Tool. https: //odindownload.com/
- SlackingVeteran. 2023. Frija v2.0.23364.3. https://github.com/SlackingVeteran/ frija/releases

# **Mobile Forensics**

Antonios Fritzelas\* Nikolina Petrovikj\* af92001@student.uni-lj.si np49868@student.uni-lj.si University of Ljubljana Ljubljana, Slovenia

### ABSTRACT

This study explores mobile forensics and focuses on MOBILedit Forensic Express's (MFE) effectiveness in locating digital evidence on cellphones, especially in Western messaging applications. We evaluate the advantages, disadvantages, and moral implications of MFE through empirical study. We also discuss the differences between government-only and commercial technologies and argue for fair use of forensic capabilities. Our results show how difficult it is to get around device security and how mobile forensics is developing. Through our analysis of messaging app data accessibility, we clarify the intricacies of this subject. In the end, this study aims to advance knowledge about how mobile forensics can fight cybercrime while protecting people's right to privacy.

#### **1 INTRODUCTION**

Smartphones have become indispensable in today's fast-paced world, thanks to their diverse functions and applications, which closely mirror those of computers. This surge in technology has elevated their significance, making them vital tools for modern living. Globally, there are roughly 6.94 billion smartphones in use, which constitutes approximately 85% of the total global population of around 8 billion. [2] Nonetheless, the widespread use of cellphones and instant messaging software has also contributed to a rise in cybercrime, in which the device typically acts either as the subject of a crime or as a data source. Digital evidence must be gathered in order to investigate such crimes, which calls for the employment of forensic software designed specifically for mobile devices.

The increasing sophistication of smartphones parallels the rise in their misuse for illicit activities, particularly through instant messaging applications such as WhatsApp, Facebook messenger, Telegram, and Snapchat. In Indonesia, for example, the MiChat application has been associated with various criminal activities, ranging from drug trafficking to human trafficking and even murder.[4] These cases underscore the necessity for robust mobile forensic methodologies to extract and analyze digital evidence from smartphones effectively.

Using popular western social networking applications as a case study, this essay seeks to compare and contrast the forensic procedures involved in retrieving digital evidence from smartphones. Using the tool MOBILedit Forensic Express (MFE), the study assesses how well this program recovers artifacts that are essential for investigating digital crimes. The findings highlight the strengths of the tool in terms of data extraction, acquisition features, identification, and evidence preservation, providing insights to aid investigators in efficiently handling digital crime cases, as well as the areas which could be improved to further advance the mobile forensics field.

#### 2 STATING THE PURPOSE OF THE EXPERIMENT

We set out on a mission to research the effectiveness of mobile forensic tools in retrieving digital evidence from popular messaging apps, as part of our exploration and evaluation of these technologies in the scope of an investigation.

In the original paper by Riadi, I., Yudhana, A., Fanani [4], the analysis primarily focuses on the Asian app MiChat. To compare these findings, we examined well-used European messaging services. Our goal was to assess the flexibility and adaptability of MOBILedit Forensic Express and other forensic tools in various linguistic and cultural contexts. We aimed to understand the types of data that can be collected and explore the potential for identifying digital crimes through thorough testing and analysis of these platforms.

Communications offer invaluable insights that can significantly aid in piecing together the puzzle of a person's behavior. By analyzing the content, tone, and frequency of messages exchanged, investigators can discern patterns of communication, uncover hidden agendas, and identify potential collaborators or co-conspirators. Moreover, nuances in language and expression within digital messages can illuminate underlying emotions, attitudes, and intentions, providing investigators with a comprehensive profile of the individual in question. Such access allows for a deeper understanding of the suspect's thought processes, motivations, and connections, thereby facilitating a more informed and effective investigative process.

#### **3 COMPARISON WITH OTHER TOOLS**

The primary goal of commercially accessible digital forensics programs like MOBILedit, Cellebrite, and Final Mobile Forensics is to retrieve data from desktops and mobile devices for use in judicial and investigative proceedings. These are legal compliance techniques that are often utilized by companies, private investigators, and law enforcement authorities. In their study, Riadi, I., Yudhana, A., and Fanani regard MOBILedit as one of the best tools available.

Numerous types of data are usually extracted by these technologies, including call records, text messages, emails, images, videos, browser history, app data, and location data. The degree of data extraction, however, may differ based on factors like device compatibility and security controls put in place by the device maker.

Conversely, government and intelligence organizations utilize extremely advanced surveillance software for covert operations. These programs are made to break into mobile devices and take

<sup>\*</sup>Both authors contributed equally to this research.

large volumes of data without the user's awareness or approval, typically utilizing zero-day exploits, and being extremely effective. [7]

Governmental spyware, in contrast to programs that are sold commercially, has the ability to go over security protocols and encryption, giving it access to almost all data that is stored on a device, even files and communications that are encrypted. This includes private data like messages, passwords, location information, and even access to the microphone and camera.

This type of espionage software contains malicious code, processes, and apps that are used to spy, collect data, and report back what the user does on the device. Some of the methods used compromise the original apps already installed on the device. As a result these tools, while more effective, are not compliant with formal forensic practices, compromising data integrity, but are still widely used.

These two types of tools differ primarily in their intended application, degree of sophistication, ethical and legal consequences. Commercially available digital forensic tools are employed legally and for valid investigative reasons; however, government spyware tools give rise to grave ethical concerns about civil liberties, privacy, and the possibility of abuse by authoritarian regimes or for unauthorized surveillance.

Pegasus and Predator are two examples of spyware technologies that have spurred discussions about the need for increased accountability, transparency, and legislation to stop their exploitation and safeguard people's right to privacy. There is growing pressure on governments and tech corporations to address these issues and make sure that surveillance technologies are used ethically and in compliance with human rights norms.[6]

One startling finding about the accessibility and availability of mobile forensic tools surfaced during our investigation. Even though programs like MOBILedit Forensic Express are available to the general public and have extensive functionality, we found that their capabilities were noticeably different from those of products that are only sold to governments, like Predator. When it comes to extracting and evaluating digital evidence, these government-only tools perform better and are more sophisticated, often utilizing malicious techniques, and highlighting the notable difference between tools available to the general public and those that are only available to governmental entities. There is an even bigger gap in the capabilites present in open source tools such as Autopsy. These disparities present relevant queries about fair access to forensic tools and raise ethical concerns. The difference emphasizes the necessity of increased openness and democratization of forensic instruments, guaranteeing that non-governmental entities and people have access to the same, lawful tools for successfully fighting cybercrime, while concurrently ensuring public safety.

#### 4 SECURITY BYPASSING

In our experimentation, we encountered challenges when attempting to access data from various devices, irrespective of their operating systems. Despite utilizing MOBILedit Forensic Express and exploring every possible avenue within the tool, we found that certain security measures on the devices impeded our efforts to extract data without the necessary access credentials. It's important to note

that the ability to bypass device security measures may vary depending on factors such as device model, operating system version, and the specific security configurations implemented by the user. While forensic tools provide valuable capabilities for digital data extraction, they may encounter challenges when faced with robust security measures designed to protect user privacy and prevent unauthorized access, especially when analyzing recently released devices on the newest available security patches. Furthermore, the legal and ethical considerations surrounding the use of forensic tools to bypass device security measures are complex and may vary across jurisdictions. In some regions, law enforcement agencies may have legal authority to compel individuals to provide access to their devices under certain circumstances, while in others, such actions may be subject to stricter legal scrutiny or limitations. Overall, our experimentation underscored the importance of ethical and lawful practices in digital forensic investigations, as well as the need for continued innovation and development of forensic tools to address the evolving landscape of device security and data protection.

# 4.1 The importance of rooting for access to secured data

Rooting or jailbreaking, is the process of bypassing restrictions on mobile devices to gain elevated privileges to the mobile system. While there are various methods and tools to do so, on Android devices this process typically involves unlocking the bootloader and flashing a custom recovery such as TWRP or Magisk, while on iOS devices it involves exploiting known security vulnerabilities such as checkra1n and chimera.

Rooting or jailbreaking a device allows forensic investigators to access data that would otherwise be inaccessible. This can be crucial in uncovering evidence related to criminal activities, such as terrorism, fraud, or cybercrime. For example, deleted messages may contain valuable evidence that could be instrumental in solving a case. Furthermore, it can enable investigators to use specialized forensic tools that require elevated privileges to function effectively. These tools can analyze the device's file system, and recover information that may have been intentionally concealed by the user. In our case, MOBILedit requires root access for the application data of Snapchat. Later, we will compare the viability of MOBILedit's unlocking methods on various devices.

Before proceeding with these methods, it's crucial to create a complete backup of the device's data. This ensures that in case anything goes wrong during the rooting or jailbreaking process, you can restore the device to its original state without losing any important information. While these methods can provide crucial access to additional data for investigators, its importance must be weighed against potential risks and legal implications. Certain kernel flaws have been reported to impact the /data partition on the eMMC chip in Android smartphones. This partition can become damaged during specific activities like wiping and flashing. The digital investigator must keep this in mind and adhere to their country's legislation for electronic evidence handling.[1]

#### 4.2 App Downgrading

Reverting a mobile application to a previous version from the one that is presently loaded on a device is known as app downgrading, and it usually happens for compatibility concerns or the desire to use features that have been deleted. In mobile forensics, app downgrading offers a practical way to obtain data, especially in situations when rooting a device might not be an option. Still, there are hazards associated with this strategy. Data integrity may be compromised, for example, by degrading programs that make unanticipated changes to the data structure or format. Additionally, devices may become vulnerable to security flaws in out-of-date software versions if earlier versions of programs are used. Furthermore, there may be problems with forensic instrument compatibility, which might make reliable data extraction difficult. MOBILedit claims to prevent the problem of data loss and detect with 100% reliability if the app downgrade can be attempted. Despite these dangers, app downgrading might yield insightful information for forensic analysis if done carefully.

#### 4.3 Exploring Alternative Paths in Mobile Data Access

The availability of mobile data to forensics experts is not always a given. In most legislations, a suspect has the right to deny providing a password, even in the case of a warrant. As we have mentioned, the amount of data that can be extracted from a locked mobile phone varies by device model and manufacturer. In this section, we will perform a comparison between different vendors and software versions and some of the known vulnerabilities to be exploited for protection bypassing. Most of these vunerability hacks are also built into MOBILedit.

#### On Android:

- Tensor Chips: A bug in the SIM unlocking operation could allow an attacker to bypass the Google Pixel lock screen without the need of a password (CVE-2022-20231, CVE-2022-20364) [8]
- Qualcomm: EDL mode can be used to create a physical dump without the need for a password or pattern lock. In some cases, data may be encrypted (vendor-specific) - included in MOBILedit
- Exynos: Samsung devices can be unlocked with some bruteforce techniques that involve rebooting into ODIN mode [3]
- Mediatek: Create physical dump using MTK mode. In some cases, data may be encrypted (vendor-specific) - included in MOBILedit

#### On iOS:

- Using iTunes backup that could be found from a suspect's computer
- If the iPhone is jailbroken, MOBILedit can extract all files, including application sandboxes or system files. To achieve this, MOBILedit needs iTunes installed

Certainly, including the use of search warrants in conjunction with biometrics and compelling individuals to unlock their devices adds another layer of complexity to mobile forensics investigations. Here's how this aspect could be addressed: In both the US and the EU, law enforcement agencies can obtain search warrants to seize and search mobile devices for evidence relevant to an investigation. However, the use of biometrics to unlock devices under the authority of a search warrant raises legal and ethical considerations. In the US, the legal landscape regarding the force of use in mobile forensics is evolving. Courts have issued conflicting rulings on whether law enforcement can compel individuals to provide biometric data, such as fingerprints or facial scans, to unlock their devices without violating the Fifth Amendment protection against self-incrimination. Some courts have held that biometric data falls under the "foregone conclusion" doctrine, where providing the data does not reveal anything the government does not already know, which constitues biometric unlock different to password unlock.

#### 5 ACCESSIBILITY OF MOBILE FORENSIC TOOLS

The demand for easily available mobile forensic tools has increased due to the surge in cybercrime. Both MOBILedit and Autopsy are strong tools for mobile forensics, but they offer unique capabilities and work in various ways. An open-source program called Autopsy usually needs a disk image or backup from a mobile device to be obtained before it can be used for analysis. This means that before the data is imported into Autopsy for parsing and analysis, data extraction (for example, via ADB for Android or iTunes for iOS) must take place independently. However, direct access to the mobile device is one of the main benefits of MOBILedit, a proprietary program. This feature streamlines the process for investigators by allowing them to extract and evaluate data in a single workflow. Without the need for a middle imaging step, MOBILedit may connect to a variety of mobile devices, execute data extraction, and provide instant insights into contacts, messages, call logs, and app data. Because of its direct access capacity, MOBILedit is a more effective option for prompt forensic analysis, especially in situations involving real-time investigations and rapid data retrieval. Even while the general public can use tools like MOBILedit Forensic Express, there is still a big difference in the level of efficacy and sophistication when it comes to tools that are only available to government agencies, like Predator. There are significant concerns regarding the availability and accessibility of sophisticated forensic tools given this discrepancy in tool capabilities. The scarcity of state-of-the-art tools that offer thorough data extraction and analysis makes it difficult for people, private investigators, and non-governmental organizations to conduct effective investigations into digital crimes. Forensic tools must become more accessible in order to guarantee a just and equitable method of investigating digital crimes. Narrowing the gap between governmental and non-governmental entities can facilitate more effective investigations and increase accountability in the digital forensic area by expanding access to advanced forensic capabilities. Transparency, cooperation, and the creation of user-friendly forensic tools that satisfy the requirements of a wide range of users should all be encouraged. This entails addressing privacy issues, guaranteeing legal compliance, and developing frameworks for the morally and responsibly use of forensic technologies.

DigFor'24, June 2024, Ljubljana, LJ, Slovenia

#### **6** EXPERIMENT

This section provides a thorough study of our experiment examining the accessibility of data across several messaging apps, comparing its findings with knowledge from previous studies on the capabilities of forensic tools, in our case only one MOBILedit Forensic express. All things considered, our experiment clarifies the differing levels of data accessibility amongst various messaging apps, which are impacted by things like platform encryption, data retention guidelines, and the extraction tool's forensic powers.

• WhatsApp: We were able to extract a large amount of data from WhatsApp, including contact information, media files (photos, movies, and audio), and text messages. This is consistent with earlier research that highlighted Michat's strong data extraction capabilities using this tool.

30 🗩 38978241889@s.whatsa	pp.net ( <u>Nikolina</u> )			Received
Hmm do you already have wh	at i asked you for?			
То	38669673584 (Nick)			
ID	3ADDB624C2368C0229D6			
Seconversation	38978241889@s.whatsapp.net			
Sent	<b>O</b> Uploaded	ODelivered	Read	
2024-05-10 22:44:21 (UTC+2)		2024-05-10 22:44:22 (UTC+2)		
31 🗩 38978241889@s.whatsa	ipp.net ( <u>Nikolina</u> )			Sent
of course				
From	38669673584 (Nick)			
ID	6F639EA7F8042269F8C935A2FFFE6444			
Seconversation	38978241889@s.whatsapp.net			
Sent	Oploaded	C Delivered	C Read	
2024-05-10 22:44:35 (UTC+2)	2024-05-10 22:44:35 (UTC+2)			
32 🗩 38978241889@s.whatsa	ıpp.net ( <u>Nikolina</u> )			Sent
we can meet at the park				
From	38669673584 (Nick)			
ID	1665DDA954AD4BF9237020A6407E9141			
Section Conversation	38978241889@s.whatsapp.net			
Sent	Oploaded	C Delivered	Read	
2024-05-10 22:44:47 (UTC+2)	2024-05-10 22:44:48 (UTC+2)			

# Figure 1: Extracted WhatsApp Messages Using MOBILedit Forensic Tool

- **Telegram**: Unfortunately, we were only partially successful in retrieving data from Telegram. Telegram's robust encryption techniques made extraction difficult. We were able to get the contacts list, and the usernames of the people which the person had an interaction with. The inability of the platform to provide us with messages, media files, or contact details brought to light possible difficulties when attempting to obtain data from the encrypted messaging apps using the MOBILedit tool.
- Facebook Messenger: Our analysis of Facebook Messenger yielded detailed insights into user conversations and media files. During our examination, we were able to get data on the account's friends and connections. Furthermore, discussions and media files posted via Facebook Messenger may be accessed, even temporary files, offering insights on user communication styles and shared material. This was the least secure application that we accessed during our experiment, and the findings highlight the importance of encryption in mobile security, and also the aggressive nature of Meta's data collection, which has been scrutinized by researchers and legislators.
- **Snapchat**: Extraction from Snapchat was unsuccessful due to security restrictions, which require rooting for data extraction, but since our experiment was conducted without rooting we did not get any meaningful data from it. The instructions for the MOBILedit tool clearly stated that for

Fritzelas and Petrovikj

the Snapchat application we need a rooted phone, otherwise we would not be able to extract any info. [5]

While certain platforms, such as Facebook Messenger and WhatsApp, produced large data sets with conversations, media files, and contact details, other platforms, such as Telegram and Snapchat, presented difficulties when it came to extracting data, especially when it came to encrypted or transient material. These revelations highlight how crucial it is to use a variety of forensic techniques and tools in order to successfully negotiate the intricacies of digital investigations. It is unclear how much more data would be accessible with root access, however as we addressed these methods also introduce risk of evidence loss and are not always effective. Ultimately, it is up to the digital investigator to decide the level of risk to take on a case by case basis (for eg. an urgent missing's persons case might have a higher level of urgency). One additional key finding relates to the availability of message metadata as an important forensic tool. Mobile phones record a multitude of interesting data points for the forensic report, and this is also evident in our application analysis. Geographical points layed out in an interactive map, timestamps and pinged devices (cell towers, bluetooth and wifi devices) provide investigators an accurate image of a suspect's movements.

#### **Table 1: Phones Used in the Experiment**

	Model	Storage	Android	Build Number
Google Pixel	G-2PW4200	32 GB	10.0	PQ3A.190605.003
	Model	Storage	IOS	Build Number
Iphone 12	A2403	128 GB	17.0	21D61

### 7 SOME INTERESTING FEATURES OF MOBILEDIT TOOL

During our experiments, we also shortly tested other innovative features offered by the MOBILedit tool.

- Photo Recognizer: This feature is designed to identify currency, drugs, extremist symbols, weapons, nudity and sensitive documents in photos and videos extracted from the device. The detections were of varying accuracy, sometimes flagging false positives. This feature seems very promising as AI models in image recognition continue to evolve, as they will be able to swiftly filter out content of interest, speeding up the investigation procedure
- Face Matcher: It is often desirable to narrow down a search to specific people associated with the suspect. Face matcher aims to do just that: filter images and videos containing people of interest.
- **Password Extractor**: Mainly addressing on-device accounts, this feature could be used to open up new investigation pathways for investigators (eg. IP camera credentials)
- Web & Cookies data: This is the most commonly-used resource in computer investigation. In the case of MOBILedit, it is able to retrieve the expected data. This could be beneficial for tracing or validating cross-device activity

#### 8 **CONCLUSION**

Our exploration into mobile forensics highlights the critical role of smartphones in digital crime investigations. While tools like MOBILedit Forensic Express offer valuable data extraction capabilities, they face challenges with security measures and encryption. Ethical concerns arise from government surveillance software like Pegasus, emphasizing the need for accountability and legislation. Our experiment reveals varying data accessibility across messaging apps, underlining the importance of diverse forensic techniques. In navigating these complexities, it's crucial to balance investigative needs with privacy rights. By advancing forensic tools and addressing ethical considerations, we can ensure fair and effective digital investigations while upholding principles of transparency and human rights. In conclusion, our experiment contributes to

the field of digital forensics by evaluating data accessibility across popular messaging platforms. Future research should focus on enhancing forensic methodologies and tools to address evolving security measures and improve data extraction efficiency in challenging environments.

#### REFERENCES

- [1] Brick (electronics).
- Forecast number of mobile devices worldwide from 2020 to 2025 (in billions)\*. [2] [3]
- How to perform samsung exynos extractions with oxygen forensic detective.
- [4] Mobile forensic tools for digital crim[5] Supported applications mobiledit. Mobile forensic tools for digital crime investigation: Comparison and evaluation.
- [6] Technical analysis of pegasus spyware: An investigation into highly sophisticated espionage software.
- [7] What is predator spyware?
- Jai Vijayan. Billions of Android Devices Open to 'Dirty Stream' Attack. Cybercrime. [8] Dark Reading, 2024.



4 - Dnevniki in forenzika / Logs and Forensics

Izzivi

Študij

Ustvarjanje novih svetov

Raziskovanje

FRI

# Every step you take, I'll be tracking you: Forensic analysis of the tile tracker application

Aljaž Sebastjan Ahtik Univerza v Ljubljani, Fakulteta za računalništvo in informatiko Ljubljana, Slovenia aa1353@student.uni-lj.si

#### ABSTRACT

The surge in popularity of personal Bluetooth trackers has sparked a demand for forensic tools that assist law enforcement in retrieving artifacts. With reported sales of 40 million Tile devices to date, Tile trackers stand as one of the most favored personal Bluetooth trackers. However, this growth has brought about challenges, as incidents of malicious use of Bluetooth trackers have also risen. We present a paper that examines the Tile ecosystem and its associated application across iOS, Android, and Windows platforms. The data extracted from the application contains geolocation coordinates spanning the previous 30 days. The authors present an open-source tool, named Tile Artifact Parser (TAP), capable of parsing these forensic artifacts from the Tile application and detecting potentially spoofed data.

#### **KEYWORDS**

Tile, Memory forensics, Application forensics, Bluetooth tracker

#### **1** INTRODUCTION

The article [11] introduces Bluetooth trackers, focusing on Tile devices, which transmit data to companion devices for tracking personal belongings. Tile trackers (tags) connect via Bluetooth Low Energy to the Tile app on a companion device, updating location data using the companion device's GPS services. To enhance location tracking, Tile creates a network of user devices, which update the location of any nearby tag. See Figure 1 for details. The market for such trackers is rapidly expanding, with Tile being a prominent player, selling over 40 million devices. However, concerns about privacy and potential misuse, such as stalking, have emerged. Life360's acquisition of Tile further raises questions about security implications. To address these concerns, the article proposes investigating the data produced by Tile devices and companion applications for potential forensic analysis. The contributions of [11] include a peer-reviewed analysis of the Tile application, the development of an open-source Python tool called Tile Artifact Parser (TAP) for forensic analysis, incorporation of memory forensics, and mitigation strategies against anti-forensics techniques.

#### 2 RELATED WORK

This section discusses the increasing popularity of Bluetooth personal trackers, referencing a prior forensic analysis conducted on similar applications like Tile, as well as existing research on application analysis.





Figure 1: Tile IoT ecosystem. The Tile device is detected by the user's companion device through bluetooth. The location of the tracker is estimated using the location of the companion device. At the same time devices of other users running the Tile application can detect the tracker and update its location.

#### 2.1 Application analysis

Research on application forensics has seen significant progress, with previous studies focusing on advancing forensic techniques and their applications [7]. Some research has delved into obtaining digital evidence and decrypting network traffic from social media applications. For instance, analysis of messaging platforms like Discord for Windows has unveiled methods to retrieve sensitive and deleted data from local application files, aiding forensic investigations [10]. Similarly, investigations into location-based applications such as Happn and Life360 have raised safety concerns. Happn, a dating app, was found to track users' locations and match them with others, as evidenced by artifacts obtained from iTunes backups and physical image artifacts [9]. Life360, a family tracking app, yielded limited initial findings on both iOS and Android devices, with more comprehensive data obtained by rooting an Android device, albeit not an ideal step in research [1].

Another study [8] examined health and fitness apps on Android, recovering a variety of information that could be useful as digital evidence, such as personal information (name, e-mail, age, gender, birthday, zip code, height, weight, and blood pressure), user health information including notes of previous doctor's visits, geolocation data, photos, and in the case of MyFitnessPal, plaintext password data. Similarly to the discussed paper, the authors of the article developed a tool to simplify the process acquiring and locating forensically relevant data from such apps.

#### 2.2 Memory forensics

In digital forensics, efforts to recover artifacts from Random Access Memory (RAM) have seen significant progress in the past decade. Memory images offer insights into actively running processes on devices. Case and Richard III [5] critically analyzed the current state and objectives of memory forensics. Today, numerous open-source tools facilitate the extraction of forensic artifacts from volatile memory, empowering researchers and investigators to identify valuable evidence [4, 6, 13].

#### 2.3 Bluetooth trackers [AirTag and Tile]

Previous research has examined prominent Bluetooth Low Energy (BLE) trackers like AirTag and Tile, with distinct focuses and findings. Vulnerabilities were identified in AirTag firmware, demonstrating potential risks such as identity spoofing and communication hijacking in [12], where a voltage glitching attack was used to extract firmware. This was then used to change functionality, clone AirTags, and downgrade the firmware. Concerns were addressed around Tile-related crimes, conducting forensic analysis on iPhone X backups to recover geolocation artifacts. Vance C. [2, 3] analyzed Tile's Android and iOS applications, identifying location traces and developing artifact plugins for forensic analysis. Security flaws were investigated in Tile's firmware storage and communication protocols, prompting fixes from Tile in response to reported vulnerabilities [14].

While there's limited peer-reviewed research on Tile, this paper compares forensic acquisition processes across Tile's Android, iOS, and desktop applications. It's noteworthy that no investigations prior to [11] have been conducted on the latest Tile version post-Life360 acquisition, nor have they explored anti-forensic techniques' impact on Tile data. Additionally, the authors present a more comprehensive parsing tool capable of extracting forensic artifacts from both databases and memory dumps introduced in the source article.

#### 3 METHODOLOGY

The authors concentrated on forensically reconstructing and extracting user data from various Tile companion devices. To cover different device types investigators may encounter, an iPhone SE, an Android Huawei tablet, an Android Samsung Phone, and a Windows Virtual Machine (VM) were utilized. The study comprised phases including Setup and Scenario Creation, Data Acquisition, Data Analysis and Results, and Tool Development: TAP (Tile Artifact Parser). A detailed overview of the devices and software used is presented in Table 1.

#### 3.1 Setup and Scenario Creation

The paper's experimental setup began with factory resetting the iPhone SE, Huawei Android tablet, and Samsung Galaxy S10b, along

with creating a fresh Windows VM. The Tile application was installed on all devices, with the same account logged in except for the Galaxy S10+. Various Tile trackers were paired, and scenarios were created to simulate real-world events. Location spoofing apps were installed on Windows and Android to test anti-forensics techniques.

Two forensic workstations, an iMac and a Windows desktop, were equipped with necessary software. Real-world investigation scenarios were designed to demonstrate Tile data acquisition's relevance, including scenarios involving discovery of a device at a crime scene, searching a suspect's home with a running Tile desktop application, and utilizing location spoofing applications.

Data generation involved walking predetermined paths with Tile and companion devices, presented on Figure 2, depositing and retrieving devices as per scenario requirements. Spoofing applications were used for injecting falsified locations. The next phase was data acquisition.

#### 3.2 Data Acquisition

Data acquisition involved capturing artifacts from the Android device using a Cellebrite Universal Forensics Extraction Device (UFED). Since a Huawei tablet was used, the UFED Decrypting Kirin Bootloader was necessary for an exhaustive backup. iPhone artifacts were retrieved by backing up the device to the iMac workstation and extracting files using iPhone Backup Extractor. For the Tile desktop application, a memory image of the VM was created using VMware, and the memory was inspected for forensic artifacts.

#### 4 DATA ANALYSIS AND RESULTS

The authors employed distinct data analysis techniques for each companion device due to differences in database formats between Android and iOS. Both devices harbored detailed log files showcasing Bluetooth payloads, encompassing MAC addresses, user UUIDs, device battery levels, and client models. However, iOS payloads offered more comprehensive data, including location coordinates, accuracy, and Tile UUIDs. These logs are illustrated in Fig. 2.

#### 4.1 iOS backup

In the iTunes backup, multiple SQLite database files were discovered, with particular importance attached to a database named com.thetileapp.tile-TileNetworkDB.sqlite.Within this database, the ZTILENTITY\_PLACEMARK table stored latitude, longitude, timestamps, postal codes, city, and street addresses. Notably, all timestamps were stored in the Apple Cocoa Core Data format. This data, when combined, effectively illustrates the general trajectory of a person or their belongings.

#### 4.2 Android backup

The full backup of the Huawei Android tablet contained various Tile data, including cache, XML, and SQLite database files. Some XML files provided details like linked email accounts, UUIDs, tokens, and cookies. Additionally, links within the Tile data opened HTML webpages displaying mapped locations corresponding to places traveled during the Setup and Scenario Creation phase.

However, the data obtained from SQLite databases was less statistically significant compared to iOS findings. One table, the Every step you take, I'll be tracking you: Forensic analysis of the tile tracker application

Hardware/Software	Use	Company	Software/Model Version
MediaPad M5 Tablet	Tile Companion Device	Huawei Technologies Co., Ltd	Android 8.0.0
iPhone SE	Tile Companion Device	Apple Inc	iOS 15.5
Galaxy s10+	Tile Companion Device	Samsung	Android 11.0.0
Windows Virtual Machine	Tile Companion Device	Microsoft Corporation	Windows 10.0.19042
VMWare Workstation Pro	Host VMs	VMWare	16.0.0 build-16894299
Tile (Android)	Tile Data Generation and Acquisition	Tile Inc.	2.75.0
Tile (iOS)	Tile Data Generation and Acquisition	Tile Inc.	2.105.0
Tile (Windows Desktop)	Tile Data Generation and Acquisition	Tile Inc.	3.3.27.0
Tile Mate	Tested Tile Device	Tile Inc	T1401S
Tile Sticker	Tested Tile Device	Tile Inc	T1501S
Tile Slim	Tested Tile Device	Tile Inc	T1601S
iPhone Backup Extractor	Extract iPhone Backup Data	Reincubate Ltd.	7.3.5.0
UFED	Phyical Acquisition for Android	Cellebrite	7.53.0.24
iMac	Data Acquisition/Analysis	Apple	Monterey 12.0.1
Alienware Aurora R12	Data Acquisition/Analysis, VM Host	Dell	Windows 10.0.19044
DB Browser for SQLite	View SQLite Databases	DigitalOcean, LLC	3.12.2
Fake GPS Location	Location Spoofing	Lexa	2.1.2
AnyGo	Location Spoofing	Shenzhen LuckyDog Tech. Co., Ltd.	5.9.2





(a) Mapping iOS location data.

(b) Mapping VMEM location data.

Figure 2: Result of TAP output with same date range.

notification\_content\_data\_table, contained a few entries with geolocation coordinates. Another table named regions resembled the ZTILENTITY\_PLACEMARK table in iOS but was not populated.

#### 4.3 Desktop application

The authors' forensic examination of the Tile desktop application involved analyzing system data files. Within the WindowsApps folder, three Tile data directories contained Dynamic Link Libraries (DLL) and other application data. Additionally, a memory image of the VM was captured while the Tile process was active.

Memory dumps (VMEM files) from the Windows VM revealed geolocation coordinates, timestamps in Epoch format, Tile UUIDs, and device names stored in plaintext. This capability to efficiently retrieve such data from a running machine offers valuable insights into a person's or their belongings' whereabouts without needing to recover a mobile device. The tool developed for this purpose, called TAP (Tile Artifact Parser), is detailed in the Tool Development section.

#### 4.4 Firmware

Tile firmware binaries were leaked via three links to an Amazon Web Service (AWS) bucket, where the files were stored server-side. These links were stored locally in the Tile iOS application data and in the Windows VMEM file. Interestingly, no authentication was necessary to download the firmware through these links. Moreover, the firmware binaries were also discovered in both iPhone and Android backups.

#### 4.5 Anti-forensics

To assess the impact of spoofed devices on the Tile Network, the authors conducted an experiment involving an iPhone and a Samsung phone:

- The iPhone and Samsung phones were set up with separate Tile accounts.
- A Tile Sticker paired with the iPhone was placed out of Bluetooth Low Energy (BLE) range, marked as lost.
- The location of the Samsung phone was modified using a GPS spoofing application.
- They walked near the "lost" Tile Sticker with the GPS spoofed Samsung phone.

Upon entering range, the iPhone received a location through the Tile Network, matching the modified location of the spoofed Samsung device. Additionally, they tested location spoofing on the iPhone. Although location spoofing apps are prohibited on the Apple App Store, they can be downloaded to a computer. Therefore, spoofing an iPhone's location requires physical connection to a computer with a spoofing application downloaded. When connected to a Windows location spoofing application, the Tile UI displayed falsified data.

# 5 TOOL DEVELOPMENT: TAP (TILE ARTIFACT PARSER)

The main goal of the article's tool development was to enhance the efficiency of acquiring and parsing forensically relevant Tile data. Tile Artifact Parser (TAP) is a command line forensics tool written in Python 3 designed for this purpose. It can parse artifacts from both memory images obtained from Windows systems and SQLite databases collected from iOS devices.

#### 5.1 Tool usage

TAP offers two primary functions:

- (1) Parsing data from Windows computer memory images.
- (2) Parsing data from SQLite databases recovered from mobile device logical backups.

It also enables detection of spoofed location data.

#### 5.2 TAP parsing function

TAP searches through images of a companion device or a VMEM file with the following goals:

- (1) Locate forensic artifacts from Tile companion devices.
- (2) Display the data in a user-friendly format.

The tool first searches a directory for any relevant files (memory dumps or SQLite databases) and then parses for geolocation data. When parsing a memory dump, TAP linearly searches through a VMEM file for a signature identifying Tile geolocation data. When parsing Tile iOS database files, TAP searches for the ZTILEN-TITY\_PLACEMARK table and extracts geolocation data.

The extracted data is then exported to a text file report, detailing input parameters and information about identified and parsed files. The report also includes a summary of the parsed data. TAP outputs geolocation coordinates from either the memory dump or the database file to an HTML file, which contains an interactive

Number of	Number of Data Points				
VMEM Input		TAP Outpu	t		
Valid	Partial/Missing	Valid	Partial/Missing		
0	0	0	0		
10	5	10	3		
10	0	10	0		
20	10	20	4		
20	0	20	0		
30	0	30	0		
50	0	50	0		
100	0	100	0		
500	0	500	0		
1000	0	1000	0		

Table 2: Comparison between the number of geolocation data points in testing of VMEM files and TAP output.

map with paths generated between coordinates based on correlated timestamps to show the path taken by a tile device.

#### 5.3 TAP spoofing detection

To address location data spoofing, TAP incorporates methods for detecting falsified data. It categorizes data with a potential risk of spoofing based on the calculated speed compared to some predefined reasonable speed (commercial airplane). It calculates speed using the distance between two points and their timestamps. Data points categorized as potentially spoofed are outputted at the bottom of the TAP report to alert investigators. These locations are also marked on the map with a black location marker.

#### 5.4 Tool evaluation

To assess the accuracy and exhaustiveness of TAP's parsing method, the authors conducted an evaluation of the output. The objective was to determine whether the data extracted by TAP statistically matched the original input data. They performed two sets of correlation tests, one for SQLite database files and another for VMEM files. To evaluate the robustness of their tool, they generated a diverse set of data files, including empty files, files with partial data, and files with a large number of data points.

#### 5.5 SQLite database and TAP correlation tests

In preliminary testing, they used a set of SQLite data directly extracted from the Tile iOS application database. The parser matched the data extracted from the input SQLite file completely. For more robust testing, they randomly generated ten SQLite datasets of varying sizes using a Python 3 script. In all tested datasets, TAP maintained a 100% accuracy rate in identifying valid data points.

#### 5.6 VMEM and TAP correlation tests

For the preliminary testing of the memory image parsing functionality, they used an 8 GB memory image collected while the Tile application was running on a Windows VM. To test TAP's collection method, they generated ten 2 GB memory images from a Windows VM. Each image contained varying numbers of geolocation coordinates randomly generated using a Python 3 script with two of the ten images including partial or missing data objects, simulating corruption during memory collection. TAP was designed to detect and document such corrupt data. As shown in table 2, comparing the VMEM input and TAP output showed a 100% accuracy in identifying valid data points. On average, invalid data points were correctly reported 50% of the time.

#### 6 **DISCUSSION**

The presented article provides a comprehensive analysis of Tile devices in 2022 across multiple platforms. The authors developed a tool, TAP, to parse the datasets found. When comparing their findings with previous work, they found that the Life360 acquisition of Tile has not significantly altered how the application stores data. Location data was found on all investigated devices, with log files containing MAC addresses, Tile UUIDs, phone battery level, and app activity. Geographic data from iPhone and Windows artifacts is timestamped, which is necessary for mapping user movements.

The article offers a forensic technique for recovering Tile data efficiently from various devices. Successful collection of data could show connections between suspects or victims at specific times and places. The technique estimates a target's location using location services from companion devices, typically mobile phones or laptops. Tile Premium allows users to view location data from the past 30 days, however this data can be recovered from an iPhone even without the Premium subscription. Additionally, the article presents an examination of the Tile desktop application, proving that valuable forensic information can be extracted. Memory analysis revealed more data points than in SQLite databases, suggesting the Tile servers store more data than local iOS databases. Furthermore, the potential for location spoofing, which could be used as an anti-forensics technique was investigated and a functionality to identify and flag potential spoofing was implemented, although it's limited to extreme cases. The authors suggest that implementing spoofing detection should be a priority for Tile to enhance user safety.

TAP's signature-based approach for parsing data from memory images limits its effectiveness against anti-forensics techniques such as injecting false data into memory. On Android devices, limited artifacts which did not contain timestamps were found due to the authors' decision not to root devices, prioritizing practicality for investigators.

#### 7 CONCLUSION AND FUTURE WORK

The article concentrated on gathering forensically relevant artifacts from Tile companion devices, analyzing them to uncover data crucial for investigations. The development of the TAP tool simplifies data extraction and analysis. Furthermore, the methodology in the article could be used to analyze and extract data from other Bluetooth trackers as well. Future research should delve deeper into anti-forensics methods and exploring approaches like positional evaluation of travel paths using consecutively collected location data points to enhance spoofing detection and into evaluating the effect the number of nearby companion devices has on the success of location spoofing. Additionally, investigating the physical Tile hardware, especially Tile Ultra as the first iOS and Android Ultra-Wideband (UWB) tracker, could yield valuable insights for the field. Moreover, the arrival of Google's Find my Device network in May of 2024 enables further research opportunities on Android devices.

#### REFERENCES

- Jason Bays and Umit Karabiyik. 2019. Forensic Analysis of Third Party Location Applications in Android and iOS. In IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). 1–6. https: //doi.org/10.1109/INFOCOMWKSHPS47286.2019.9093781
- [2] Vance C. 2019. Android Locating Location Data: The Tile App. https://blog. d204n6.com/2019/08/android-locating-location-data-tile-app.html 6.5.2024.
- [3] Vance C. 2020. iOS Tile App Part 2: Custom Artifact Boogaloo. https://blog. d204n6.com/2020/09/ios-tile-app-part-2-custom-artifact.html 6.5.2024.
- [4] Andrew Case and Golden G Richard III. 2016. Detecting objective-C malware through memory forensics. *Digital Investigation* 18 (2016), S3–S10.
- [5] Andrew Case and Golden G Richard III. 2017. Memory forensics: The path forward. Digital investigation 20 (2017), 23–33.
- [6] Peter Casey, Rebecca Lindsay-Decusati, Ibrahim Baggili, and Frank Breitinger. 2019. Inception: virtual space in memory space in real space-memory forensics of immersive virtual reality with the HTC vive. *Digital Investigation* 29 (2019), S13–S21.
- [7] Maxim Chernyshev, Sherali Zeadally, Zubair Baig, and Andrew Woodward. 2017. Mobile forensics: advances, challenges, and research opportunities. *IEEE Security & Privacy* 15, 6 (2017), 42–51.
- [8] Courtney Hassenfeldt, Shabana Baig, Ibrahim Baggili, and Xiaolu Zhang. 2019. Map My Murder: A Digital Forensic Study of Mobile Health and Fitness Applications. In Proceedings of the 14th International Conference on Availability, Reliability and Security (Canterbury, CA, United Kingdom) (ARES '19). Association for Computing Machinery, New York, NY, USA, Article 42, 12 pages. https://doi.org/10.1145/3339252.3340515
- [9] Shawn Knox, Steven Moghadam, Kenny Patrick, Anh Phan, and Kim-Kwang Raymond Choo. 2020. What's really 'Happning'? A forensic analysis of Android and iOS Happn dating apps. Computers & Security 94 (2020), 101833. https://doi.org/10.1016/j.cose.2020.101833
- [10] Kyle Moffitt, Umit Karabiyik, Shinelle Hutchinson, and Yung Han Yoon. 2021. Discord Forensics: The Logs Keep Growing. In 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC). 0993–0999. https://doi. org/10.1109/CCWC51732.2021.9376133
- [11] Lauren R. Pace, LaSean A. Salmon, Christopher J. Bowen, Ibrahim Baggili, and Golden G. Richard. 2023. Every step you take, I'll be tracking you: Forensic analysis of the tile tracker application. *Forensic Science International: Digital Investigation* 45 (2023), 301559. https://doi.org/10.1016/j.fsidi.2023.301559
- [12] Thomas Roth, Fabian Freyer, Matthias Hollick, and Jiska Classen. 2022. AirTag of the Clones: Shenanigans with Liberated Item Finders. In 2022 IEEE Security and Privacy Workshops (SPW). 301–311. https://doi.org/10.1109/SPW54247.2022. 9833881
- [13] Gandeva Bayu Satrya and Febrian Kurniawan. 2020. A novel Android memory forensics for discovering remnant data. Int. J. Adv. Sci., Eng. Inf. Technol. 10, 3 (2020), 6.
- [14] Mira Weller, Jiska Classen, Fabian Ullrich, Denis Waßmann, and Erik Tews. 2020. Lost and found: stopping bluetooth finders from leaking private information. In Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks. 184–194.

Received 12 May 2024; revised TBD; accepted TBD

# Analysis of Windows 10 & 11 Logging Mechanisms for Use in Digital Forensics

Gorazd Gorup Faculty of Computer and Information Science, University of Ljubljana Ljubljana, Slovenia gg6898@student.uni-lj.si

ABSTRACT

Due to Windows' high market share [5], forensic analysis of Windows 10 and 11 is a high-value target for researchers. In this paper we review existing methods for analysing Windows' diagnostic data, explain the structure behind the EventTranscript.db database, and re-implement our own version of Park and Lee's *DiagAnalyzer* tool [2] in order to obtain valuable forensic evidence from a snapshot of a Windows machine. Alongside Park *et al.*'s research into USB storage, Wi-Fi connection and Edge browser usage events, our paper proposes two new types of diagnostic data that is of high forensic value: laptop battery level, and detailed application activity.

#### **KEYWORDS**

Digital Forensics, Logging, Diagnostic Data, Windows, Application Activity

#### **1** INTRODUCTION

The Windows operating system is widely used [5], and therefore, the research on extracting information, which could be useful in digital forensics, is still being actively conducted. One of the ways to investigate the system is through gathered analytics and diagnostic data that Microsoft collects to troubleshoot and improve their products [1]. This data hints at users' behavioural and usage patterns, such as mounting of different storage devices, establishing wireless connections, using the Microsoft Store, etc. [1, 2]. All of this can aid forensic investigators in reconstructing the user's actions on the system and their timeline.

This work primarily builds upon the paper by Park *et al.* [2], reimplementing its *DiagAnalyzer* tool in Rust, and exploring methods of forensically analysing logging and diagnostic data from Windows 10 and Windows 11 systems.

First we tackle the related work on the subject of Windows telemetry and diagnostic data storage, covering some of the papers and efforts to decode and analyse potentially useful data for forensic investigations, including our reference paper above. We then describe how this type of data is stored in newer versions of Windows 10 and 11 systems, going over the details needed for careful forensic examination. We present our own contribution – a program that extracts certain events that differ from ones implemented in *DiagAnalyzer*, in order to aid digital investigators in finding relevant data. Finally, we share some of our notes and observations about the methods presented.

Simon Goričar

Faculty of Computer and Information Science, University of Ljubljana Ljubljana, Slovenia sg7710@student.uni-lj.si

#### 2 RELATED WORK

With the arrival of Microsoft's Windows 10 operating system, the company introduced collection of telemetry through a service called DiagTrack [1]. The files containing diagnostic and telemetry information were initially binary RBS files [1] but since Windows 10 update 1709, the data is instead collected into a simple database file [2]. The described approaches here work on Windows 11 as well, since it shares the same format.

In 2020, Han *et al.* [1] focused on the aforementioned older binary RBS file format, extracting and analysing various diagnostic information to aid in digital forensics investigations. Their research shows that the operating system and application's life-cycle, hardware information, history, etc. can all be acquired from Windows diagnostic data.

After Windows 10 update 1709, various contributions have been made to the research of the new data collection mechanism, one being Kroll's 2021 analysis by Rathbun *et al.* [4], where a deep dive of the EventTranscript database was conducted. Their findings are also available in an open source format on GitHub [3], which includes detailed information about the types of diagnostic data collected on Windows machines.

As previously mentioned, our paper is based on the recent 2022 study by Park and Lee [2], where the EventTranscript database was further analysed and helpful forensic information was extracted. Their efforts were focused on information about attached and detached USB devices, web browser activity, and wireless network activity. With the help of open wireless networks databases, they were also able to link logged networks with physical locations and consequently reconstruct a timeline of physical locations of the machine.

#### 3 WINDOWS LOGGING AND DIAGNOSTIC DATA

As pointed out in the original paper [2], the Windows Diagnostics service stores captured events on the operating system inside an SQLite3 database, which is stored on disk as the EventTranscript. db file. On Windows 10 and 11, the file is stored in C:/ProgramData/Microsoft/Diagnosis/EventTranscript folder, where C: is Windows' primary (installation) partition.

Table 1 presents important data tables available in the SQLite database, and also includes descriptions of their contents, as observed both by us and by the authors of the original paper. There also exist additional tables acting as a means for many-to-many relationships.

The table, storing the events we are most interested in, is the events\_persisted table. The most important columns of this table

Table	Description
categories	During our capture of the events on the system, this table remained unpopulated. Its attributes
	also do not hint at a potential use or meaning. The related research does not seem to discuss this
	table either.
events_persisted	This is where the diagnostic events are stored – this table is the main source of forensic information.
producers	Stores information about software that produced a specific event.
provider_groups	The internal meaning of this table is unknown to us; our guess is that it is similar to the producers
	table, in that it groups events together. In live data, this table was populated with several values,
	however their meaning was unclear, due to there being no apparent way to map group GUIDs to
	their meaning.
tag_descriptions	Contains tags and their descriptions. Here, we get a hint at what kind of data is collected in the
	database. For example, one of the entries describes Device Connectivity and Configuration.

Table 1: Tables in the EventTranscript.db database, and their roles (based on [4], [2], and our observations).

Attribute	Description
sid	ID of the Windows account associated with the event.
timestamp	Time of the event, stored as an LDAP timestamp: 100-nanosecond intervals since 1st
	January 1601.
full_event_name	The descriptive name of the event. We observed that it generally consists of a
	$reverse\ domain\ name-like\ notation,\ such\ as\ {\tt Microsoft.Windows.FeatureQuality.}$
	StateChange.
full_event_name_hash	Hash of the full_event_name column. Its value is also used as an ID in relation to
	event tags and categories the persisted event belongs to.
is_core	Indicates whether the process associated with the event is a core component of the
	Windows system.
provider_group_id	Correlates the event to entries in the provider_groups table.
logging_binary_name	File name of the process triggering the event.
<pre>friendly_logging_binary_name</pre>	Name of the process triggering the event.
producer_id	Correlates the event to entries in the producers table.
payload	Contains more detailed event information in text JSON format.

Table 2: Attributes of the events\_persisted table (based on [4] and [2]).

are described in table 2. It is important to note that the structure of the database makes virtually all attributes nullable, and therefore any value can be missing. In practice, however, almost every column did contain a value.

### 3.1 JSON Payload

The JSON payload, present in the payload column of the events\_ persisted table (see table 2), does not seem to have any particular scheme it fully adheres to. Fields and data vary between different events, but upon analysis of a live system, some of them seem consistent between events of the same kind. Based on the original paper and our own observations, most payloads share the following attributes:

- ver presumably the version of the JSON payload or its schema; 4.0 in our case.
- time timestamp the event was emitted, encoded as an ISO 8601 formatted UTC date and time.
- name the full name of the event in question; in most cases, its value is equal to the full\_event\_name column.
- data a JSON object, the contents of which depend on the event type. It usually contains some of the most useful

information about the event, which both *DiagAnalyzer* by Park *et al.* [2] and our own re-implementation focus on.

#### 3.2 Settings and Database Management

For the Windows Diagnostics service to create and maintain the EventTranscript.db database, the option *View Diagnostic Data* in the *Diagnostics and Feedback Settings* portion of the Windows Settings needs to be enabled [3] [4]. The database will eventually be created and populated. Note that it is possible to agree to this telemetry option while installing the operating system, which means an unsuspecting or uneducated user may not realise the data they are sharing.

For certain events, including those that we discuss in this paper, the optional telemetry described above needs to be enabled in the *Diagnostics and Feedback Settings* portion of the Windows Settings.

While the database gets updates very frequently, transactions are first accumulated in the EventTranscript.db-wal write-ahead logging file and eventually written to the storage.

Analysis of Windows 10 & 11 Logging Mechanisms for Use in Digital Forensics

#### 3.3 Application Events

The diagnostic events created by the DiagTrack service are quite extensive. Park *et al.* [2] described how attaching and detaching USB devices, using the Edge browser, and connecting to a wireless network, all generate events stored in EventTranscript.db. This may allow forensic experts to, in certain cases, even locate the machine's whereabouts by using Wi-Fi geolocation services, such as WiGLE [7], and the now-defunct Radiocells.org service they used in the paper.

As part of our re-implementation of an EventTranscript parser, which will be discussed in chapter 4, we took a deep dive into the various other events that the original paper did not discuss in-depth. We came up with the following events of potentially high forensic use:

- laptop battery level, and
- application activity.

In the following sections, we'll discuss both the events presented in the original paper, and our own findings during implementation of our own analysis tool.

3.3.1 Laptop Battery Level. If the operating system is running on a portable device with a battery, such as a laptop, we have found that a Microsoft.Windows.Kernel.Power.BatteryChargePercentageChange event is emitted each time the battery percentage changes by the battery either charging or discharging. Inside the JSON payload is encoded the current percentage (field data.RemainingPercentage).

While this event is unlikely to carry high-value forensic evidence by itself, it could very well be used by forensic professionals to correlate a multitude of related events together.

For example, if multiple evidence suggests the suspect was stationary for a duration at a certain time, and the battery data at those times shows charging activity, we can interpret this as the suspect charging their device at that place. Or, if the power level suddenly declines at a faster rate, a resource intensive process may have been executed on the device at that time.

3.3.2 **Detailed Application Activity**. While analysing various event transcripts from our sample data, we came upon a very significant and common event called Win32kTraceLogging.App InteractivitySummary. This event is emitted shortly after just about any graphical application is closed. Upon further analysis, we realised its forensic potential, because all of the following is available (and more):

- name of the executable and its SHA1 hash,
- approximate time the application was opened,
- time the application was closed,
- how long the user had the application in focus,
- how many times the user switched to a different application (number of lost focus events),
- width and height of the application window,
- how much time the user spent interacting with the application, including details on duration of mouse and keyboard input,
- if, and for how long, the user played audio using the application, and

• if, and for how long, the user captured (recorded) audio using the application, for example in an audio call application like Messenger, WhatsApp, or Discord.

In addition to the above, a significant amount of other secondary telemetry data is included, such as the user's monitor size, type of window, number of spawned processes, application version, etc.

*3.3.3* **USB Events.** When attaching or detaching a USB device, such as a USB storage device, a set of diagnostic events are generated in the event transcript. Some of them relate to their drivers being installed, while others relate to the actual disconnect event of the device. By carefully analysing the contents, researchers were able to extract the exact time of the connection, as well as the device manufacturer, device name, serial number, and the drive the storage is mounted on [2].

When a USB storage device is attached, events called Microsoft. Windows.Storage.Classpnp.DeviceGuidGenerated and Microsoft.Windows.Storage.StorageService.SdCardStatus are generated [2].

When a USB storage device is disconnected from the system, Microsoft.Windows.Storage.Classpnp.DeviceRemoved, Microsoft.Windows.FileSystem.NTFS.SurpriseRemove, and Microsoft.Windows.Storage.StorageService.SurpriseRemoval events are generated. The data included in these events allows forensic experts to determine with precision, which of the connected devices were removed, including their serial number [2], allowing for correlation with previous events.

*3.3.4* **Microsoft Edge Events**. Since the Edge web browser is integrated into the operating system, and Microsoft has taken several steps in order to have it be the user's default browser [6], it is no surprise that the analysis performed by Park *et al.* showed a heap of diagnostic and subsequently forensic data connected to the Edge browser.

Opening and using the Edge browser generates a significant amount of events. For example, when the Edge browser is opened, Aria.\*.Microsoft.WebBrowser.HistoryJournal.HJ\_BrowserInfo and Aria.\*.Microsoft.WebBrowser.HistoryJournal. HJ\_BrowserLaunchInfo are generated. When the user opens or closes a new tab, Aria.\*.Microsoft.WebBrowser.HistoryJournal.HJ\_TabCreated and Aria.\*.Microsoft.WebBrowser.HistoryJournal.HJ\_TabClosed are emitted, respectively.

Visiting a site emits an event with the name Aria.\*.WebBrowser.HistoryJournal.HJ\_BeforeNavigateExtended.Closing the browser generates the Aria.\*.Microsoft.WebBrowser.History Journal.HJ\_TabAllClosed event.

Many of these events contain additional data, which can be important to forensic investigation, such as temporal information about website visits, their underlying connection type, and the site being visited [2]. As we can see, the Edge browser is a treasure trove of forensic information.

*3.3.5* **Wireless Connection Events**. The Windows operating system logs several events related to wireless connections. We'll focus on two of them in this section, using research findings on this topic by Park *et al.* [2].

When scanning for nearby Wi-Fi access points, a WlanMSM.WirelessScanResults event is created. It contains both the SSID and the BSSID of all access points in the vicinity, allowing forensic investigators to correlate them with a physical location, given enough data.

When connecting to a specific Wi-Fi access point, a set of events are generated: Microsoft.OneCore.NetworkingTriage.GetConnected.WiFiStartConnectionAction, and Microsoft.OneCore. NetworkingTriage.GetConnected.WiFiConnectedEvent. If the user uses the Wi-Fi UI provided by the operating system, Micro soft.OneCore.NetworkingTriage.GetConnected.UXViewWi-FiConnectClickedEvent and Microsoft.OneCore.Networking Triage.GetConnected.UXModelWiFiProcessUserInputAction events are also created. The aforementioned events contain data about the status of the connection, and the SSID and BSSID of the access point.

When the user disconnects from a given Wi-Fi, Wlanvc.Wlan Disconnect is emitted. Just like the previous events, it contains information about the access point's SSID, BSSID, authentication algorithm, and cipher algorithm [2].

Out of all of the event types mentioned, wireless connection events are among the most useful, since the inclusion of SSID and BSSID data allows us to correlate them with physical locations. This can be done manually by the forensic team, or by using a public Wi-Fi geolocation service, such as WiGLE [7], or the nowdefunct Radiocells.org service Park *et al.* [2] mentioned in the paper. These services rely on community-submitted world-wide SSID and BSSID data, correlated with the recorded location of the submitter. Using triangulation, and by averaging many users' submissions, the approximate physical location of a Wi-Fi access point can often be recovered, proving the value of this kind of diagnostic data in a forensic investigation.

#### **4 OUR IMPLEMENTATION**

In their research paper, Park *et al.* proposed a Python tool called *DiagAnalyzer* for extracting forensic evidence off Windows 10 or 11 devices, revealing information about nearby Wi-Fi access points, USB connection events, and usage patterns of the Microsoft Edge browser. As part of our review, we re-implemented a similar tool, named *winspy*, which is available on GitHub under the GPLv3 licence at Grimpy101/FRI-2024-DF-seminar.

*Winspy* is a command-line tool, whose job is to analyse the contents of the EventTranscript.db SQLite3 database, and extract meaningful and useful forensic information. Because several events, described in chapter 3.3, had been already analysed by Park *et al.*, and included in their *DiagAnalyzer* tool, we instead analysed and proposed two new forensically-significant Windows diagnostic events: battery level (section 3.3.1) and application usage (section 3.3.2), which our tool extracts. Additionally, we implemented a simple browser-based visualisation tool for the generated forensic data.

In essence, the entire process, from receiving a disk image, to visualising the results, is made up of three steps (see figure 2):

• We first extract the relevant data from an image of the target computer - in our case this is the EventTranscript.db file (see chapter 3).

- Then, we use the *winspy* command-line tool to process the extracted diagnostic database and generate structured JSON output.
- Finally, we launch our browser-based visualisation tool (see figure 1) and load *winspy*'s output. We can now inspect the processed forensically-significant events that our tool is programmed to detect and describe.



Figure 1: A screenshot of the visualization tool for *winspy*. The elements in the list are labeled by the events. Clicking on an element reveals details about the event. The filters dropdown controls which types of events are shown.



# Figure 2: Analysis flow from obtaining the target system to analysing the contents of its diagnostic log.

The *winspy* tool itself is a Rust-based command-line tool utilising sqlx for SQLite support, and serde (and serde\_json) for JSON serialization. On a high level, *winspy* is made up of three components (see figure 3):

• The event transcript database is processed by an event reader implementation that produces strongly-typed event



Figure 3: Internal architecture of the winspy tool.

structures and links them to related event producers, categories, and other relations that are available in the database.

• The parsed events are forwarded to a set of stateful detectors, each of them specialised to one specific area of forensic information. In our case, we have two detector implementations: a battery level detector (see section 3.3.1) and an application activity detector (see section 3.3.2). Each detector receives each parsed event, processed it, and

subsequently emits zero or more events for the structured forensic report. The detector system is modular; as such, the software is easy to extend for additional event types and detectors.

• The report writer aggregates the processed events and writes them to an output file as JSON.

#### **5** FINDINGS

After developing our forensic tool named *winspy* (described in chapter 4), we put it to the test in a controlled environment.

We collected our EventTranscript.db samples from three separate machines: two static workstations and one virtual machine on a laptop, all using up-to-date versions of the Windows 10 operating system. The databases on all machines were largely consistent, although we noticed a slight inconsistency in the database table producers, where the column producer\_id\_name is instead called producer\_id\_text on some machines. We suspect this inconsistency comes from version mismatches between the systems, and updated our *winspy* software to support both formats.

We noticed that even with an elevated telemetry level, we were not able to collect all the events documented in the reference paper by Park *et al.* [2]. While some USB events were recorded, some parts of the mounting and unmounting chain of events were missing. When using Microsoft Edge, only events on opening and closing the program were stored, but not the events about tab management or visited sites.

Wireless connection event testing was done inside the virtual machine on a laptop in an attempt to get a larger sample of access points. A wireless USB adapter was used to connect to wireless networks inside the virtual machine. Unfortunately, no useful events were triggered to help us with our analysis.

We attribute this lack of logged data to various privacy and telemetry settings, the locality of our machines (we are under the EU law), and some hidden parameters regarding Microsoft's data sampling we likely had no control over. As such, we decided to focus on other significant diagnostic data that was available to us, as described in the implementation chapter.

#### 6 CONCLUSION

With the popularity and market share of Windows machines hovering above 70% [5], understanding Windows' diagnostic data is a high-value target for forensic investigators.

In our paper, we primarily reviewed work by Park and Lee [2], as well as introduced new information that was not covered in their work. We revisited existing approaches to analysing Windows diagnostic data, and covered the structure of the EventTranscript.db file and some of its various events.

Finally, we re-implemented the idea behind researchers' *DiagAnalyzer* in our own open-source Rust tool called *winspy*. Using our tool, and an additional simple in-house web-browser based visualisation application, we were able to determine, with a worrying accuracy, individual graphical application activity on the user's

machine, including information about their interactions with the software.

Reviewing Windows' DiagTrack service, and the EventTranscript.db database, served as a wonderful introduction into the space of Windows forensics. We have observed many other uncovered event types during our analysis, and as such, we expect the research into this topic to become even more detailed in the future.

#### REFERENCES

- Jaehyeok Han, Jungheum Park, Hyunji Chung, and Sangjin Lee. 2020. Forensic analysis of the Windows telemetry for diagnostics. arXiv:2002.12506 [cs.CR]
- [2] Sungha Park and Sangjin Lee. 2022. DiagAnalyzer: User behavior analysis and visualization using Windows Diagnostics logs. *Forensic Science International: Digital Investigation* 43 (09 2022), 301450. https://doi.org/10.1016/j.fsidi.2022. 301450

- [3] Andrew Rathbun. [n. d.]. GitHub AndrewRathbun/EventTranscript.db-Research: A repo for centralizing ongoing research on the new Windows 10/11 DFIR artifact, EventTranscript.db. — github.com. https://github.com/AndrewRathbun/ EventTranscript.db-Research. [Accessed 09-05-2024].
- [4] Andrew Rathbun, Josh Mitchell, and Devon Ackerman. [n. d.]. Diving Deeper into EventTranscript.db – kroll.com. https://www.kroll.com/en/insights/ publications/cyber/forensically-unpacking-eventtranscript/diving-deeper-intoeventtranscript. [Accessed 10-05-2024].
- [5] StatCounter. [n. d.]. Desktop Operating System Market Share Worldwide. https: //gs.statcounter.com/os-market-share/desktop/worldwide [Accessed 2024-05-12].
- [6] Tom Warren. [n.d.]. Microsoft is making it harder to switch default browsers in Windows 11. https://www.theverge.com/22630319/microsoft-windows-11default-browser-changes [Accessed 2024-05-12].
- [7] WiGLE. [n.d.]. WiGLE: Wireless Network Mapping. https://www.wigle.net/ [Accessed 2024-05-12].

Received 12 May 2024; revised 9 June 2024

# Forenzična analiza transakcijskih dnevniških zapisov SQL strežnika na nedodeljenem delu datotečnega sistema

Gregor Gabrovšek Fakulteta za računalništvo in informatiko, Univerza v Ljubljani Večna pot 113 Ljubljana, Slovenia gg4792@student.uni-lj.si

#### POVZETEK

Podatkovne baze so ključni del računalniških sistemov, ki hranijo pomembne informacije o poslovnih okoljih. Njihova uporaba se povečuje, s tem pa tudi število incidentov kot so poškodba podatkov, vdori in nenamerno razkritje podatkov. Ker so danes informacije bolj pomembne kot strojna oprema, je digitalna forenzika ključnega pomena. Za obnovo izgubljenih podatkov so pomembni predvsem dnevniški zapisi o transakcijah. Ti zapisi se običajno shranjujejo v dnevniških datotekah transakcij, ki se jih redno varnostno kopira (kopije se periodično brišejo zaradi prihranka prostora). Obravnava se problem pridobivanja izbrisanih dnevniških zapisov iz nedodeljenega pomnilnika. Z uporabo znanja o strukturi dnevnika transakcij MSSQL ter analize nedodeljenega pomnilnika je mogoče identificirati in delno obnoviti izbrisane zapise. Eksperimenti na izmišljenih podatkih in realnih podatkih so pokazali uspešnost pristopa, ki se je izkazal kot obetaven za forenzične preiskave na področju podatkovnih baz.

#### Ključne besede

digitalna forenzika, transakcijski dnevniški zapisi SQL, nedodeljeni del datotečnega sistema, rekonstrukcija podatkov

#### 1. UVOD

SQL podatkovne baze so pomemben del marsikaterega računalniškega sistema. Sami podatki v bazi pa niso edina stvar, ki jo podatkovna baza običajno hrani. Poleg samih podatkov je pomembno hraniti tudi dnevniške zapise o dogajanju v sistemu.

V teh dnevniških zapisih pa se ne hranijo le npr. časovni žigi zagona baze, temveč se ob pravih nastavitvah baze hranijo tudi informacije o operacijah na bazi. Te informacije lahko vključujejo časovne žige operacij ali celo samo vsebino operacij. Martin Dagarin Fakulteta za računalništvo in informatiko, Univerza v Ljubljani Večna pot 113 Ljubljana, Slovenia md4212@student.uni-lj.si

Tako podrobni dnevniški zapisi o operacijah, kot so na primer INSERT, DELETE in UPDATE, nam omogočajo, da lahko ugotovimo stanje vsebine baze za poljubno točko v relativno nedavni zgodovini.

Težava pa je, da so ti dnevniški zapisi načeloma na voljo le za omejen čas, zato je za potrebe forenzične preiskave potrebno najti pristop, kako dostopati do informacij tudi po tem, ko so dnevniški zapisi že izbrisani. Dostop do čim več dnevniških zapisov je ključno opravilo za zbiranje dokaznega gradiva, zato je pomembno raziskati tehnike za pridobivanje težje dostopnih zgodovinskih dnevniških zapisov.

V članku bomo v poglavju 2 opisali ozadje problema in zakaj je forenzika na podatkovnih bazah smiseln pristop. V poglavju 3 bomo našteli dosedanje pristope na tem področju, v poglavju 4 pa bomo opisali najmodernejši forenzični pristop na področju preiskave izbrisanih dnevniških zapisov. Nazadnje pa bomo v poglavju 5 opisali uspešnost opisane metode na realnem in umetnem scenariju.

#### 2. OPIS PROBLEMA

Strežniki SQL zabeležke najpogosteje zapisujejo v datoteke. Te datoteke vsebujejo strukturiran zapis dnevniških zapisov, običajno pa so zapisi razdeljeni v več datotek glede na neko časovno obdobje (ena datoteka dnevniških zapisov na primer vsebuje zapise enega dne, ko je bila baza vzpostavljena).

Ker pa so posamične podatkovne baze neredko vzpostavljene na enem strežniku leta in leta, bi se lahko zgodilo, da bi že same datoteke dnevniških zapisov zapolnile celoten prostor na disku. Velike datoteke dnevniških zapisov lahko tudi povzročijo, da zagon podatkovne baze traja dlje. Zato je pri večini podatkovnih baz moč nastaviti konfiguracijsko spremenljivko, ki določa življenjsko dobo oziroma obdobje hranjenja zapisov. To lahko običajno spreminja tudi upravljalec baze sam, vrednost pa nastavi glede na lastne potrebe oz. glede na oceno, kako davno zgodovino baze želi pregledovati z uporabo dnevniških zapisov (zapisi lahko pridejo prav v primeru, da se baza poruši ali v primeru okvare podatkov).

Ko je življenjska doba dnevniškega zapisa pretekla, se datoteka z zapisom običajno pobriše. To pa še ne pomeni, da je podatek za vekomaj izgubljen. V primerih, ko je pomembno pridobiti že izbrisane dnevniške zapise, se lahko poslužimo raznih metod za pridobitev teh podatkov. Glede na tip diska in datotečnega sistema obstajajo različni pristopi, kako povrniti vsebino dnevniških zapisov. Če je bila datoteka nedavno izbrisana in se še cela nahaja na disku, je postopek obnovitve datoteke moč opraviti s standardnimi pristopi obnavljanja datotek. V primeru, ko pa je npr. datotečna glava že izbrisana, pa lahko ta postopek postane mnogo težji.

Ta problem želimo nasloviti tako, da izrabimo pomembno lastnost datotek z dnevniškimi zapisi. Zabeležke so v datotekah namreč shranjene v posebnem formatu, ki ga je moč prepoznati in razbrati. Na ta način je mogoče rešiti vsaj nekatere podatke in do neke mere dobiti vpogled v pretekle operacije, ki so bile izvedene na podatkovni bazi.

#### 3. SORODNA DELA

Obnavljanje starih dnevniških zapisov je že znan problem. Področje je razmeroma široko, saj obstaja mnogo različnih tipov podatkovnih baz, vsak s svojimi posebnostmi, zato je prostora za raziskave veliko.

Fowler (2008) v knjigi opisuje mnoge tehnike, ki pridejo prav pri forenzičnem delu, povezanem s podatkovnimi bazami tipa SQL [6]. Opiše veliko pristopov, ki pomagajo pri potrjevanju, ali je prišlo do vdora, luščenju dokaznega gradiva iz lažje in težje dostopnih delov podatkovne baze, predstavi pa tudi realne primere forenzične analize podatkovnih baz.

Chopade in Pachghare (2019) sta opravila temeljit pregled celotnega področja forenzičnih preiskav na podatkovnih bazah [5]. Osredotočila sta se tako na relacijske, kot tudi na NoSQL podatkovne baze. Opisala sta zgodovino od nastanka področja, kar se je začelo okoli leta 2007. Predstavila sta ključne korake pri forenzični preiskavi podatkovne baze, popisala vse vrste in vire dokazov, ki jih je mogoče najti v sistemu, kjer se nahaja podatkovna baza, opravila pa sta tudi primerjavo podatkovnih baz iz vidika forenzike. Navedla sta, da se je večina raziskav do sedaj osredotočala na relacijske podatkovne baze, prihodnost raziskav pa vidita na bazah tipa NoSQL, kjer še obstajajo priložnosti za nova odkritja.

Choi in Lee (2023) sta se osredotočila na podatkovno bazo Microsoft SQL Server, ki je priljubljena izbira razvijalcev računalniških sistemov [4]. V svojem delu opišeta strukturo dnevniških zapisov podatkovne baze MSSQL, predstavita pa tudi svojo metodo, ki omogoča pridobivanje starih dnevniških zapisov iz nedodeljenega dela datotečnega sistema, kjer so se nekoč nahajale datoteke z zapisi, potem pa so bile izbrisane.

### 4. METODOLOGIJA

Naš cilj je torej odkriti čim več ostankov dnevniških zapisov MSSQL podatkovne baze na nedodeljenem delu datotečnega sistema. Da lahko to storimo, moramo najprej poznati format zapisov, nato pa moramo v ostankih podatkov identificirati tak format, podatke razčleniti in nato vrniti obnovljene zapise.

#### 4.1 Format dnevniških zapisov baze MSSQL

Podatkovna baza MSSQL ustvarja več različnih tipov datotek z zabeležkami. Ta, ki nas za naš primer uporabe najbolj zanima, pa je datoteka z dnevniškimi zapisi o transakcijah (angl. *transaction log file*).



Slika 1: Struktura virtualnih dnevniških datotek transakcij MSSQL [4].

			FileSize				FSeqNo			StartOffset						
Offset(h)	00	01	02	03	04	05	06	<sup>1</sup> 07	08	09	0A	ов	0C	0D	0E	OF
00002000	AB	40	05	00	25	00	00	00	00	00	00	00	01	00	00	00
00002010	00	E0	03	00	00	00	00	00	00	20	00	00	00	00	00	00
00002020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00002030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00002040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00002050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00002060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	Cre	CreateLSN(4byte:4byte:2byte)														

#### Slika 2: Blok znotraj navidezne dnevniške datoteke transakcij MSSQL [4].

Same datoteke z zapisi o transakcijah so razdeljene na več datotek [7]. Med transakcijske dnevniške zapise spadajo zapisi o sledečih operacijah:

- INSERT, UPDATE in DELETE operacije, ki vstavljajo, posodabljajo, ali brišejo posamezno vrstico v neki tabeli
- CREATE in DROP operaciji, ki ustvarjata ali brišeta same tabele

Poleg tega pa se dnevniški zapis ustvari tudi o začetku in koncu posamezne transakcije.

MSSQL razdeli dnevniške zapise transakcij v več navideznih datotek (Virtual Log Files - VLF), ki jih v datotečnem sistemu vidimo kot eno samo datoteko (slika 1). Navidezna datoteka je razdeljena v odseke po 512B, ki jih imenujemo bloki. Velikost dnevniških datotek ni omejena (razen omejitev datotečnega sistema), prav tako pa ni omejeno število dnevniških datotek. Teh vrednosti ni možno nastaviti, saj za optimalno velikost in število datotek skrbi MSSQL.

Blok (slika 2) znotraj navideznih dnevniških datotek se začne s paritetnim bajtom, ki ob inicializaciji vsebuje vrednost 64, kasneje pa se njegova vrednost ob ponovni uporabi izmenično spreminja v 64 ali 128. Paritetnemu bajtu sledi glava, ki vsebuje zaporedno številko dnevniške datoteke, velikost datoteke ter odmik začetka podatkovnega dela v datoteki. Za glavo sledijo podatki, ki so shranjeni znotraj bloka.

Informacije o navideznih datotekah dnevniških zapisov transakcij podatkovne baze lahko dobimo z ukazom *dbcc loginfo*, ki izpiše sledeče atribute:



Slika 3: Vizualizacija strukture bloka dnevniških zapisov [4].

- RecoveryUnitId
- FileId
- FileSize velikost virtualne dnevniške datoteke
- StartOffset odmik od začetka podatkovnega dela bloka
- FSeqNo zaporedna številka virtualne dnevniške datoteke
- Status označuje ali je virtualna dnevniška datoteka v uporabi ali ne
- Parity pariteta virtualne dnevniške datoteke
- CreateLSN zaporedna številka dnevniškega zapisa

Da lahko iščemo dnevniške zapise na nedodeljenem delu diska, kjer nimamo točnega podatka, kje natančno se nahaja datoteka, pa moramo poznati strukturo, s katero so zapisi zapisani v datoteko.

Posamezni zapisi so v datoteko zapisani kot del bloka dnevniških zapisov (angl. *log block*). Ta je sestavljen iz glave segmenta (velik 72 bajtov), samih dnevniških zapisov, seznama odmikov dnevniških zapisov in iz paritetnih bitov, kot je prikazano na sliki 3.

Odmiki so shranjeni v obrnjenem vrstnem redu, to pa pomeni, da se prvi vnos nahaja na koncu seznama odmikov. Odmik je shranjen kot vrednost velikosti 2 bajta. V blokih delov dnevniških zapisov je v paritetnem bajtu poleg paritete zapisano tudi ali gre za prvi blok dnevniškega zapisa (5. bit) ali za zadnji blok dnevniškega zapisa (4. bit), zato vrednost v prvem bajtu bloka ni nujno 0, 64 ali 128. Originalna vrednost paritete bloka je shranjena v prvem bajtu, kjer je shranjen blok dnevniškega zapisa transakcije.

Dnevniški zapis posamezne transakcije je sestavljen iz skupine dnevniških zapisov, ki so shranjeni v obliki blokov. S pomočjo sistemske funkcije *fn\_dblog()* lahko dostopamo do dela datoteke dnevnika transakcij, ki se še vedno spreminja (je aktiven). Z opazovanjem omenjenega dela lahko dobimo vpogled v spreminjanja trenutnega stanja podatkovne baze. Vsi posamezni dnevniški zapisi se začnejo z fiksnim delom velikosti vsaj 24 bajtov, preostali del zapisa pa se razlikuje glede na tip operacije, ki je bil izveden. Obravnavan članek [4] se osredotoča na sledeče operacije:

- vstavljanje vrstic
- sprememba vrstic
- brisanje vrstic
- začetek ter konec transakcije

Celotno velikost fiksnega dela zapisa določa polje Log Record *Fixed Length*, ki pa je odvisen od tipa operacije, ki jo opisuje posamezen zapis. Zapis vsebuje tudi informacijo o verigi dnevniških zapisov, ki ji pripada, ter identifikator transakcije, ki jo zapis opisuje. V zapisu je shranjena tudi lokacija podatkov, na katere se posamezni dnevniški zapis nanaša (datoteka, tabela, vrstica, odmik v vrstici) ter velikost spremenjenih podatkov. Podatkovni del zapisa pri vstavljanju vsebuje vstavljeno vrednost podatkov, pri brisanju staro vrednost podatkov, pri urejanju pa zapis vsebuje dva podatkovna dela: prvi vsebuje staro vrednost podatkov, drugi del pa novo vrednost. Zapis, ki oznanja začetek ali konec transakcije, poleg ostalih informacij vsebuje tudi čas začetka in konca transakcije ter čas izvajanja. Čas je zapisan z 8 bajti, kjer prvi 4 bajti določajo število dni od datuma 1. 1. 1900, drugih 4 bajtov pa število period (1/300 sekunde) od polnoči.

#### 4.2 Pridobivanje dnevniških zapisov

Članek [4] se osredotoča na pridobivanje dnevniških zapisov iz nedodeljenega dela pomnilnika, kjer so ostali pretekli zapisi. To pa se zgodi v dveh primerih:

- 1. ko se velikost datoteke dnevnika transakcije zmanjša
- 2. ko je datoteka, ki vsebuje varnostno kopijo dnevnika transakcij, izbrisana

V prvem primeru se velikost datoteke zmanjša v naslednjih primerih:

- izvedemo ukaz za izdelavo varnostne kopije dnevnika transakcij
- omogočimo enostaven obnovitveni način
- nikoli ne naredimo polne varnostne kopije podatkovne baze

Kljub temu, da bi večina ljudi predpostavljala, da v prvem primeru podatki ostanejo na delu, kjer je bila včasih datoteka dnevnika zapisov transakcij, so tam podatki ob zmanjšanju datoteke ponastavljeni in s tem za vedno izbrisani. To pomeni, da so vsi dnevniški zapisi, ki se nanašajo na transakcije pred zadnjo aktivno transakcijo, označeni za ponovno uporabo (na mesto teh zapisov se lahko shrani nov zapis, ker ni več starih podatkov).
Drugi primer pa se nanaša na varnostne kopije dnevnika transakcij, ki jih MSSQL izdeluje, da zavaruje podatke pred nezaželenimi dogodki ter nam omogoči obnovitev podatkov. Ker pa hramba varnostnih kopij zaseda prostor v disku naprave, podatkovna baza periodično briše varnostne kopije dnevnika transakcij. Del podatkov izbrisane varnostne kopije ostane na mestu, kjer je bila shranjena datoteka, zato lahko na tem delu opravimo analizo in obnovimo del dnevnika transakcij ter posledično tudi transakcij. V okviru članka je bila opravljena analiza zgolj na podlagi oblike, saj je poudarek na dnevniških zapisih.

Analiza oblike je razdeljena na:

- 1. identifikacijo zapisov, ki se nahajajo znotraj dnevniškega zapisa transakcij
- 2. rekonstrukcijo poizvedbe

Identifikacija zapisov je sestavljena iz sledečih korakov:

- 1. pridobitev (nedodeljenega/sproščenega) dela pomnilnika
- preiskava, kjer se išče podpis operacije, ki se nahaja v prvih štirih bajtih posameznega bloka. Podpis vsake možne operacije ima več možnih vrednosti, saj je potrebno upoštevati tudi paritetni bajt, ki lahko zavzema različne vrednosti
- 3. dodajanje zapisov na seznam, kjer se podpis ujema

Sledi rekonstrukcija poizvedbe, ki je enostavna za operacije vstavljanja, urejanja, brisanja, saj jo lahko izvedemo na podlagi analize enega dnevniškega zapisa, medtem ko je rekonstrukcija poizvedb, ki ustvarijo ali brišejo tabele, kompleksna, saj je za rekonstrukcijo teh poizvedb potrebno več dnevniških zapisov.

Rekonstrukcijo poizvedbe začnemo z *analizo oblike zapisa*, kjer je potrebno izračunati tudi dolžino zapisa, da lahko naložimo vsebino zapisa iz pomnilnika. Sledi *analiza oblike vsebine* za katero pa je potrebna tudi datoteka s podatki, saj je za analizo oblike vsebine potrebno poznati tudi informacije o tabelah v podatkovni bazi. Na podlagi uspešnih analiz lahko iz obnovljenih dnevniških zapisov razveljavimo učinke vstavljanja ali brisanja posameznih vrstic v tabeli. V primeru, da želimo razveljaviti učinek poizvedbe, ki je posodobila vrednosti, pa potrebujemo tudi podatkovno datoteko, da je obnovitev prejšnjih vrednosti mogoča, saj dnevniški zapis vsebuje zgolj spremenjene vrednosti.

Avtorji članka [4] so v programskem jeziku Python implementirali naslednja modula:

- 1. Za analizo datoteke dnevniških zapisov transakcij (končnica .ldf), ki lahko skupaj z datoteko s podatki (končnica .mdf) obnovi podatke prejšnjih poizvedb.
- Za preizkus metode zaznavanja zapisov na nedodeljenem/sproščenem delu pomnilnika in analizo strukture zaznanih zapisov ter obnovitvijo poizvedb s pomočjo datoteke s podatki.

### 5. REZULTATI

Za pridobitev rezultatov so avtorji članka uporabili dva nabora podatkov:

- nabor izmišljenih podatkov, ki so jih avtorji ustvarili in prilagodili postopku omenjenemu v članku
- podatki iz strežnika, kjer se je v času tekel MSSQL strežnik

Nad izmišljenimi podatki sta bila izvedena 2 eksperimenta in sicer:

- 1. zaporedje 1000 vstavljanj vrstic (po 3 naenkrat), 1000 posodobitev vrstic in 1000 brisanj vrstic
- 2. uporaba navideznega nosilca, na katerem se je izvedlo sledeče zaporedje operacij: izdelava varnostne kopije (celotne baze, dnevnika transakcij), 1000 vstavljanj vrstic, izdelava varnostne kopije dnevnika transakcij, 1000 vstavljanj vrstic, izdelava varnostne kopije dnevnika transakcij, 1000 vstavljanj vrstic, izdelava varnostne kopije dnevnika transakcij, posodobitev 1000 vrstic, brisanje 1000 vrstic, izdelava varnostne kopije (celotne baze, dnevnika transakcij), brisanje prve in tretje varnostne kopije dnevnika transakcij, pridobitev pomnilnika izbrisanih datotek z orodjem EnCase (OpenText Encase Forensic [2])

Z eksperimentom 2 je bilo simulirano obnašanje periodičnega brisanja starih datotek dnevniških zapisov.

Na izmišljenih podatkih je implementiran modul uspešno identificiral in obnovil skoraj vse zapise. Uspešnost prepoznave zapisov implementiranega modula na izmišljenih podatkih je bila 95%. Rezultati so bili preverjeni tudi s komercialnimi orodji ApexSQL Recover [1] in SQL Transaction Log Analyzer Tool [3].

Realni podatki za analizo so bili pridobljeni na MSSQL strežniku, ki je bil v času nastanka članka v uporabi približno 8 let. Na strežniku je potekala redna izdelava varnostnih kopij, ki so bile periodično izbrisane, ko jim je potekel čas veljavnosti. Zajem podatkov je potekal dva tedna v dnevnem in urnem intervalu. Zajete podatke se je hranilo na 2 TB velikem nosilcu. Podatke za analizo se je iz nosilca pridobilo s pomočjo orodja EnCase [2] (1,18 TB). Te se je skupaj z datoteko podatkov podatkovne baze uporabilo za analizo z implementiranim modulom. V tem primeru je bilo število obnovljenih poizvedb zelo majhno v primerjavi s številom zaznanih podpisov operacij. To je posledica velike količine podatkov in dolge uporabe MSSQL strežnika zaradi katere so podatki porazdeljeni po prostoru.

Pri analizi so se pri implementiranem postopku obnove poizvedb pokazale naslednje omejitve:

• Zaradi poudarka na analizi posameznih zapisov dnevniškega zapisa transakcije, se ni preverjalo paritete skupine zapisov.  Če so bili zapisi porazdeljeni zaradi omejitve datotečnega sistema (presežena omejitev velikosti enote datotečnega sistema), je bila obnova zapisa težka.

# 6. ZAKLJUČEK

Izpostavljena je pomembnost dnevniških zapisov transakcij v relacijskih podatkovnih bazah ter izzive, povezane z obnovitvijo izbrisanih dnevniških zapisov transakcij za namen forenzične preiskave. Uporaba podatkovnih baz za hrambo poslovnih podatkov se konstantno povečuje, posledično prihaja tudi do več incidentov kot so izguba podatkov, poškodba podatkov ... Sistem za upravljanje podatkovnih baz ustvari več tipov dnevniških zapisov (transakcij, napak, delovanja sistema ...), ampak je zaradi omejitve obsega fokus na metodi za pridobivanje težje dostopnih zgodovinskih dnevniških zapisov transakcij, ki se nahajajo na področju nedodeljenega dela diska (področje izbrisanih datotek). Poudarek je na analizi dnevniških zapisov in rekonstrukciji transakcij sistema za upravljanje s podatkovno bazo MSSQL predvsem na zapisih operacij vstavljanja, urejanja in brisanja podatkov, saj so te zapisane zgolj v enem posameznem dnevniškem zapisu.

Rezultati raziskave vključujejo uspešnost identifikacije in obnove na izmišljenih podatkih in realnih podatkih, kjer je bil uspeh manjši zaradi kompleksnosti in obsega podatkov. Pokazale so se tudi omejitve pri obnovi zaradi fokusa na posamezni zapis ter omejitve datotečnega sistema.

# 7. VIRI

- [1] Apex sql. Dostopno na https://www.apexsql.com/sql-tools-recover/.
- [2] Opentext encase forensic. Dostopno na https://www.opentext.com/products/encase-forensic.
- [3] Sql transaction log analyzer tool. Dostopno na https://www.stellarinfo.com/mssql-log-analyzer.php.
- [4] H. Choi and S. Lee. Forensic analysis of sql server transaction log in unallocated area of file system. *Forensic Science International: Digital Investigation*, 46:301605, 2023.
- [5] R. Chopade and V. K. Pachghare. Ten years of critical review on database forensics research. *Digital Investigation*, 29:180–197, 2019.
- [6] K. Fowler. SQL server forenisc analysis. Pearson Education, 2008.
- [7] R. West. Sql server transaction log architecture and management guide, May 2023.



# 5 - Forenzika gesel in šifriranje / Password Forensics and Cyphering

Izzivi

Študij

Ustvarjanje novih svetov

Raziskovanje

# Forenzična analiza dvofaktorske avtentikacije v nekaterih aplikacijah

Karmen Frank Fakulteta za računalništvo in informatiko Večna pot 113 Ljubljana, Slovenija kf7941@student.uni-lj.si

# POVZETEK

V seminarski nalogi predstavimo koncept dvofaktorske avtentikacije in njeno uporabo. Naštejemo različne načine njene implementacije in se poglobimo v dvofaktorsko avtentikacijo (2FA) s TOTP. Opišemo rezultate predhodnih člankov na temo iskanja tajnih ključev, varnejše implementacije 2FA in analize različnih 2FA aplikacij. Osredotočimo se na delo v članku o analizi 2FA aplikacij. Predstavimo njihove rezultate in sklepe. Več različnih 2FA aplikacij ima enak algoritem za izračun TOTP, kar je potencialno koristno za delo digitalnih forenzikov. Na koncu izvedemo še lasten test obhoda TOTP in dokumentiramo svoja opažanja. Ugotovimo, da se ujemajo z opažanji izhodiščnega članka.

#### Ključne besede

Dvofaktorska avtentikacija, digitalna forenzika, artefakti, OTP, tajni ključi

# 1. UVOD

Dvofaktorska avtentikacija (kratica 2FA) je varnostna metoda, ki za dostop do virov in podatkov zahteva identifikacijo v dveh oblikah. Prvič so jo za komercialne namene uporabili leta 1986, ko je podjetje RSA razvilo elektronske obeske za ključe. Obesek je izpisal kratko kodo iz števk, ki so jo uporabniki uporabili poleg svojega gesla. Uporaba dvofaktorske avtentikacije se je začela širiti predvsem po letu 2010 [10], največjo rast uporabe pa je dosegla med letoma 2017 in 2021, ko je odstotek uporabnikov 2FA zrasel za 51% [3].

Podatki s Statiste [13] kažejo, da spletne storitve uporablja 5,35 milijard uporabnikov, od tega 5,04 milijard oseb uporablja tudi socialna omrežja. Z rastjo števila oseb, ki uporabljajo splet, se je povečalo tudi število kibernetskih napadov. Med letoma 2017 in 2022 je bilo vloženih 3,26 milijonov pritožb v povezavi s kibernetskim kriminalom [5], zgolj v letu 2023 pa več kot 880.000 pritožb [6]. Nepridipravi pogosto napadajo uporabniške račune in jih skušajo ugrabiti (angl. account hijacking). To lahko dosežejo z uporabo

Eva Jug Fakulteta za matematiko in fiziko Jadranska ulica 19 Ljubljana, Slovenija ej57750@student.uni-lj.si

različnih vrst napadov, kot so phishing, ugibanje šibkega gesla, socialni inženiring idr. [15]. Ugrabitev uporabniškega računa omogoči napadalcu dostop do podatkov, financ in identitete žrtve. Pred tem nas lahko do določene mere zaščiti dvofaktorska avtentikacija.

Dvofaktorska avtentikacija se najpogosteje pojavlja v kombinaciji uporabniškega imena in gesla s sledečo drugo stopnjo avtentikacije. Pogosto se uporablja še eno napravo, naše biometrične podatke ali pa lokacijo, na kateri se nahajamo (manj uporabljan pristop). Primer: na mobilni telefon ob prijavi z imenom in geslom prejmemo SMS sporočilo z enkratno kodo. Nekatere spletne strani pa recimo zahtevajo, da z ustrezno aplikacijo na mobilnem telefonu skeniramo QR kodo [7].

Napogostejše implementacije 2FA vsebujejo strojne žetone (angl. hardware tokens), potisna obvestila (angl. push notifications), verifikacijo preko SMS sporočil, U2F varnostne ključe, časovno odvisna enkratna gesla (angl. time-based one time passwords, TOTP) idr. [7, 11]. Strojni žetoni so običajno majhne naprave, npr. pametna kartica ali elektronski obesek za ključe, ki imajo majhen spomin, v katerem hranijo certifikat oz. identifikator. Potisna obvestila na pametnem telefonu omogočijo uporabniku, da potrdi oz. zavrne vpis v njegov uporabniški račun. Verifikacija preko SMS sporočil poteka tako, da uporabnik na mobilno napravo prejme SMS sporočilo z enkratno verifikacijsko kodo, ki je običajno sestavljena iz šestih števk, nato pa jo vpiše v aplikacijo, v katero se prijavlja. Težava tega pristopa je, da SMS sporočila med prenašanjem niso kodirana, kar omogoča napad moža v sredini. U2F (angl. universal second factor) je standard za avtentikacijo z USB ključem, ki ga vključimo v računalnik, na katerem se prijavljamo [14].

Ob vzpostavitvi 2FA s TOTP aplikacija dodeli uporabniku tajni ključ. Le-ta je shranjen tudi na strežniku. Verifikacijska koda, ki je običajno veljavna 30 s, se ob vsaki prijavi generira upoštevajoč tajni ključ in časovni žig. Izračun je opravljen posebej na uporabnikovi in na strežnikovi strani, zato ni potrebe po prenašanju kode po omrežju [14]. V primeru da naprava, preko katere je imel uporabnik vzpostavljen TOTP, ni na voljo, je ta način dvofaktorske avtentikacije mogoče zaobiti.

V poglavjih 3 in 4 se bomo osredotočili predvsem na artefakte, ki so jih izbrane aplikacije za dvofaktorsko avtentikacijo pustile za sabo, kot so to beležili v članku [1]. V poglavju 5 bomo opisali predlog za obhod TOTP. Potem pa se bomo v naslednjem poglavju poglobili še v praktične poskuse po zgledu članka.

# 2. SORODNA DELA

Dvofaktorsko avtentikacijo potrebujemo ob prijavljanju v Googlove storitve, na socialna omrežja, razne spletne trgovine idr. 2FA nam omogoča višjo stopnjo varnosti, kot če bi uporabljali zgolj geslo, ni pa odporna na kibernetske napade [1]. V poglavju 2.2 opišemo nekatere izmed tovrstnih napadov.

V članku [1] so se osredotočili na iskanje tajnega ključa oz. semena, s katerim se generira verifikacijska koda. Večina drugih izpostavljenih del v članku raziskuje predvsem o razširjeni uporabi dvofaktorske avtentikacije.

# 2.1 Najdeni artefakti v sorodnih delih

Z iskanjem tajnih ključev, ki generirajo enkratna gesla, so se ukvarjali tudi v [12]. Osredotočili so se na naprave z operacijskim sistemom Android in aplikacije za mobilno avtentikacijo. Skušali so pridobiti tajni ključ oz. seme v nekodirani obliki (angl. cleartext). Z uporabo obratnega inženiringa jim je uspelo pridobiti semena petih aplikacij iz notranjega pomnilnika in sedmih iz delovnega pomnilnika. Izpostavili so, da so informacije bolje zašifrirane v notranjem pomnilniku, medtem ko naj bi bil delovni pomnilnik težje dostopen za strokovnjake obratnega inženiringa.

Nasprotno, pa so v drugem članku [17] predlagali dvofaktorsko avtentikacijo, ki bi bistveno omejila možnosti analiziranja sledi prenosa informacij. Namreč, večina 2FA shem je osnovanih na zahtevnosti t.i. težkih problemov. Ker pa se kvantni računalniki (in tudi kvantni algoritmi) dandanes vse hitreje razvijajo, težki problemi niso več dovolj zahtevni za zaščito informacij. Zato so v članku predlagali Quantum2FA shemo, ki bi bila odporna na napade s kvantnim računalnikom. Osnovana je na avtentikaciji s pametno kartico. V shemo so vpeljali mrežno izmenjavo ključev in tehniko verifikacije [16], ki se že uporablja na področju kriptografije in varnosti [8, 4, 2].

# 2.2 Hekanje večfaktorske avtentikacije

2FA je specifičen primer večfaktorske avtentikacije (kratica MFA), ki zahteva vsaj dve stopnji avtentikacije. V [9] so predstavili več različnih načinov, kako zaobiti oz. shekati MFA. Pristopi obsegajo socialni inženiring, iskanje napak v implementaciji MFA, postavljanje lažnih spletnih strani, s katerimi ukradejo uporabnikove podatke za prijavo, ugrabitve seje idr. Nekaj primerov je opisanih v nadaljevanju.

Ko se uporabnik uspešno prijavi na spletno stran, mu je dodeljen unikaten sejni žeton ne glede na to, ali se je prijavil le z geslom ali uporablja tudi 2FA/MFA. Pri napadu z ugrabitvijo seje napadalec pridobi ta žeton (npr. s phishing e-pošto) in se lahko, dokler seja traja, pretvarja, da je uporabnik.

Uporabnikov računalnik se lahko okuži s trojanskim konjem, ki išče bančne podatke. Trojanski konj zaznava ključne besede v povezavi z bančništvom in tako prepozna, ko se uporabnik prijavlja v svoj bančni račun. Ko se uporabnik uspešno prijavi v bančni račun, se v ozadju požene še ena seja, preko katere lahko virus dostopa do bančnega računa, spreminja kontaktne informacije in prosto upravlja s sredstvi. Pred tem napadom se banke trudijo zaščititi z dodatnim potrjevanjem transakcij s potrditvenimi kodami.

Napadalec lahko pridobi novo SIM kartico za določenega uporabnika (angl. SIM swap) in jo aktivira v svojem mobilnem telefonu. Na ta način napadalec dobi dostop do vseh SMS sporočil, kar lahko izkoristi za prijavo v izbran žrtvin uporabniški račun, ki uporablja večfaktorsko avtentikacijo z SMS sporočili.

Socialni inženiring izkorišča človeški element v neki situaciji napram lastnostim elektronskih naprav. Gre za izredno preprosto in cenovno ugodno obliko hekanja. Lahko je celo tako preprost, da napadalec uporabniku "gleda čez ramo" in na tak način npr. ujame PIN uporabnika.

# 3. METODOLOGIJA

Forenzična preiskava, opisana v [1], je bila izvedena nad petnajstimi 2FA aplikacijami. Navedene so v Tabeli 1, kjer lahko najdemo tudi podatek o tem, za kateri operacijski sistem so namenjene. V preiskavi so uporabili pametni telefon Galaxy S6 (Android), ki je bil odklenjen/zlomljen s postopkom, ki ima v angleškem jeziku izraz "rooting". Uporabljen je bil prav tako odklenjen/zlomljen iPhone 7 (iOS) in virtualni računalnik z operacijskim sistemom Windows 10.

Aplikacije so bile pridobljene v trgovini Microsoft Store<sup>1</sup>, Google Play<sup>2</sup>, na platformi Github<sup>3</sup>, na spletni strani Authy<sup>4</sup> in v trgovini Apple App Store<sup>5</sup>. Pripadajo tako velikim organizacijam kot tudi manjšim in se razlikujejo po številu prenosov oz. uporabnikov. Storitve, nad katerimi je bila uporabljena 2FA avtentikacija z zgoraj omenjenimi aplikacijami, so bile Dropbox, Facebook, Instagram, Snapchat in Twitter (sedaj X).

Postopek testiranja je bil sledeč. 2FA aplikacije so bile najprej nameščene na naprave, ki jih podpirajo. Aplikacija Twilio Authy je za vzpostavitev potrebovala tudi telefonsko številko. Po potrebi je bilo ustvarjeno geslo oz. pin številka, s katero se zaklene aplikacija in v nekaterih primerih tudi uporabi za šifriranje vseh podatkov v povezavi z aplikacijo v napravi. Testirani sta bili obe različici tovrstnih aplikacij, torej brez in z geslom. V storitvenih aplikacijah (navedena 4 socialna omrežja in storitev za shranjevanje podatkov v oblaku) je bila omogočena dvofaktorska avtentikacija in izbrana možnost uporabe aplikacije za avtentikacijo.

Storitvena aplikacija je nato generirala tajni ključ, ki ga je bilo treba vpisati v 2FA aplikacijo, da se je lahko povezala z uporabniškim računom. V nekaterih primerih je bilo vnešeno tudi ime uporabnika in/ali ime storitve. Z vzpostavljeno povezavo med storitvijo in 2FA aplikacijo, je lahko slednja generirala kodo iz šestih števk. Z vpisom kode v storitev je bila prijava v uporabniški račun opravljena.

<sup>&</sup>lt;sup>1</sup>https://apps.microsoft.com/

<sup>&</sup>lt;sup>2</sup>https://play.google.com/store

<sup>&</sup>lt;sup>3</sup>https://winauth.github.io/winauth/download.html

<sup>&</sup>lt;sup>4</sup>https://authy.com/download/

<sup>&</sup>lt;sup>5</sup>https://www.apple.com/app-store/

Tabela 1: Preiskane aplikacije in operacijski sistemi, ki jih podpirajo.

Aplikacija	Verzija	Windows	Android	iOS
WinAuth	3.5.1	√		
2 Factor Auth	2.5.1804	$\checkmark$		
Twilio Authy	4.8.8	✓	<ul> <li>✓</li> </ul>	√ \
Aegis	2.0.3		$\checkmark$	
FreeOTP	1.5		<ul> <li>✓</li> </ul>	
TOTP	1.89		$\checkmark$	
Google	5.20 R4		$\checkmark$	<ul> <li>✓</li> </ul>
Microsoft	6.2207.4624		$\checkmark$	√ \
2FAS	3.17.0		$\checkmark$	<ul> <li>✓</li> </ul>
Okta Verify	7.9.0			<ul> <li>✓</li> </ul>
Two-Factor	1.5			<ul> <li>✓</li> </ul>
FIS Authenticator	4.4.6			√ \
Epic authenticator	1.0			<ul> <li>✓</li> </ul>
IBM Verify	2.5.2			<ul> <li>✓</li> </ul>
Authenticator+	2.0.4			<ul> <li>✓</li> </ul>

Za pridobivanje podatkov z naprave z operacijskim sistemom Android sta bili uporabljeni orodji Magnet Acquire<sup>6</sup> in Android Debug Bridge (ADB)<sup>7</sup>, z naprave z operacijskim sistemom iOS pa orodje ArtEx<sup>8</sup>. Naprave so bile na internetno omrežje povezane preko WiFi vroče točke, ki je izolirala promet, ki so ga ustvarile naprave tekom testiranja. Promet mobilnih naprav, pregledan z orodjem Wireshark<sup>9</sup>, je bil šifriran.

Na virtualni Windows napravi je bilo uporabljeno orodje Fiddler<sup>10</sup> za zajem omrežnega prometa, orodje iDumpIt<sup>11</sup> pa za zajem izpisa pomnilnika. Datotečni sistemi oz. podatki so bili pregledani s programom Autopsy<sup>12</sup> oz. FTK Imager<sup>13</sup>.

#### 4. NAJDENI PODATKI

Odkritih je bilo 25 artefaktov, ki so nosili najpomembnejše podatke. Od tega jih je bilo 13 izluščenih iz naprave z operacijskim sistemom Android, preostali pa so bili pridobljeni z naprav z operacijskim sistemom iOS in Windows.

Od opazovanih 15 aplikacij jih je 12 shranjevalo uporabniška imena v nekodirani obliki. 14 aplikacij je vrnilo ime svojega izdajatelja, npr. Facebook ali Instagram. Unikatna univerzalna identifikacija (UUID) je bila spravljena v 5 aplikacijah. UUID identificira vsako aplikacijo, ki uporablja 2FA. Na 8 aplikacijah se je shranil časovni žig, kot bi se ustvaril npr. pri stvarjenju Instagram računa. Aplikacija Twilio Authy specifično je hranila ogromno informacij o identiteti naprave in njenega uporabnika. Med drugim je bilo mogoče najti telefonsko številko (edina od 15 aplikacij, ki jo je zahtevala), IP naslove in lokacijo naprave ter podatke o drugih napravah, vezanih na podano telefonsko številko. Tajni ključi so se nahajali v 14 od 15 testiranih aplikacij. Nekateri med njimi so bili shranjeni v nekodirani obliki, preostale ključe pa se je dekodiralo s pomočjo PIN-ov, kjer jih je bilo mogoče najti. Pomembna opomba je še, da kodiranje ključev ni bilo nujno odvisno od obstoja PIN-a, nekatere aplikacije so ključe kodirale, ne glede na to ali je uporabnik napravo zaklenil ali ne.

Na virtualni Windows napravi je bil dodatno izveden še zajem prometa na omrežju. Pomembnejši podatki so bili zajeti, ko je bila aplikacija v uporabi. V pomnilniku namreč ni ostalo veliko sledi po tem, ko je bila aplikacija povsem zaprta. Orodje Strings omogoča iskanje nizov v izpisu pomnilnika, kar je olajšalo iskanje tajnih ključev in drugih podatkov. Podobno iskanje je bilo izvedeno s orodjem Fiddler na omrežnem prometu.

Vse aplikacije, testirane na virtualni Windows napravi, so vrnile telefonske številke in e-poštne naslove. Kot omenjeno že prej, je vrnila aplikacija Twilio Authy še nekoliko več osebnih podatkov.

#### 5. OBHOD 2FA S TOTP

TOTP pri izračunu enkratnega gesla uporabi trenutni čas in tajni ključ. V večini implementacij gre za kodo iz šestih števk, ki običajno velja okoli 30 s, kot je bilo že omenjeno v poglavju 1. Tajni ključ se pridobi ob povezovanju storitve z 2FA aplikacijo in se shrani na napravo.

TOTP je različica HOTP (HMAC one time passwords), ki temelji na HMAC (angl. hash-based message authentication code) algoritmu. HMAC uporablja kriptografsko zgoščevalno funkcijo, ki izračuna t. i. hash fiksne dolžine. Ker zgoščevalna funkcija uporablja tudi tajni ključ, gre za kriptografsko zgoščevalno funkcijo. Tega skupaj z vhodnim besedilom preslika v hash. Pri TOTP vhodno besedilo predstavlja trenutni čas.

Ideja, ki so jo uporabili v članku [1], je temeljila na tem, da vse TOTP aplikacije uporabljajo enak HMAC algoritem. Če nam uspe pridobiti tajni ključ za HMAC izbrane TOTP aplikacije v nešifrirani, tekstovni obliki (angl. plain-text), bi ga lahko uporabili za katerokoli drugo TOTP aplikacijo. V članku [12] so tovrstne ključe pridobili na napravah z operacijskim sistemom Android in poročali, kje v pomnilniku se nahajajo. Pri tem so uporabili pristop inverznega inženiringa in razne forenzične metode za analizo spomina.

V članku [1] so iskanje tajnih ključev še dodatno razširili in testirali 2FA aplikacije zapisane v Tabeli 1. Teste so izvedli nad storitvenima aplikacijama Facebook in Twitter na operacijskih sistemih Android, iOS in Windows 10. Postopek, ki so ga predlagali, se začne s pridobitvijo nešifriranega tajnega ključa iz pomnilnika ali prometa na omrežju z uporabo ustreznega forenzičnega orodja. Vsak pridobljeni tajni ključ se nato vnese v druge 2FA aplikacije, torej ne v tisto, ki ga je ustvarila. Nova 2FA aplikacija sedaj lahko generira veljavno enkratno geslo na podlagi vnešenega tajnega ključa in trenutnega časa. Generirana koda (angl. one-time password, kratica OTP) se lahko nato vnese v storitveno aplikacijo, zaščiteno z 2FA. Opisani postopek je prikazan na Sliki 1.

Izziv predstavljajo tiste 2FA aplikacije, ki tajne ključe šifri-

 $<sup>^{6} \</sup>rm https://www.magnetforensics.com/resources/magnet-acquire/$ 

<sup>&</sup>lt;sup>7</sup>https://developer.android.com/tools/adb

<sup>&</sup>lt;sup>8</sup>https://www.doubleblak.com/ArtEx/?key=ArtEx

<sup>&</sup>lt;sup>9</sup>https://www.wireshark.org/

<sup>&</sup>lt;sup>10</sup>https://www.telerik.com/fiddler

<sup>&</sup>lt;sup>11</sup>https://www.magnetforensics.com/resources/magnet-

dumpit-for-windows/#Our%20Blog

<sup>&</sup>lt;sup>12</sup>https://www.autopsy.com/

 $<sup>^{13} \</sup>rm https://www.exterro.com/digital-forensics-software/ftk-imager$ 



Slika 1: Koraki obhoda 2FA s TOTP.

rajo. Za nekatere je to privzeta nastavitev, za druge pa opcijska. Šifrirane ključe je treba posledično najprej dešifrirati, kar je pogosto precej težko. Težava nastane tudi tedaj, ko tajni ključ ni več veljaven. To se zgodi npr., ko 2FA začasno onemogočimo, ob ponovni vzpostavitvi pa nam je dodeljen nov, drugačen tajni ključ. Iz vidika varnosti uporabnika je seveda zaželjeno, da aplikacija za TOTP šifrira tajne ključe.

Cilj eksperimenta obhoda TOTP v [1] je bil predvsem pokazati, da bodo ob vpisu enakega (veljavnega) tajnega ključa vse različne 2FA aplikacije generirale enak OTP. Tajna ključa, ki sta bila ustvarjena tekom vzpostavljanja 2FA na Facebooku in Twitterju, sta bila vnešena v ostale aplikacije za dvofaktorsko avtentikacijo. Aplikacije so generirale OTP-je, katere so med seboj primerjali in potrdili, da so povsem enaki. V vsakega izmed računov storitvenih aplikacij so se nato prijavili z generiranim enkratnim geslom in tako potrdili, da je veljaven.

Pri opisanem postopku velja predpostavka, da že poznamo uporabniško ime in geslo računa, v katerega se prijavljamo. Tovrstne informacije digitalni forenzik običajno pridobi že vnaprej. Uspešna pridobitev tajnega ključa za TOTP mu olajša preiskovalno delo, saj mu posledično ni treba uporabljati originalne naprave. Ta lahko ostane na varnem mestu in mu z njo ni treba rokovati. S tem se izogne tudi morebitni spremembi podatkov na originalni napravi. Tajni ključ lahko vpiše v drugo TOTP aplikacijo in dostopa do storitve, ki je bila zaščitena z dvofaktorsko avtentikacijo.

#### 6. TEST PREDLAGANEGA OBHODA TOTP

Predlagani obhod TOTP, opisan v poglavju 5, smo testirali nad storitveno aplikacijo Facebook in aplikacijami za generiranje OTP, ki so zapisane v Tabeli 2. Vse aplikacije so bile naložene na operacijski sistem Android.

Tabela 2: Testirane aplikacije.

Aplikacija	Razvijalec
Aegis Authenticator	Beem Development
2FA Authenticator (2FAS)	2FAS
Microsoft Authenticator	Microsoft Corporation
FreeOTP Authenticator	Red Hat
Google Authenticator	Google LLC

#### Koraki samostojnega testiranja

- 1. Najprej smo vklopili *dvojno preverjanje pristnosti* v aplikaciji Facebook. Ta je generirala tajni ključ, ki ga TOTP aplikacije potrebujejo za generiranje 6-mestne kode za vstop v aplikacijo Facebook.
- Nato smo na napravo z operacijskim sistemom Android naložili 5 različnih aplikacij za generiranje OTP-jev.
- V vsaki aplikaciji smo dodali Facebookov tajni ključ, kar je sprožilo periodično generiranje 6-mestnih kod.
- Primerjali smo generiranje 6-mestnih kod v vseh petih aplikacijah in potrdili, da so kode povsem enake, torej uporablja vseh 5 aplikacij enak algoritem za generacijo.
- 5. Preverili smo, da nam generirane kode res dovoljujejo vstop v aplikacijo Facebook.

#### 6.1 Težave pri zajemanju

Ker je aplikacija Twilio Authy zahtevala več osebnih podatkov kot druge aplikacije in v primerjavi z ostalimi bolj slabo ščiti le-te [1], smo se odločili proti njenemu testiranju. Ob uporabi aplikacij s TOTP smo želeli prikazati postopek vnašanja tajnega ključa, a je imelo vseh 5 aplikacij onemogočeno shranjevanje posnetkov zaslona med uporabo. Android napravo smo zato povezali na računalnik, pri čemer smo uporabili orodja  $SDK^{14}$  in  $scrcpy^{15}$ . S tem smo skušali zaobiti omejitve aplikacij, vendar je bil zaslon prek prenosa na računalniku zatemnjen in zato nismo mogli narediti lepih posnetkov. Edina izmed aplikacij, ki je v korakih nastavitve dovoljevala zajem zaslona, je bila FreeOTP Authenticator. Ta je omogočala zajem vse do glavnega menija aplikacije, torej še med vnosom varnostnega gesla za samo aplikacijo.

## 6.2 Opažanja med uporabo različnih 2FA aplikacij

Zaklepanje aplikacije sta avtomatsko ponudili Aegis Authenticator in FreeOTP Authenticator. Po prvotni nastavitvi gesla, FreeOTP Authenticator ni več zahteval vnašanja gesla, dokler je bila aplikacija odprta v ozadju. Aplikacija Microsoft Authenticator je zahtevala zaklep aplikacije takoj po vnosu prvega tajnega ključa. Poleg tega je zahtevala odklepanje vsakič, ko je bila odprta na zaslonu, tudi če je bila pred tem že odprta v ozadju naprave. Aegis Authenticator in 2FA Authenticator (2FAS) avtomatsko ponujata možnost ustvarjanja varnostne kopije. 2FA Authenticator (2FAS) specifično ponuja povezavo z Google računom in hrambo kopije na Google Drivu.

 $<sup>^{14} \</sup>rm https://developer.android.com/tools/releases/platform-tools$ 

 $<sup>^{15} \</sup>rm https://github.com/Genymobile/scrcpy/releases$ 

Vse aplikacije omogočajo možnost uvažanja podatkov iz drugih TOTP aplikacij in skeniranje QR kode namesto ročnega vnašanja tajnega ključa. Google Authenticator ob ročnem vnosu za razliko od ostalih TOTP aplikacij nujno zahteva ime storitve (*Facebook* v našem primeru). Poleg tega je aplikacija FreeOTP Authenticator omogočala tudi ročno nastavljanje lastnosti generiranja varnostnih kod. Med drugim je omogočala izbiro med TOTP ali HOTP, izbiro med 6- ali 8-mestno kodo, nastavitev dolžine intervala in izbiro algoritma. S primerjanjem različnih algoritmov in ujemanjem aplikacij smo ugotovili, da se v algoritmu za dvofaktorsko avtentikacijo aplikacije Facebook uporablja SHA-1.

#### 6.3 Kratek pregled izračuna TOTP

SHA-1 (angl. Secure Hash Algorithm 1) je specifičen tip varne zgoščevalne funkcije, ki se med drugim uporablja tudi pri izračunu TOTP. Kot je bilo že omenjeno, potrebujemo za izračun TOTP tajni ključ in trenutni čas v Unix Epoch času.

Najprej izračunamo število časovnih korakov od začetka Unix Epoch časa  $t_u$ .  $t_s$  v formuli predstavlja čas enega koraka, torej kako dolgo velja ena 6-mestna koda. V našem primeru velja  $t_s = 30s$ . Z  $N_{HEX}$  (enačba 1) označimo število časovnih korakov v šestnjastiškem zapisu s 16 števkami. Zapisu po potrebi spredaj dodamo ničle.

$$N_{HEX} = \left\lfloor \frac{t_u}{t_s} \right\rfloor \tag{1}$$

 ${\cal N}_{HEX}$ zapišemo v obliki 64 bitov.

Tajni ključ je po generaciji v aplikacijah, ki omogočajo dvofaktorsko avtentikacijo, v šestnajstiškem zapisu. Zapišemo ga v obliki 160 bitov. Nato s tajnim ključem in  $N_{HEX}$  v primerni obliki izračunamo HMAC hash s pomočjo SHA-1. S tem pridobimo zapis v 160 bitih.

Zadnje 4 bite izračunanega zapisa pretvorimo v desetiški zapis. Ta številka označuje odmik (angl. *offset*). Odmik nam pove, kako dobimo 32-bitni izsek HMAC hasha za naslednji korak. Nad tem izsekom in številom 0x7FFFFFF uporabimo operacijo IN. Pridobljeni rezultat označimo s k. Izračunamo žeton m kot

$$m = k \mod 10^n,\tag{2}$$

kjer je ndolžina žetona. V našem primeru gre za 6-mestno kodo, zato veljan=6.

Na takšen način se izračuna enkratno geslo oziroma 6-mestna koda na obeh straneh - v aplikaciji, ki zahteva geslo, in naši izbrani 2FA aplikaciji [12]. Če se obe gesli ujemata, to potrdi, da imamo pravi tajni ključ za račun, v katerega se skušamo vpisati.



Slika 2: HMAC izračun zavoljo TOTP.

# 7. ZAKLJUČEK

V okviru seminarske naloge smo izluščili glavni del izbranega članka [1] in na njegovi osnovi še naprej raziskali področje aplikacij za dvofaktorsko avtentikacijo. Našli smo druge članke, ki ocenjujejo uporabnost tovrstnih aplikacij, in potrdili sklepe glavnega članka s samostojnim delom. Ugotovili smo, da je mogoče enak tajni ključ uporabiti v različnih 2FA aplikacijah, saj uporabljajo enak algoritem za izračun enkratnega gesla. Potrudili smo se še sami raziskati delovanje 2FA aplikacij na Android napravi, kar je med drugim zahtevalo uporabo novega programja.

Na osnovi predhodnih člankov in lastnega dela smo prišli do sklepa, da je v sodobnem času poudarek na varnosti predvsem v digitalnem svetu - izredno pomemben. Nekatere aplikacije, ki zahtevajo osebne podatke uporabnika, svojih datotek (in s tem osebnih podatkov) niti ne šifrirajo. To omogoča vsakemu tehnično usposobljenemu nepridipravu, da se dokoplje do podatkov, ki mu omogočajo dostop tudi do drugih aplikacij, ki jih žrtev uporablja. Čeprav so takšne pomanjkljivosti koristne pri delu digitalnih forenzikov, bi bilo njihovo delo v splošnem zmanjšano, če bi imeli nepridipravi do njih manj dostopa.

#### 8. LITERATURA

- J. Berrios, E. Mosher, S. Benzo, C. Grajeda, and I. Baggili. Factorizing 2fa: Forensic analysis of two-factor authentication applications. *Forensic Science International: Digital Investigation*, 45:301569, 2023.
- [2] J. W. Byun et al. A generic multifactor authenticated key exchange with physical unclonable function. Security and Communication Networks, 2019, 2019.
- [3] D. Childers. State of the auth 2021. URL: https://duo.com/assets/ebooks/state-of-the-auth-2021.pdf (pridobljeno 18. 3. 2024).
- [4] E. Erdem and M. T. Sandıkkaya. Otpaas—one time password as a service. *IEEE Transactions on Information Forensics and Security*, 14(3):743–756, 2018.
- [5] Internet crime report 2022. URL: https://s3.documentcloud.org/documents/23707016/ 2022\_ic3report.pdf (pridobljeno 25. 3. 2024).
- [6] Internet crime report 2023. URL: https://www.ic3.gov/Media/PDF/AnnualReport/ 2023\_IC3Report.pdf (pridobljeno 25. 3. 2024).
- J. Fruhlinger. 2fa explained: How to enable it and how it works. URL: https://www.csoonline.com/article/563753/2faexplained-how-to-enable-it-and-how-it-works.html (pridobljeno 25. 3. 2024).
- [8] P. Gope, A. K. Das, N. Kumar, and Y. Cheng. Lightweight and physically secure anonymous mutual authentication protocol for real-time data access in industrial wireless sensor networks. *IEEE transactions* on industrial informatics, 15(9):4957–4968, 2019.
- R. Grimes. 12+ ways to hack multi-factor authentication. URL: https://www.knowbe4.com/hubfs/12+\_Ways\_to\_Hack\_ Two-Factor\_Authentication-1.pdf (pridobljeno 7. 4. 2024).
- [10] L. W. Kevin Bankston, Ross Schulman. What is

two-factor authentication (2fa)? URL: https://www.newamerica.org/in-depth/gettinginternet-companies-do-right-thing/case-study-2offering-two-factor-authentication (pridobljeno 18. 3. 2024).

- [11] What is two-factor authentication? URL: https://www.microsoft.com/en-us/security/business/ security-101/what-is-two-factor-authentication-2fa (pridobljeno 18. 3. 2024).
- [12] C. Ozkan and K. Bicakci. Security analysis of mobile authenticator applications. In 2020 International Conference on Information Security and Cryptology (ISCTURKEY), pages 18–30, 2020.
- [13] A. Petrosyan. Number of internet and social media users worldwide as of january 2024. URL: https://www.statista.com/statistics/617136/digitalpopulation-worldwide (pridobljeno 18. 3. 2024).
- [14] K. Reese, T. Smith, J. Dutson, J. Armknecht, J. Cameron, and K. Seamons. A usability study of five Two-Factor authentication methods. In *Fifteenth Symposium on Usable Privacy and Security (SOUPS* 2019), pages 357–370, 2019.
- [15] Account hijacking: Understanding the threat and how to protect yourself. URL: https://www.linkedin.com/pulse/account-hijackingunderstanding-threat-how-protect-yourself-(pridobljeno 18. 3. 2024).
- [16] D. Wang and P. Wang. Two birds with one stone: Two-factor authentication with security beyond conventional bound. *IEEE transactions on dependable* and secure computing, 15(4):708–722, 2016.
- [17] Q. Wang, D. Wang, C. Cheng, and D. He. Quantum2fa: efficient quantum-resistant two-factor authentication scheme for mobile devices. *IEEE Transactions on Dependable and Secure Computing*, 20(1):193–208, 2021.

# **Decrypting TPM-Protected BitLocker Volumes**

Jernej Jezeršek Faculty of Mathematics and Physics Jadranska ulica 19 Ljubljana, Slovenia jj13824@student.uni-lj.si Erazem Kokot Faculty of Computer and Information Science Večna pot 113 Ljubljana, Slovenia ek0273@student.uni-lj.si Luka Opravš Faculty of Mathematics and Physics Jadranska ulica 19 Ljubljana, Slovenia Io14200@student.uni-Ij.si

# ABSTRACT

TPM-protected BitLocker volumes can pose significant challenges in digital forensic investigations. This seminal work researches different methods for encryption key recovery including Cold Boot attacks, TPM sniffing, DMA bug exploitation, and processor debugging. Our focus is primarily on the replication and evaluation of the TPM sniffing method.

### **Keywords**

Trusted Platform Module, TPM, Cold boot attack, TPM sniffing, Encryption key recovery, BitLocker

# 1. INTRODUCTION

In the realm of digital forensics, encrypted storage drives pose one of the most daunting challenges to investigative efforts, dramatically complicating the retrieval of crucial evidence compared to traditional methods of data hiding. Encryption technologies such as Microsoft's BitLocker have become commonplace, leveraging robust encryption algorithms to secure data. Particularly, BitLocker volumes protected by the Trusted Platform Module (TPM) present a formidable barrier due to the TPM's design to safeguard cryptographic keys, making unauthorized access significantly more challenging. The evolution of encryption has thus necessitated equally advanced forensic methodologies to decrypt such volumes without the original credentials.

Our research takes its foundational basis from the work titled "Forensic method for decrypting TPM-protected Bit-Locker volumes using Intel Direct Connect Interface (DCI)"[3]. We explored a range of forensic methods that can potentially uncover the encryption keys essential for accessing secured data. These methods include the Cold Boot attack, TPM sniffing, exploiting Direct Memory Access (DMA) bugs, and processor debugging techniques.

Among the various methods evaluated, we determined that

the TPM sniffing approach held the most promise for yielding successful decryption results. Consequently, we elected to focus our experimental efforts on recreating this specific technique. In this paper, we present a detailed report of our findings and the methodology employed in the implementation of the TPM sniffing method.

# 2. BITLOCKER

BitLocker is a Microsoft security feature that provides fullvolume encryption and prevents data theft from stolen, lost, or inappropriately decommissioned devices. It is included by default in most modern Windows OS [12].

It encrypts each sector of a memory device individually with an encryption key derived from the Full Volume Encryption Key (FVEK) and a sector number. Using the sector number ensures that parts of the disk containing identical data will have different binary representations when encrypted. This makes attempts to discover keys by analyzing the encrypted data much more difficult. The algorithm used for encryption is AES [3].

The FVEK is stored encrypted on the disk as part of the volume metadata. It is encrypted with a Volume Master Key (VMK), which is also stored in volume metadata and is encrypted with one or a combination of more protectors. When BitLocker is first enabled, it also generates a recovery key. It can be used as a backup key if the disk cannot be decrypted with the use of protectors. For this reason, another copy of VMK encrypted with the recovery key is also stored in volume metadata. The recovery key can be stored in the user's Microsoft account. Both the VMK and FVEK are 256-bit keys. The FVEK is encrypted using the AES algorithm, while the VMK can be encrypted either with AES or RSA, depending on the used protectors.

Some commonly used protectors are Trusted Platform Module (TPM), PIN, and startup key. Combinations of multiple protectors can also be used. Other protectors exist, but they fall outside the scope of this paper [13].

The TPM is one of the most commonly used protectors. It is a low-cost crypto-processor primarily utilized for measuring the integrity of the boot process, providing OS authentication methods, and safeguarding the keys required by BitLocker to decrypt the VMK. The TPM protects the Storage Root Key (SRK), an RSA public/private key pair used for VMK decryption. The SRK is stored in the TPM's non-volatile protected memory, which is accessible only to a few secure operations. For instance, encryption of a message with the key is permitted, whereas copying the key is restricted. This prevents external programs from accessing the keys, ensuring that even with unrestricted access to our entire computer, sensitive information within the TPM remains inaccessible. When the OS wants to encrypt/decrypt the VMK, it sends it to the TPM, which encrypts/decrypts it with the SRK and returns the result [3]. Starting with Windows 11, TPM 2.0 has become a hardware requirement, establishing BitLocker encryption via the TPM as a standard practice.

A user-provided PIN code can also be used as a protector. In this case, it must be entered by the user whenever decrypting the device. Because BitLocker allows unlimited unlocking attempts, short PINs can be brute-forced.

Another example of a protector is the startup key. It is an encryption key stored on another memory device (for example a USB stick) that must be provided for decryption of VMK.

When the OS wants to access the protected storage device, it must first decrypt the VMK with its protectors or recovery key. Once the VMK is decrypted, it is in turn used to decrypt the FVEK, which is finally used to decrypt individual sectors on the disk. Therefore to decrypt the disk, one of the following is required:

- A FVEK
- A VMK
- A recovery key

The reason for using the VMK is that it can allow us to safely change the compromised protectors without the need to reencrypt the entire disk. If a protector is compromised, the encrypted VMK might also be compromised, as it is often passed to protectors during decryption (for example, it is passed to TPM). In such cases, security can be restored by decrypting the FVEK, changing the compromised protector and VMK, and then re-encrypting the FVEK with the new VMK. Since FVEK is still the same, we do not need to reencrypt the data on the disk. Even if an attacker manages to decrypt the old VMK, they will still be unable to decrypt the new FVEK and access the data. Of course, this relies on the assumption that the attacker cannot acquire the old encrypted FVEK. If there is suspicion that this may have occurred, disk re-encryption becomes necessary [3].

It is worth noting that we could not find any non-anecdotal evidence of how BitLocker impacts the performance of a computer, but according to a Microsoft article Windows 10 and Windows 11 don't aggressively request resources while encrypting, so the performance impact should be minimal [14].

# 3. KEY RECOVERY METHODS

Although BitLocker is presented by Microsoft as an impervious security system, which "helps mitigate unauthorized data access by enhancing file and system protections, rendering data inaccessible when BitLocker-protected devices are decommissioned or recycled" ([12]), there have long been many well-known ways of accessing BitLocker-encrypted volumes, without having access to the encryption keys. This chapter describes a few of them in detail.

## **3.1** Cold boot attack

When a computer is turned off, its DRAM retains its contents for a short time. The duration for data to be erased depends on the temperature, but even at room temperature, data usually persists for several seconds, even if the DRAM is removed from the computer. This enables attackers with physical access to a working computer to power it off and quickly copy the contents of the DRAM.

To prevent data decay, one could simply plug the DRAM into another computer and turn it on. However, booting the computer will overwrite some parts of the data in DRAM. It is better to use tiny special-purpose programs that use trivial amounts of RAM and copy the rest to an external medium. With a bootable USB device, such a program can even be executed directly on the target computer. An attacker could turn the computer off, insert the bootable USB, turn the computer on, and boot it with the USB. The code on the USB would then copy the contents of DRAM to the USB device.

Once the data is securely copied, it can be analyzed. Often it is possible to recover cryptographic keys used for encrypting the computer disk and use them to decrypt the disk. Other sensitive information such as user passwords might also be recovered.

This method can be used to bypass BitLocker's protection. After powering off the computer, the FVEK will persist in DRAM. Contents of DRAM can be dumped onto an external disk and analyzed. Potential AES keys can then be identified and extracted from the memory dump. These extracted keys can then be used to attempt to decrypt the disk. Details about this attack can be found in the paper "Lest We Remember: Cold Boot Attacks on Encryption Keys" [10].

If BitLocker uses only TPM, we can bypass protection even if the computer has been turned off for a long time. When the computer is powered on, the VMK key is automatically decrypted with TPM and stored in DRAM. This attack can then be used to dump the contents of DRAM to an external drive and keys can then be extracted.

While this method does require physical access to the target computer, it offers numerous advantages. It can be utilized on any running computer and requires only a simple USB key. Furthermore, it works on most encryption systems, not just BitLocker (for example, FileVault and TrueCrypt). Additionally, if BitLocker uses only TPM as its protector, then the disk can be decrypted with this method even if the computer is initially turned off.

# 3.2 TPM sniffing

Since TPMs are ubiquitous and used by default on Windowsbased machines, they are a very powerful attack vector. A TPM's communication with the rest of the hardware can happen over an LPC, SPI, or I2C bus, depending on the specific chip [5]. TPM sniffing relies on connecting to the data lines of these buses, listening to signals with a logic analyzer, oscilloscope, or a similar device, and decoding them,

NC	1 ()	28	LPCPD
NC	2	27	SERIRQ
NC	3	26	LAD0
GND	4	25	NC
NC	5	24	VPS
NC	6	23	LAD1
PP	7 TSSOF	<b>28</b> 22	LFRAME
VNC	8	21	LCLK
VNC	9	20	LAD2
VPS	10	19	NC
GND	11	18	GND
NC	12	17	LAD3
NC	13	16	LRESET
NC	14	15	NC

Figure 1: ST33TPM12LPC pinout [16]

as is shown in Section 4.

This approach also requires physical access to the device, as well as disassembling the chassis and connecting probing wires to the TPM pins. Although not all TPMs are the same, Figure 1 shows a pinout of a TPM communicating over the LPC bus.

To retrieve the communication data, the probes must be attached to signals LAD1-3, LCLK, LFRAME, and LRESET [2] (explained further in Section 4.3).

There have been multiple proof-of-concept attacks on TPMs, such as those in [2] and [5], and there are also a few opensource projects that can be used in combination with a microcontroller or another portable computer, such as the Raspberry Pi Pico, which allow TPM sniffing on the go ([19], [20]).

Due to this style of attack, using just a TPM alone for storing the encryption keys is not enough, and the use of a 2nd-factor protector, such as a PIN, can easily mitigate the attack vector. The downside of this is that the user must now enter two passwords before they can log into the system, one to decrypt the volume, and another to log in to their account. Alternatively, a Smart card or USB key can be used to provide the necessary additional encryption material.

#### 3.3 Direct Memory Access bugs

Direct Memory Access (DMA) is a commonly used feature of modern computer systems that allows hardware components to access system memory without the need for the Central Processing Unit (CPU) to be involved during the transfer. In systems without DMA, the CPU is fully occupied for the duration of the reading/writing, preventing other work, while with DMA, the CPU only initiates the transfer and the hardware component can then interact with the system memory without its involvement. The CPU is notified of the end of the transfer through the use of an interrupt, issued by the DMA controller (DMAC). Attacking the DMA means exploiting the ports that are used for DMA to intercept communication and then abusing the direct access to system memory to bypass OS security features [17]. Because DMA allows access to any available memory address, this is an extremely powerful attack vector, but luckily it requires physical access to the device and in some cases, it can be disabled in BIOS/UEFI if DMA is not needed.

DMA attacks can be performed similarly to TPM attacks, e.g. by using an FPGA, connected to the Thunderbolt, ExpressCard, or PCIe hardware interfaces, and using it to intercept and modify the sent data. Examples of DMA attacks have been shown by Microsoft Mechanics, as well as security researchers from Ledger at Black Hat USA in 2022 [17]. There also exist open source implementations of the attack (either in combination with external hardware or just in software), such as [8].

#### **3.4 Processor debugging with Intel DCI**

Intel Direct Connect Interface (DCI) enables JTAG (Joint Test Action Group) debugging on computers with newer Intel CPUs (Skylake 6th gen and newer), by connecting two computers using USB debug cables or specialized Intel Silicon View Technology (SVT) adapters. DCI enables full control of the CPU, e.g. stopping the execution and running instructions step by step, adding breakpoints, or modifying register values [3]. All of this can be done using official Intel SDKs and debugging code can even be written using Python, if used in combination with Intel's OpenIPC.

Intel DCI can be enabled or disabled in BIOS/UEFI, so besides needing physical access, the option also has to be enabled for the attack to work.

With a little reverse engineering of the Windows Boot Manager to obtain the memory location of the return address of the TPM's unsealing function, the Intel DCI functionality can be exploited to read out the full VMK of the BitLockerencrypted volume, as is shown in [3].

## 4. RECREATING BITLOCKER KEY RECOV-ERY

In our research, we endeavored to replicate the TPM sniffing technique for extracting BitLocker encryption keys as delineated in the works by [5] and [2]. Despite our efforts, we encountered constraints in terms of time and hardware resources, which impeded our ability to successfully recover the encryption key. Nonetheless, significant progress was made in other technical aspects; we successfully identified the location of the TPM on the motherboard, engaged in probing the Low Pin Count (LPC) bus, and conducted a thorough inspection of its signals. These steps are crucial for understanding the communication dynamics and operational mechanisms within the system, offering foundational insights for future attempts and refinements in the TPM sniffing process.

#### 4.1 **Preparing the test device**

For our experimental setup, we selected a Lenovo ThinkPad T420, a model from 2011, due to its low cost and it is the



Figure 2: Enabled BitLocker encryption of the system volume.

same make as the laptop used in the study by Thomas Dewaele, Julien Oberson [5].

To prepare the device for the experiment, we activated the Trusted Platform Module (TPM) in the BIOS settings and proceeded to install a fresh version of Windows 10 Pro. To mirror a realistic operational environment typically encountered in contemporary digital systems, we enabled BitLocker encryption on the C: system volume, as depicted in figure 2. The encryption key was securely stored on the Trusted Platform Module (TPM) of the device.

#### 4.2 Sniffing the TPM signals

The Lenovo ThinkPad T420 utilized in our experiments is equipped with a built-in TPM 1.2 module. By accessing schematic and board view files that were freely available on an online forum [1], we were able to figure out how the TPM is connected on our specific laptop model. Our investigation revealed that the TPM was interfaced with the system motherboard via the Low Pin Count (LPC) bus, which is commonly used for that purpose. Furthermore, we identified unused pads close to the TPM that could be used to sniff the LPC bus (Figure 4).

Upon identifying the relevant TPM connection points, we located both the TPM module and the corresponding unpopulated pads on the motherboard (Figure 3). We soldered thin probe wires to these pads, establishing the necessary physical connections for signal capture and analysis (Figure 5). This setup enabled us to directly monitor and record the communication signals transmitted across the LPC bus.

# 4.3 Analyzing the LPC bus

The LPC protocol requires 7 digital active-low signals to enable bidirectional data transfer [21].

- LCLK: 33.3 MHz clock
- LRESET: Bus reset signal
- LFRAME: A signal that indicates the beginning of a frame
- LADO, ..., LAD3: 4 signals that can carry multiplexed address, data, and other information



Figure 3: TPM chip (U39) and unpopulated pads (J5) on the motherboard of the laptop.



Figure 4: Motherboard in OpenBoardView showing the connection between the unpopulated pads and the TPM



Figure 5: Soldered probes on the LPC bus. White-green LAD0, green LAD1, white-blue LAD2, blue LAD3, white-brown LFRAME, brown GND



Figure 6: Laptop with attached oscilloscope probes on LFRAME, LDA0, LDA1, LDA2 lines. The oscilloscope shows a burst of activity on the LPC bus during the Windows boot process.

To confirm the successful probing of the LPC bus on our Lenovo ThinkPad T420, we connected the probe wires to an oscilloscope for real-time signal inspection during the system boot process, as illustrated in Figure 6. Due to the limitations of our equipment, which included access to only a 4-channel oscilloscope, we could monitor just the LADO, LAD1, LAD2, and LFRAME signals.

Consistent with typical LPC bus behavior, these signals are designed to be active-low. This configuration means that under idle conditions (no data transmission), the signals maintain a high state (3.3V), and they dip to a low state (0V) during data transmission activities. We configured the oscilloscope to trigger on the LFRAME signal to analyze a single LPC frame.

Upon triggering, we captured a specific frame, depicted in Figure 7. Despite some degree of signal noise, we were able to distinctly observe the LFRAME signal dropping to the low state momentarily at the commencement of the data frame. Following this, the data signals LADO, LAD1, and LAD2 demonstrated a sequential pattern indicative of byte transmission. This pattern confirms the active communication over the LPC bus and validates our probing setup.

#### 4.4 Retriving the BitLocker key

To further our investigation into TPM signal interception and decoding, we selected the Raspberry Pi Pico microcontroller as our hardware platform due to its efficient handling of high-speed digital signals via its Programmable Input/Output (PIO) capabilities and very low cost. We utilized the "Pico TPM Sniffer," a software developed by Thomas Roth, known within the community as Stacksmashing, which is designed to exploit the Pico's PIO features to decode the LPC bus signals and extract the BitLocker Master Volume Key (MVK) [19].

We cloned the software repository from GitHub, compiled the source code into a .uf2 file - a format compatible with



Figure 7: Analyzing one frame of data on the LPC bus. The green line shows the LFRAME signal indicating the beginning of the frame. Yellow, magenta and blue lines show the LDA0, LDA1, and LDA2 signals respectively.

the Raspberry Pi Pico - and subsequently uploaded it to the microcontroller.

In alignment with the requirements specified by the Pico TPM Sniffer software, we established connections between the LPC bus signals and the Raspberry Pi Pico GPIO pins as follows:

- LADO to GPIO1,
- LAD1 to GPIO2,
- LAD2 to GPIO3,
- LAD3 to GPIO4 and
- LFRAME to GPI05.

It is important to note that we did not connect the LCLK signal because the Pico TPM Sniffer software is designed to deduce the clock signal directly from the LFRAME signal transitions.

Figure 8 shows the complete setup with Raspberry Pi Pico connected to the laptop via soldered probe wires.

Upon completing the setup and initiating the Raspberry Pi Pico, we connected to it via a serial port, receiving a positive initial response, as shown in Figure 9. Despite these promising early signs, when we attempted to decode the MVK following the target PC's boot process, the extraction was unsuccessful.

To resolve these issues, we tested a patch proposed by another developer, b8vr [20], which purportedly extends support to different VMK headers within the Pico TPM Sniffer software. However, even with this modified version, the program failed to decode the MVK successfully.

This outcome suggests that further modifications and testing are required to enhance the software's compatibility and functionality with various TPM configurations and encryption headers. Given the complexity of TPM interactions and



Figure 8: Laptop with attached Raspberry Pi Pico sniffing the LPC bus during boot process.



Figure 9: Screenshot of the Raspberry Pi Pico running the TPM Sniffer program by Stacksmashing

the variety of TPM versions and configurations, the tool's current limitations may stem from its inability to correctly interpret or synchronize with the specific signaling and data formats used by different TPM models or firmware versions.

# 5. CONCLUSION

Throughout this study, we delved deeply into the operational mechanisms of BitLocker, gaining substantial insight into its internal workings. Our exploration covered a spectrum of forensic methods aimed at recovering BitLocker encryption keys, such as Cold Boot attacks, TPM sniffing, exploiting DMA bugs, and processor debugging. A focal point of our research was an attempt to replicate a TPM sniffing method to retrieve encryption keys. This endeavor, however, presented significant challenges, primarily due to the variability among TPM models, which complicated the process more than initially anticipated.

Despite these challenges, our findings have laid a robust foundation for future investigations. Going forward, we plan to enhance our approach by employing a logic analyzer to analyze the LPC signals in more detail. This will allow us to follow the communication protocol more closely with the goal of successfully extracting the Volume Master Key (VMK). We hope to refine these forensic techniques and expand the toolkit available to cybersecurity professionals dealing with encrypted devices, contributing to a broader understanding and capability in digital forensics. Through persistent effort and refined methodology, we hope to overcome the current limitations and achieve a more comprehensive understanding of TPM-related security vulnerabilities.

#### 6. **REFERENCES**

- Board view and schematic source. https://www.badcaps.net/forum/ troubleshooting-hardware-devices-and-electronics-theory/ troubleshooting-laptops-tablets-and-mobile-devices/ schematic-requests-only/ 86677-need-lenovo-t420-board-view, 2021. Accessed: 2024-05-11.
- [2] D. Andzakovic. Extracting bitlocker keys from a tpm. https://pulsesecurity.co.nz/articles/TPM-sniffing, 2019. Accessed: 2024-05-04.
- [3] M. B. de Assumpção, M. A. Reis, M. R. Marcondes, et al. Forensic method for decrypting tpm-protected bitlocker volumes using intel dci. *Forensic Science International: Digital Investigation*, 44:301514, 2023.
- J.-C. Delaunay. Practical dma attack on windows 10. https://www.synacktiv.com/en/publications/ practical-dma-attack-on-windows-10.html, 2018. Accessed: 2024-05-11.
- T. Dewaele and J. Oberson. Tpm sniffing. https://blog.scrt.ch/2021/11/15/tpm-sniffing/, 2020. Accessed: 2024-05-04.
- [6] M. B. Elena Agostini. Bitcracker: Bitlocker meets gpus. International Journal of Information Security, 21:1005–1018, 2022.
- [7] K. Ezirim, W. Khoo, G. Koumantaris, R. Law, and I. M. Perera. Trusted platform module – a survey. 2014.
- U. Frisk. Pcileech direct memory access (dma) attack software. https://github.com/ufrisk/pcileech, 2021. Accessed: 2024-05-11.
- [9] U. Frisk. Memprocfs the memory process file system. https://github.com/ufrisk/MemProcFS, 2024. Accessed: 2024-05-11.
- [10] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten. Lest we remember: cold-boot attacks on encryption keys. *Commun. ACM*, 52(5):91–98, may 2009.
- [11] N. Heninger and H. Shacham. Reconstructing rsa private keys from random key bits. pages 1–17, 08 2009.
- [12] Microsoft. Bitlocker overview. https://learn.microsoft.com/en-us/windows/security/ operating-system-security/data-protection/bitlocker/, 2023. Accessed: 2024-05-11.
- Microsoft. Bitlocker overview. https://learn.microsoft. com/en-us/windows/security/operating-system-security/ data-protection/bitlocker/countermeasures, 2023. Accessed: 2024-05-11.
- Microsoft. Bitlocker configuration: known issues. https://learn.microsoft.com/en-us/troubleshoot/ windows-client/windows-security/ bitlocker-configuration-known-issues, 2024. Accessed: 2024-06-10.
- [15] H. Nurmi. Sniff, there leaks my bitlocker key. https://labs.withsecure.com/publications/ sniff-there-leaks-my-bitlocker-key, 2020. Accessed:

2024-05-07.

- J. Oberson. St33tpm12lpc pinout. https://blog.scrt.ch/ wp-content/uploads/2020/07/TPM\_chip\_pinout.png, 2021. Accessed: 2024-05-11.
- [17] C. Parrot. What is a dma attack? understanding and mitigating the threat. https://www.kroll.com/en/insights/publications/cyber/ what-is-dma-attack-understanding-mitigating-threat, 2018. Accessed: 2024-05-11.
- [18] A. Pilkey. The chilling reality of cold boot attacks. https://blog.f-secure.com/cold-boot-attacks/, 2018. Accessed: 2024-05-3.
- T. Roth. Pico tpmsniffer. https://github.com/stacksmashing/pico-tpmsniffer, 2024. Accessed: 2024-04-13.
- [20] T. Roth and b8vr. Pico tpmsniffer patch. https://github.com/b8vr/pico-tpmsniffer, 2024. Accessed: 2024-04-13.
- [21] Wikipedia. Low pin count bus. https://en.wikipedia.org/wiki/Low\_Pin\_Count. Accessed: 2024-04-18.

# Enhanced Password Cracking: Optimizing Context-Based Dictionaries with NLP and Sociological Insights

Manca Strah ms50362@student.uni-lj.si Faculty of Mathematics and Physics Faculty of Computer and Information Science University of Ljubljana Slovenia

# ABSTRACT

The increasing frequency of password usage in the digital age demands advanced techniques for password cracking. In this paper, we present sociological insights into password creation and common password patterns while considering the vulnerabilities commonly exploited by attackers. We discuss a few known password cracking techniques such as brute-force attacks, Rainbow tables and Markov-based models, with a special focus on password cracking by using contextual dictionaries, which is presented in the main article. We present tools like DBpedia and natural language processing to explain how authors made and improved special contextual dictionaries that match certain sets of information. Their methodology involved the extraction of thematic concepts from Wikipedia articles and the ranking of keywords based on their thematic distance from seed words. The results of their research are presented, highlighting the performance of customized contextual dictionaries compared to generic dictionaries like Ignis-10M.

# **KEYWORDS**

password cracking, contextual dictionary, natural language processing

# **1 INTRODUCTION**

Throughout history, passwords have been essential for controlling access to secure areas and sensitive information. In medieval times, knights would use secret codes or phrases as passwords to gain entry to castles or fortresses during sieges. Similarly, secret societies and organizations have long employed passwords as a means of verifying membership and maintaining confidentiality.

Usage of passwords in connection with computers dates back to 1961, when the first operating system that required password login was introduced at MIT. That was the Compatible Time-Sharing System, or CTSS. Similarly, the Multics (Multiplexed Information and Computing Service) operating system, which was released in 1967 and developed collaboratively by MIT, General Electric, and Bell Labs, also employed password-based authentication for user logins. These early systems laid the foundation for modern password-based authentication methods in computing. Since then, passwords have remained a fundamental aspect of computer security, evolving alongside advancements in technology to address emerging threats and challenges. Dmitar Zvonimir Mitev dm42852@student.uni-lj.si Faculty of Mathematics and Physics Faculty of Computer and Information Science University of Ljubljana Slovenia

In today's digital age, passwords are everywhere. They act as the main shield for personal accounts, private data, and confidential information. Whether it is signing into email, unlocking phones, or managing online banking, passwords are crucial for protecting privacy and security in our highly connected world.

In the field of digital forensics, cracking passwords is not just a tool for malicious activities but a crucial method for uncovering vital evidence. Whether it is unraveling the details of a cybercrime or accessing encrypted data for investigative purposes, proficient password cracking techniques are crucial in forensic investigations. These methods serve as a gateway to unlocking crucial insights and piecing together the puzzle of digital evidence, ultimately easing the pursuit of justice in our digitized society.

1.0.1 Overview. In section 2, we present the common patterns and reasoning behind creating passwords. The role of dates in passwords is mentioned and the problem of reusing passwords is presented. In section 3, we present some password cracking techniques. Besides the brute-force attack, Rainbow tables and Markov-based models are explained. After that, we focus on contextual dictionaries. In section 4, some crucial tools for creating them are described. Section 5 focuses on the research made in [3]. Methodology and main results are summarized. We conclude the article in section 6 by discussing our key takeaways.

# 2 CHOOSING PASSWORDS

In this section, we focus on the sociological aspect of passwords. Predictable frequent practices of users are presented, based on some of the researches mentioned in [3].

#### 2.1 RockYou leak

In 2009, website RockYou.com was hacked and a collection of more than 30 million stolen passwords was leaked. This gave researchers an oppurtinity for analysis.

The most common password in the dataset was "123456", representing about 0.9 % of all passwords. It was followed by sequences "12345" and "123456789". Besides such digit sequences, the list of 20 most common password choices consists of dictionary words (e.g. "password", "princess", "monkey"), names ("daniel", "jessica"), common phrases (e.g. "iloveyou", "rockyou") and keyboard patterns (e.g. "qwerty"). Interestingly enough, the ten most common passwords represent more than 2% of the database.

#### 2.2 Common patterns

In order to explore why patters like this appear, the authors in [10] performed an experiment with 49 participants. Those participants were asked to create passwords for a few fictitious websites and explain their reasoning behind it. Authors observed some predictable human behaviour while creating passwords. The passwords the participants created were often derived from the name of the website or were closely related to its contents (for example building a password with a base word "money" for an online banking account). However, by using unrelated words and unique phrases, some participants managed to create secure passwords.

Some of the participants actually decided to use less safe passwords for low-value accounts while others created weak passwords, but assumed that they were secure enough. Participants falsely believed that they automatically made their passwords more secure by:

- adding a symbol or a digit at the end of the password (especially if the password would be too short otherwise);
- using words that are more difficult to spell;
- using common keyboard patterns ("qwerty", "asdfghjk", "yaq1xsw2");
- capitalizing the first letter of each word;
- replacing letters with similarly looking characters (e.g. writing "@" instead of "a" or "3" instead of "e");
- using personal information that they haven't published online before (for example pet's name or even birth date).

It should be noted that adding digits, symbols and capitalized letters to a password does not automatically make it more secure, which turned out to be a common misconception amongst participants.

#### 2.3 The role of dates

A study [12] focused particularly on the role of the dates in passwords. The database of leaked RockYou passwords was also analysed. Over 15 % of passwords contained digit sequences of length between 5 and 8 and authors concluded that about 38 % of them could classify as dates.

In general, years turn out to be a popular choice for fourdigit passwords. Years after 1969 are more often used. If the required password length is 8 characters, users frequently simply duplicate the year. Besides years, dates which correspond with significant events such as birthdays and anniversaries are common choice. Dates that represent holidays, for example Valentine's day, Christmas and New Year, appear often too. In false belief of providing additional security, users frequently spell out the name of month while using a particular date as a password.

#### 2.4 Reusing passwords

Another security risk is reusing passwords. People tend to do it to cope with the difficulties of having to remember too many passwords. In [10], two different studies are mentioned that report high level of reusing passwords. Participants reported that they haven't yet had any negative experience deriving from such behaviour. They considered the passwords that they reuse to be secure.

## **3 PASSWORD CRACKING TECHNIQUES**

Password cracking is the practice of exploiting weak or compromised passwords for gaining unauthorized access, employing various techniques for such purposes. However, beyond illicit activities, password cracking methods can also be indispensable in aiding criminal investigations. Such approaches prove significant in digital forensic undertakings. In this section, we take a look at some of the most common password cracking techniques.

## 3.1 Brute-force attack

The term "brute force" refers to attackers using exhaustive trial-and-error methods to gain access to user accounts. Despite being an old cyberattack method, brute force attacks are tried and tested and remain a popular tactic with hackers.

Apart from the traditional approach, where an attacker tests every possible combination until a match is found, there are also a few other ones that are considered brute-force. We briefly mention two such.

- Dictionary attacks: an approach where hackers are utilizing a precompiled list of phrases, often derived from research on the target or variations of commonly used (or potentially used) passwords, which are then tested against a specific username. This list, referred to as a "dictionary", comprises modified or slightly altered words and character combinations.
- Hybrid Attacks: A technique involving the combination of traditional brute force attacks with dictionary attacks. The hacker selects the most prevalent phrases and words from the "dictionary" and tries multiple iterations of possible passwords until finding a successful combination.

#### 3.2 Rainbow Tables

Rainbow tables were first introduced in [6] as an application of an earlier, simpler algorithm [1]. They represent a precomputed table for caching the outputs of a hash function. Passwords are usually stored not in plain text form but rather as hash values. Therefore, if such a database of hashed passwords falls into the hand of a hacker, they may use a precomputed rainbow table to recover the plain text passwords. A common defense against such attacks is to compute the hashes using a key derivation function that adds a salt to each password before hashing it, with different passwords receiving different salts. The salts are stored in plain text along with the hash.

Rainbow tables are an example of a space-time tradeoff: they require less computer processing time and more storage than a brute-force attack which calculates a hash on every attempt, but more processing time and less storage than a simple table that stores the hash of all possible passwords.

#### 3.3 Markov-based models

In most languages, certain symbols are naturally followed by certain other symbols. In English, the bigram "in" is highly frequent, and many nouns have plural forms ending with "s". Hence we can use this knowledge, to extract the most probable next symbol in a word, a phrase, or a password.

An interesting approach for password cracking, based on the above, is presented in [11] with the use of Markov models. A Markov model is a sequence of events, usually called states, for which the probability is dependent only on the event immediately preceding it [7]. For example, if the current state is A, the probability that the next state will be B is based only on A and we can assign probability to it. With two additional states that represent the beginning and the end of a word or phrase, the START and the STOP state, and with the added knowledge of frequent bigrams, we can construct a model that produces the most likely next character. A simplified version can be seen in Figure 1 in which there are only two letters used: A and B, and the sequence length is limited to 2. This approach can also be generalized so that the next state depends on the previous s states instead of just the current one.



Figure 1: A simplified version of a start-end Markov model. The drawing is from [11].

Using three different datasets: an English dictionary, a real password list and a generated list of parts of words (near words), the authors experimentally prove, that the 'e'  $\rightarrow$  'r' transition is very likely, many passwords start with 'E', 'O' or 'A' and many passwords end with an 'e'.

#### 4 CONTEXTUAL DICTIONARIES

The distribution of passwords is not random and passwords often contain personal information about the user. The term "mangling rules" is used in [3] to describe the common human behaviours and substitutions. Ubiquitous practices, such as adding digit sequences to the end of the passwords, were already presented above.

Combining that knowledge with information, acquired with analysis of datasets of leaked passwords, and personal information of particular user, custom dictionary lists can be created. They represent potential passwords of a targeted user and can therefore succeed even when non-personalized ways of password cracking normally fail.

The main source [3] focuses specifically on creating such custom contextual dictionaries. For better understanding of the article, some crucial concepts and tools are presented in this section.

## 4.1 DBpedia

Wikipedia is among the ten most visited websites in the world and contains articles in more than 280 different languages. However, it can be challenging to extract particular structured data from those articles.

As presented in [4], the DBpedia project was created in 2007 in order to build large knowledge databases by extracting data from Wikipedia articles in 111 languages. On their website<sup>1</sup> it is written, that it is a "crowd-sourced community effort to extract structured content from the information created in various Wikimedia projects." Using those databases, finding answers to particular queries (such as: Italian composers born in the 18th century) is facilitated.

Wikipedia articles consist of text and different types of structured information such as categorization information, images, links to external web pages, redirects between Wikipedia articles and geo-coordinates. The DBpedia extraction of such article consists of four phases: input, parsing, extraction and output. Firstly, Wikipedia article is read, either from a Wikipedia dump or directly fetched from a MediaWiki installation using the MediaWiki API. After that, each article page is parsed by the wiki parser. It transforms the source code of a Wikipedia page into an abstract syntax tree. This abstract syntax tree is then consumed by an extractor. DBpedia provides different extractors for different purposes. For example, feature extraction uses extractors that are created for extracting a single feature from a page, such as redirects to other pages. Extractor yields a set of  $RDF^2$  (resource description framework) statements. Those statements are then written in different formats.

Regarding [3], the most important is NLP extraction. Such extraction yields datasets that are created to support natural language processing. One of such datasets is the thematic concept dataset. Its goal is to capture the idea of a "theme" of the title subject of the article, based on the fact that many Wikipedia categories are linked to the article that describes the main topic of that category.

#### 4.2 Natural language processing and word2vec

The goal of natural language processing (NLP) is to give computers the ability to support and manipulate human language, to "understand" the contents of documents. This is achieved using probabilistic or rule-based machine learning approaches which process natural language datasets.

<sup>&</sup>lt;sup>1</sup>https://www.dbpedia.org/

 $<sup>^2\</sup>mathrm{In}$  general, RDF statements express relationships between the resources and are used to integrate data from multiple resources.

The representation of a word (usually with a real-valued vector) is called a word embedding. Those values encode the meaning of the words and words that are close in created vector space could be similar in meaning.

In NLP, word2vec is a group of related models that are used for obtaining word embeddings. Word2vec was developed by Tomáš Mikolov and colleagues and published in 2013 [5]. In it, each word is represented as a high-dimensional vector and words that seem to be contextually similar appear closer in vector space.

The toolkit **wikipedia2vec**<sup>3</sup> is a Python-based open-source toolkit for obtaining, learning and visualising the embeddings of words or entities from Wikipedia. Embeddings can be trained with Wikipedia dumps as an input, however there are already 12 existing embeddings available, for 12 different languages.

#### 4.3 Hashcat

Hashcat [8] is an advanced password recovery tool that utilizes brute-force attacks, dictionary attacks, rule-based attacks and hybrid attacks to crack hashed passwords. Hashcat operates by generating potential plaintext passwords and then hashing them using the same algorithm as the original hashed password. It then compares the resulting hash values with the target hash to check for a match. Until 2015, it operated with a proprietary code base, but has since transitioned into open-source software [9]. Versions are available for Linux, macOS, and Windows.

Examples of hashcat-supported hashing algorithms are MD4, MD5, SHA-family and Unix Crypt. This makes it versatile for cracking passwords stored in different formats and systems. Moreover, hashcat is highly optimized to leverage the parallel processing power of modern GPUs and CPUs, enabling fast password cracking.

From an academic point of view, hashcat serves as a tool for studying password security, allowing researchers and analysts to evaluate the strength of passwords and the effectiveness of different hashing algorithms. It also highlights the importance of utilizing robust password hashing algorithms and the potential vulnerabilities associated with weak passwords.

## 5 ARTICLE OVERVIEW

In this section, we focus on research and findings of [3]. In the article, the authors present a methodology for optimization of customized contextual dictionaries and the ranking techniques used to create them. Example scenario of a digital forensics triage is described, followed by an elaborate discussion on uses, benefits and limitations of using context dictionaries in context of password cracking.

#### 5.1 Methodology

Contextual dictionaries prove to be useful when the investigator is trying to access the encrypted device of a suspect. Based on the suspect's interests, a personalized dictionary can be created, using those interests as seed words (for example "tennis" or "jazz").

Minding the ethical standards and data protection, testing contextual dictionaries was done on communities' passwords rather than individuals'.

The authors chose to use leaked databases of 4 different websites, which can be seen in Table 1. These databases only contained passwords, not providing any information about users (such as usernames or emails).

Dataset	Number of passwords
AxeMusic	252,752
JeepForum	239,347
Wattpad	23,531,304
MangaTraders	618,237

Table 1: Sizes of used databases.

In order to determine whether such contextual dictionaries perform better than other already existing dictionaries, authors used dictionary Ignis-10M<sup>4</sup> for comparison. Ignis-10M is a collection of 10 millions of leaked passwords from various websites that was assembled in 2020. Using the database of leaked passwords from RockYou was considered, but the authors opted for Ignis-10M because it is more recent.

5.1.1 Creating contextual dictionaries. The method for creating contextual dictionaries for each of the four datasets in Table 1 followed [2]. The main tool used is DBpedia. Using it, structured information from Wikipedia is extracted. For building contextual dictionaries, the list of all the links from particular Wikipedia artucle is used. That is due to the fact that links in Wikipedia articles are mostly keywords or key phrases, closely related to the contents of the article.

In order to create a contextual dictionary, a seed word that is thematically connected to the targeted user (or in our case, community) is needed. The corresponding DBpedia's entry is visited and all the links on that entry are collected. Those entries are called "layer 1" entries because, as mentioned, those are keywords and key phrases that are in general closely related to the given seed word.

For example, the chosen seed word for the leaked database of the website AxeMusic was "music". If the Wikipedia page for keyword "music" is inspected, the first few links are keywords "sound", "form", "harmony", "melody" and "rhythm" and key phrases "expressive content", "musical composition", "musical improvisation" and "human voice".

In the second step, all those entries are visited and for each of them, all the links are again collected. Collected keywords now form "layer 2". In general, the deeper the traversal, the less the keywords will be thematically connected to the original seed word.

It is of course possible that an article, corresponding with a keyword from layer 1, contains a link back to the original seed word or any other keyword from layer 1. The same holds for deeper layers. Such duplicated keywords are removed from the final dictionaries, as well as possible picture links.

<sup>&</sup>lt;sup>3</sup>https://wikipedia2vec.github.io/wikipedia2vec/

<sup>&</sup>lt;sup>4</sup>https://github.com/ignis-sec/Pwdb-Public/

In [3] the authors estimated layer 3 to be sufficient for databases AxeMusic, JeepForum and Wattpad and layer 4 for MangaTraders, since layer 3 provided considerably less keywords than the ones for other three datasets. The sizes of created contextual dictionaries can be seen in Table 2.

Dataset	Seed word	Size of created dictionary
AxeMusic	music	1,001,173
JeepForum	car	853,825
Wattpad	fanfiction	641,007
MangaTraders	manga	6,348,947

Table 2: Sizes of contextual dictionaries.

5.1.2 Ranked dictionary based on thematic distance. To improve existing contextual dictionaries, additional, ranked versions were created. In order to evaluate the thematic distance among the seed word and found keywords, the NLP model wikipedia2vec was used. As already mentioned in section 4, it is an NLP model based on word2vec for obtaining vector representations of words.

Two approaches were taken in this case:

- the similarity score of a phrase was computed as an average of similarity scores of individual words in the phrase,
- (2) the similarity score of a phrase was equal to the maximal of the similarity scores of individual words in the phrase.

In that manner, two rankings of dictionaries were created. The first ranked list contained phrases as well and in the other, those phrases were split into individual words, where repeated words and stopwords (such as "the", "is" and "and") were omitted.

Once the similarity score was calculated, words in the dictionary were ranked according to their connection with the seed word. The words higher in the list will be checked first, but more importantly, more permutations of them using mangling rules will be tested.

5.1.3 Quality of the dictionary list. When estimating the quality of a created contextual dictionary list, the conditions of the particular case should be taken into consideration. For example, one larger dictionary may have a higher success rate, but also take much more time. It could also be the case that two dictionaries have a similar success rate in similar time, but one finds passwords of higher complexity than the other.

5.1.4 Example scenario. Having created customized dictionaries, a fictitious digital forensics triage scenario was determined – the investigator must access the data on the encrypted device in the shortest time possible. The execution time was limited to 15 minutes.

The goal of this experiment was to evaluate how well contextual dictionaries perform compared to the general Ignis-10M dictionary. Before the actual password cracking, all dictionaries were altered using rule set OneRuleToRuleThemAll<sup>5</sup> with mangling rules taken into account. For password cracking the tool hashcat was used.

#### 5.2 Results

While evaluating contextual dictionaries containing only keywords and contextual dictionaries containing phrases as well, it turned out that the dictionaries containing only individual words performed better. For that reason, only those were taken into account.

The authors reported that the dictionary Ignis-10M had the best performance over time, which is not that surprising considering the fact that it is the largest and the most diverse dataset. However, custom contextual dictionaries performed well too. The total number of passwords found using each of those three dictionaries can be found in Table 3.

Dataset	Ignis-10M	Ranked dict.	Unran. dict.
AxeMusic	132,009	106,782	86,384
JeepForum	122,061	107,365	89,001
Wattpad	4,103,525	3,016,762	2,367,223
MangaTraders	352,544	289,573	231,097

#### Table 3: Total number of passwords found.

It is also worth mentioning the number of passwords found exclusively by (ranked or unranked) custom dictionaries, i.e. the ones that Ignis-10M did not find. This can be seen in Table 4.

Dataset	Ranked dict.	Unranked dict.
AxeMusic	7,773	2,698
JeepForum	6,025	2,212
Wattpad	268,670	86,267
MangaTraders	22,128	9,635

Table 4: Number of passwords found exclusively by ranked or unranked dictionaries.

5.2.1 Strength of found passwords. Besides the percentage of successfully cracked passwords, their complexity should be taken into account as well. In order to estimate the difficulties of cracked passwords, authors used password strength meter **zxcvbn** [13]. This strength meter categorizes passwords into 5 categories. Class 0 is the class of easiest passwords and class 4 contains the most complicated ones. Passwords in classes 0, 1 and 2 are considered to be easier to crack and most current password cracking techniques should succeed in cracking them.

Password strength analysis on all four leaked databases showed that most cracked passwords (more than 70%) fall into class 1 and class 2. The authors then focused on passwords in class 4, that is the strongest passwords. On average, 50% of those passwords were found exclusively using custom contextual dictionaries. This information encouraged the authors to perform a combined attack, that is an attack that combines the results of custom contextual dictionaries and the results from Ignis-10M. The success rate of such approach was between 20 and 52.2 % higher than the previous attempts.

 $<sup>^{5} \</sup>rm https://not so secure.com/one-rule-to-rule-them-all$ 

5.2.2 Authors' conclusions. It should be emphasized that contextual dictionaries are highly customizable as any word or a phrase that has an existing Wikipedia article can be taken as a seed word. Additionally, the investigator can choose how many layers will be explored and how high the similarity threshold will be. It is also possible to use several seed words and create combined contextual dictionaries. Another advantage of this approach is that once the dictionary is created, it can be reused. This especially holds for ubiquitous themes, so the investigators could reuse already created dictionaries.

However, if runtime and the difficulty of cracked passwords are not taken into account, large generic dictionaries that contain passwords from several leaked websites will perform better and the total number of found passwords is expected to be higher. Another mentioned problem is that many words with high similarity score in created dictionary are not likely to be words chosen for passwords. For example, if the seed word is "manga", the word "series" has 0.54 similarity score, but it is not likely to appear in a password.

#### 6 CONCLUSION

In this article, we looked into sociological aspects of choosing passwords. We observed that some common patterns occur when people are choosing their passwords and later explored how this fact can be exploited when cracking passwords. Then, we looked into common password cracking techniques, including brute-force attack, rainbow tables and Markovbased models. Furthermore, we focused on the password cracking technique that was presented in [3], namely password cracking with usage of contextual dictionaries. In order to deepen the knowledge on the subject, we inspected some tools used by the authors. Lastly, the content of the main article was presented. We looked into the authors' methodology and the obtained results. To summarize, considering custom context dictionaries, especially ranked ones, when cracking passwords proves to be valuable, because the link among the contents of the website and the corresponding passwords is undeniably present.

A good starting point for future work could be filtering out password candidates, such that those with high similarity to the seed word but low probability of being used as passwords are removed. Another option is to consider dictionary attacks for languages that have grammatical cases. An idea is to include the words and phrases in the dictionary in all possible cases. For example, in Slovenian, this would mean that each word or phrase would be added in the forms of all six cases: nominative, genitive, dative, accusative, instrumental, and locative. This way, the negative implications of grammatical cases would be mitigated.

#### REFERENCES

- M. Hellman. 1980. A cryptanalytic time-memory trade-off. IEEE Transactions on Information Theory 26, 4 (1980), 401-406.
- [2] Aikaterini Kanta, Iwen Coisel, and Mark Scanlon. 2022. A Novel Dictionary Generation Methodology for Contextual-Based Password Cracking. *IEEE Access* 10 (2022), 59178–59188.

- [3] Aikaterini Kanta, Iwen Coisel, and Mark Scanlon. 2023. Harder, better, faster, stronger: Optimising the performance of contextbased password cracking dictionaries. Forensic Science International: Digital Investigation 44 (2023), 301507. Selected papers of the Tenth Annual DFRWS EU Conference.
- [4] Jens Lehmann. 2012. DBpedia A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. https://svn.aksw. org/papers/2013/SWJ\_DBpedia/public.pdf. [Online; accessed 8. May 2024].
- [5] Tomas Mikolov, G.s Corrado, Kai Chen, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. 1–12.
- [6] Philippe Oechslin. 2003. Making a Faster Cryptanalytic Time-Memory Trade-Off. In Advances in Cryptology – CRYPTO 2003, Dan Boneh (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 617–630.
- [7] L.R. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* 77, 2 (1989), 257–286.
- [8] Jens Steube and Gabriele Gristina. 2009. Hashcat. https:// hashcat.net. Accessed: 2024-05-11.
- [9] Jens Steube and Gabriele Gristina. 2015. Hashcat github repository. https://github.com/hashcat/hashcat. Accessed: 2024-05-11.
- [10] Blase Ur, Fumiko Noma, Jonathan Bees, Sean M. Segreti, Richard Shay, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. 2015. "I Added '!' at the End to Make It Secure': Observing Password Creation in the Lab. In *Eleventh Symposium On Usable Privacy* and Security (SOUPS 2015). USENIX Association, Ottawa, 123– 140.
- [11] Renier van Heerden and Johannes Vorster. 2008. Using Markov Models to Crack Passwords. 3rd International Conference on Information Warfare and Security.
- [12] Rafael Veras, Julie Thorpe, and Christopher Collins. 2012. Visualizing semantics in passwords: the role of dates. In Proceedings of the Ninth International Symposium on Visualization for Cyber Security (Seattle, Washington, USA) (VizSec '12). Association for Computing Machinery, New York, NY, USA, 88–95.
- [13] Daniel Lowe Wheeler. [n.d.]. zxcvbn. https://github.com/ dropbox/zxcvbn. Accessed: 2024-05-11.



Izzivi

# 6 - Forenzika brskalnikov / Browser Forensics

# **Forensic Browser Memory Analysis**

Daniel Brommer Karlsruhe Institute of Technology Karlsruhe, Germany daniel.brommer@student.kit.edu

### ABSTRACT

Internet browsing activity generates vital evidence for forensic investigations. The browsing history is particularly significant in this context. However, private browsing modes prevent the creation of cache and history records, which complicates finding evidence substantially. This report reviews methods used in forensic analysis to address this challenge.

# **KEYWORDS**

Digital forensics, Volatile data, Memory forensics, Web browser, User activity, Physical memory

### **1** INTRODUCTION

Nowadays, most individuals use the Internet extensively for communicating, acquiring information and planning. Therefore, web browsing activity has become a valuable source of information for investigators. Websites visited, search queries entered and files downloaded can assist in establishing a timeline of events and can provide evidence of criminal activity.

#### 2 BROWSER MEMORY ANALYSIS

#### 2.1 Traditional methods

The conventional method of obtaining browser data assumes that the suspect uses standard browser tabs. In this mode, the browser caches a webpage and its associated images to the hard drive upon the first visit to the page. Additionally, a history of visited websites is typically stored in a database format like SQLite, the specific location of which varies depending on the browser and its version. Details such as the creation dates of cached images, search queries and URLs, along with timestamps in the browsing history, can reveal the intentions of the suspect and provide vital evidence for the investigation.

However, when individuals use private browsing mode, traditional investigation techniques prove inadequate. Private modes prevent the creation of local data traces on the hard disk like history and cache. Instead, data is loaded into RAM, where it is likely lost forever once the computer is shut down.

The likelihood of data recovery from private browsing depends on the state in which the computer is discovered at the crime scene. If it is found powered on and operational, a memory dump of volatile memory can be taken, increasing the chances of finding evidence significantly. But if the computer is shut down, live forensic techniques are not applicable and the swap file might be the only chance. Theresa Kleinert Universität Freiburg Freiburg, Germany kleinert@informatik.uni-freiburg.de

# 2.2 Live forensics

Live forensics involves gathering volatile data and examining devices while they are still operational. The system's RAM content and running processes can provide a wealth of information, including browsing activity. On modern systems, a memory dump of the RAM content has a size of gigabytes, which is impractical to search through manually. Several methods for analyzing the memory dump have been proposed.

As outlined in [4], one approach involves searching for strings like "https://" or "search?q=". Another method proposed in the same publication is pattern-matching a format used by Windows for storing searched keywords. However, directly searching within a memory dump poses some problems.

Firstly, the presence of a URL in the dump does not necessarily imply that the webpage has been visited. URLs can also appear because of background processes or prefetching. The latter technique is used to accelerate the browsing experience by caching webpages and images before the user actually clicks on them. For instance, an examination of our own memory revealed foreign news webpages, likely part of a preloaded collection for accelerated browsing. Lacking contextual information, mere string-searching cannot confirm actual visitation.

Secondly, simple string searches fail to extract contextual information like timestamps. Analyzing our own memory dump, we found websites visited in past sessions, dating back several days. Since timestamps significantly affect the strength of evidence, it is very valuable to have this context. For example, a Google search for "arsenic" holds vastly different implications depending on timing random versus just before a suspected poisoning.

Lastly, identifying relevant keywords poses a significant challenge. Given the unstructured and vast nature of dump files, searches must be based on suspicions. A simple search for "https://" will likely yield an overwhelming number of hits, indicating the need for strategic keyword selection.

In section 3, we will discuss a tool that addresses those problems.

#### 2.3 Offline forensics

Offline forensics is employed when the computer is discovered turned off at the crime scene. In such instances, RAM content cannot be collected, but a bit-stream copy of the hard drive can be obtained. Since private browsing modes do not save data to the hard drive, this seems useless, but there remains a nonzero chance of discovering traces [6]. The first of two possibilities arises from modern memory management. If the physical RAM of a machine is insufficient for the current workload, the operating system may swap out less frequently used data from RAM to the swap file on the hard drive. This can potentially leak private browsing data to the disk, making it available even after shutdown.

On Windows systems, a second option involves checking for the existence of a hibernation file. Hibernation mode takes a snapshot of RAM before powering off, allowing users to resume their activities upon startup without delay.

While both file locations are worth checking, it is not guaranteed that evidence will be found. For that, the user must have used the computer under adequately high load to use the swap file, and the system must be configured accordingly. For example, hibernation is usually not enabled by default, the swap file can be reduced in size or the system can be configured to clear the swap file on shutdown.

#### **3 CHRACER**

Chracer is a proof-of-concept tool developed to extract users' web browsing-related artifacts from the virtual memory of any Chromium-based browser [3]. It is able to obtain browsing-related artifacts of users browsing in the incognito mode of Chromium based browsers. According to the chromium project, the incognito mode enables users to browse without creating a persisting history [5].

Chracer was developed with the goal of rendering string search based approaches obsolete. As described in Section 2.2, the string search based approaches suffer from the inability to automatically extract the context in which a searched string appears. Chracer's approach is to extract information about the browser and its history by analyzing the object layout stored in the virtual memory.

#### 3.1 Background

An operating system allocates a virtual address space to each process, defining the virtual memory accessible by that process. The operating system translates these virtual address spaces into physical address spaces. This mechanism ensures that processes cannot interfere with the memory of other processes, and it allows them to use more memory than physically available. Given that Chromium is mainly developed using C++, the virtual memory of a Chromium process contains C++ memory objects [1]. While compiling C++ code, an *object layout algorithm* is used to map fields used in C++ classes to machine-level memory. If the structure of the underlying C++ class is known, the fields of this class can be extracted from the memory.

G. Choi et al. identified the following key browsing-related C++ classes from the publically available Chromium source code to be of forensic value:

- (1) Browser Used to manage a single Chromium window
- (2) TabGroup Used to represent a group which can contain multiple tabs
- (3) Tab Used to render the web page of a visited URL

(4) NavigationEntry Used to store information about a visited URL

Figure 1 shows the relation between the key classes and the forensically most important fields they contain.

The sizes of Standard Template Library (STL) C++ objects are known; for instance, a *std::string* has a size of 24 bytes. Chracer connects these known sizes with the Chromium source code to determine which object resides where inside the *Browser* object.

When building Chromium from source on Windows, a Program Database (PDB) symbol file is generated. The PDB contains debugging information about the built program, including the object layout and function names. Using a tool like Ghidra [2], the PDB information about the object layout can be parsed into a readable XML file. Chracer uses this XML file to obtain the object layout of the built Chromium program to determine the location of each C++ object in memory.

### 3.2 Chracer operation

Chracer combines all the previously discussed concepts to extract the Chromium browser history of a virtual memory dump. In essence, Chracer tries to carve the Browser object of the virtual memory dump. If the Browser object is found, all browsing related information can be extracted by following the structure of the underlying C++ class Browser. The carving is done by inspecting all objects present in the Virtual Address Descriptor (VAD) tree. The VAD tree is a hierarchical data structure that organizes the virtual memory space of a process into regions and their object representations. The structure of the Browser object is accessible through the XML files generated by Ghidra. Every object in the VAD tree is tried to be validated as the Browser object by validating all fields present in it according to the Browser class. Because the Browser class does not only contain basic data types, but other classes as fields, the validation will be done recursively on every class that is present as a field.

When the *Browser* object is found, the browsing related data can be extracted. Because the whole *Browser* object is found, Chracer is able to extract more context related information compared to traditional string search based approaches. Figure 1 shows forensically relevant fields that can be extracted from the *Browser* object. This includes the title of a visited website, the URL of a visited website, the favicon and its origin, information about the used SSL certificate and the time when the website was accessed. All this information can be extracted even if the incognito mode of Chromium is used.

#### 3.3 Results

Chracer shows promising results, as it is able to carve the *Browser* object out of the virtual memory of a Chromium process and extract the data contained in the *Browser* object. It correctly identified used tab groups and the tabs included in a Chromium window. The *Browser* class contains the field *profile\_*. This field indicates, whether the Chromium window was opened using incognito mode or not. In the tests conducted by the authors of the paper, Chracer was able to correctly determine which Chromium windows have been



Figure 1: Key browsing related classes and their fields of most forensic value in the Chromium source code [3]

opened using incognito mode and which ones have been opened without the incognito mode.

#### 3.4 Limitations

Even though Chracer is able to extract forensically relevant information from the virtual memory of a Chromium based browser process, there are some limitations. One limitation is that browsers that are Chromium based, like Google Chrome, Microsoft Edge and Brave, can change the classes related to the Browser class. This will invalidate the XML file based on the PDB file, because the offsets of certain fields will differ from a specific implementation and the implementation used by Chromium. One example for this is the field title\_ in the class NavigationEntryImpl. In the object layout generated by the open source Chromium implementation, this field has an offset of 0xE0. The analysis of the virtual memory of the object layout of a Google Chrome (v111.0.5563.65) process shows that in this implementation the offset of the same field is 0xC8. Since the source code of popular Chromium based browsers is not publicly available, these offsets need to determined and updated manually. Note that these offsets could change depending on the version of the investigated browser. Therefore, future research is needed on how to automatically determine the offsets of forensically relevant fields and classes.

Another weakness of Chracer is its performance. Because it tries to validate every object in the VAD tree, the search for the *Browser* object will result in a long execution time. Future development should focus on how to speed up this process and how to validate checked objects faster.

#### 4 CONCLUSION

We have seen that private browsing significantly reduces the likelihood of finding evidence. Investigators have to rely on fortunate circumstances such as seizing the computer in a powered-on state or finding browsing data within the swap file. We have reviewed techniques which can lead to data findings, but also might not.

We want to highlight that it is already helpful to recognize in which cases private browsing has likely destroyed all evidence, as this indicates that resources are better allocated to investigating data elsewhere. For example, getting authorization for and looking through ISP logs.

#### REFERENCES

- [1] 2024. Chromium. https://source.chromium.org/chromium
- [2] 2024. Ghidra. https://ghidra-sre.org/
- [3] Geunyeong Choi, Jewan Bang, Sangjin Lee, and Jungheum Park. 2023. Chracer: Memory analysis of Chromium-based browsers. Forensic Science International: Digital Investigation 46 (2023), 301613. https://doi.org/10.1016/j.fsidi.2023.301613
- [4] S Dija, J Ajana, V Indu, and M Sabarinath. 2021. Web Browser Forensics for Retrieving Searched Keywords on the Internet. In 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N). 1664–1668. https://doi.org/10.1109/ICAC3N53548.2021.9725457
- [5] The Chromium Projects. 2024. Incognito. https://www.chromium.org/userexperience/incognito/
- [6] Huwida Said, Noora Al Mutawa, Ibtesam Al Awadhi, and Mario Guimaraes. 2011. Forensic analysis of private browsing artifacts. In 2011 International Conference on Innovations in Information Technology. 197–202. https://doi.org/10.1109/ INNOVATIONS.2011.5893816

# Cross-platform cloud data access through browser credential migration

Vid Smole University of Ljubljana, Faculty of Computer and Information Science Ljubljana, Slovenia vs1481@student.uni-lj.si Vid Purgar University of Ljubljana, Faculty of Computer and Information Science Ljubljana, Slovenia vp0863@student.uni-lj.si

# ABSTRACT

Browser credential migration has become a pivotal focus in digital forensic investigations, driven by the widespread adoption of cloud services and the critical role of web browsers in accessing sensitive information. This article offers a comprehensive examination of browser data migration methodologies, challenges, and practical solutions for transferring user information across different operating systems.

#### **KEYWORDS**

Browser, Cookies, Firefox, Chrome, Chromium, DPAPI, keychain

# **1 INTRODUCTION**

In today's digital landscape, the secure management and transfer of browser credentials have become very important. With the proliferation of online activities and the increasing reliance on web browsers for accessing sensitive data, the need to effectively migrate browser credentials between systems has emerged as a critical concern for users and forensic investigators.

The ubiquity of personal and confidential information stored within browsers, including cookies, passwords, and browsing history, underscores the importance of secure storage for such information. Understanding the methodologies employed by browsers for key management, such as the Data Protection API (DPAPI) on Windows and keyrings on Linux, is essential for forensic investigators tasked with retrieving and analyzing browser data.

With many browsers and operating systems available, the number of combinations a user might use grows quickly, but the investigator needs to be able to access browser information regardless of the exact configuration used by the user. For this purpose, this article also describes the process for moving the cookie database between computers for Firefox and Chromium-based browsers, both on Windows and Linux operating systems. Furthermore, to make the process as easy as possible, we also developed an automated script for decrypting and encrypting the database on Chromiumbased browsers.

#### 2 RELATED WORK

Browser credential migration has been a subject of forensic investigations ever since cloud service became a part of our everyday routines. There are many existing articles that investigate the forensic potential on different operating systems and with different ways of storing user information There is, however, no definitive approach to migrating user information across different operating systems.

Uk Hur et al. [11] propose a new method to migrate the credentials stored by the web browser to other devices and effectively utilize them. The article explores the methodology and implications of browser data migration in digital forensic investigations. It begins by detailing encryption methods used in various browser versions, particularly focusing on AES-GCM and the CryptProtectData and CryptUnprotectData DPAPI functions. Migration processes for different browsers are outlined, including generating and encrypting keys for data transfer. Notably, some browsers lack encrypted credentials, simplifying migration. The article highlights the breadth of data accessible through migration, spanning cloudstored content, user locations, email exchanges, and social media activity. Additionally, it discusses data obtainable from email, cloud storage, productivity tools, and social networking services. The main takeaway of the article is that they verified that it is possible to log in and collect data on 20 types of web services that are frequently used with migrated credentials.

Hyunji Chung et al. [5] discussed the importance of forensic investigation of cloud storage services in cases involving data breaches or leaks. It presents a detailed examination of various cloud storage platforms such as Amazon S3, Dropbox, Evernote, and Google Docs, outlining the artefacts left behind on Windows, Mac, iOS, and Android systems after using these services. They then presented a hypothetical case study involving a suspect leaking a confidential document is presented to illustrate the significance of cloud storage analysis in forensic investigations. Despite initial lack of evidence on the suspect's personal computer, artefacts found on their Dropbox account revealed the leaked document. The article highlights the importance of adapting investigation techniques to include cloud storage services across different operating systems, enhancing the ability of forensic examiners to gather evidence effectively in cases involving cloud-based data.

In addition to these articles, there are also some projects, that were made on the topic. The first started as a question on Stack Overflow [9], which then evolved into a Github project [10]. The repository presents a Python library aimed at simplifying the process of accessing and decrypting browser cookies for websites visited with the Google Chrome and Chromium browsers. It provides functions to extract cookies from the browser's storage and convert them into a format that can be easily used in Python scripts. The repository is designed to work with Python 2 and Python 3, it is also compatible with Linux and macOS operating systems, support for Windows is not yet added.

There is also a blog post [3] by user Apra, in which the author delves into decrypting browser credentials stored locally by Google Chrome and Mozilla Firefox in Windows environments. The author created HarvestBrowserPasswords.exe [4], a tool to extract and decrypt credentials from both Firefox and Chromium-based browsers. For Google Chrome, the tool locates Chrome profiles, extracts encrypted login data, and decrypts passwords using DPAPI. The article also outlines the decryption process for Firefox logins, including locating profiles, decrypting the master key, and decrypting login credentials from logins.json. The article then also discusses potential forensic implications and security event logs that are created after using the tool.

## 3 BACKGROUND

#### 3.1 Key management

A common system-wide software problem prevalent in nearly all systems is key management – how to securely store private keys and other confidential information. Most modern operating systems provide APIs which allow developers to encrypt or store credentials generated by their applications. In most implementations these credentials are encrypted with user's logon information, but a separate authentication option is possible as well. Due to their simplicity and general safety such APIs are used by many applications, such as Dropbox, Thunderbird, OpenVPN, Firefox, Chrome and most other browsers.

3.1.1 DPAPI. On Windows the API for encrypting and decrypting secrets is called Data Protection API (DPAPI) [12]. It was introduced in Windows 2000 and consists of functions for encrypting and decrypting memory and data. The two most important functions are:

CryptProtectData - encrypts provided data

CryptUnprotectData - decrypts provided encrypted blob

Developers can use these functions to encrypt private information and then store it on the disk. The API itself only provides functions for encryption and decryption, without secure storage.

To illustrate how a developer might use DPAPI, the signature of function CryptProtectData is shown in the listing 1. Parameters of CryptUnprotectData are similar and will therefore not be shown here.

#### Listing 1: Signature of function CryptProtectData

DPAPI\_IMP BOOL CryptProtectData ( DATA\_BLOB \*pDataIn, LPCWSTR szDataDescr, DATA\_BLOB \*pOptionalEntropy, PVOID pvReserved, CRYPTPROTECT\_PROMPTSTRUCT \*pPromptStruct, DWORD dwFlags, DATA\_BLOB \*pDataOut

);

The parameters are

- pDataIn- a pointer to the plaintext data we want to encrypt
- szDataDescr optional description of the data
- pOptionalEntropy optional additional entropy
- pvReserved reserved for future use
- pPromptStruct not needed for encryption and decryption

• dwFlags - flags

• pDataOut- pointer to the location for the encrypted blob Microsoft intended for the mechanism to only allow decryption on the same computer that encrypted the data<sup>1</sup>, therefore the encryption key is not required when using these functions. Microsoft does not disclose the internal structure and workings of DPAPI, which is problematic for many reasons, such as security validation, alternative implementations<sup>2</sup> and of course digital forensics.

One of the most successful reverse engineering attempts is described in the 2010 paper Recovering Windows Secrets and EFS Certificates Offline[8], which describes how the algorithm works and more importantly also provides a program for decrypting DPAPIencrypted data offline called DPAPIck. Since then many alternative and updated implementations of the decryption algorithm have emerged [7, 13]. The only information needed to decrypt DPAPIencrypted data is

- user's logon password or password hash, which can be obtained with tools such as mimikatz,
- master key file, which is stored on the disk: %APPDATA%\Microsoft\Protect\<SID>, where SID is user's security identifier<sup>3</sup>.

## 3.2 Linux keyrings

Linux does not have a direct alternative to DPAPI. The closest and most often used alternative are keyrings - user space programs that implement the secret service API. While DPAPI encrypt the credentials and lets the developer handle storing the encrypted data, keyrings act as key-value databases for storing confidential data. These programs are usually specific to the desktop environment used and only store credentials per-user instead of system-wide like DPAPI. Most commonly used implementations are GnomeKeyring for Gnome-based desktop environments and KWallet for KDEbased desktop environments. These keyrings are most often integrated with user login, which allows applications to use them in the background without prompting the user for a password. In this manner then are just as transparent to the end user as is DPAPI [2]. Some password managers, such as KeePassXC also implement the secret service API and can act as keyrings. Such password managers almost always require another authentication step which protect the stored data even if the user is logged in.

#### 3.3 Browser data storage

Browsers act as gateways to an enormous amount of private and confidential information. Everything from session tokens, cookies, bookmarks, browsing history and autofill passwords to cached and local storage for websites needs to be stored on disk in order to persist between browsing sessions. Such information is almost always stored in SQLite databases somewhere in the browser profile directory, but JSON and similar files are also common. In this article we concentrate on the two most popular browser implementations: Firefox and Chromium-based browsers. The vast majority

 $<sup>^1 \</sup>rm When$  using roaming profiles data can be decrypted for other computers on the network as well.

<sup>&</sup>lt;sup>2</sup>This is especially important for project such as Wine, which attempt to run Windows applications on Linux systems.

<sup>&</sup>lt;sup>3</sup>Protected operating system files should be unhidden in order to see the contents of this directory.

Cross-platform cloud data access through browser credential migration

of Chromium-based browsers have the exact same cookie storage [11] and a very similar internal file structure, so we will use the terms "Chrome" and "Chromium-based" interchangeably.

#### 3.3.1 Firefox. Firefox stores the profiles at

%appdata%\Mozilla\Firefox\Profiles\<profile> on Windows
and ~/.mozilla/firefox/<profile>/ on Linux [1]. The value
<profile> is a randomly generated string which acts as an ID for
each profile. The most interesting files are

- cookies.sqlite cookies,
- places.sqlite browsing history,
- formhistory.sqlite search bar and form history,
- logins.json usernames and password for the integrated password manager,
- key4.db master key for decrypting data in logins.json.

It is important to note that Firefox stores all data except the master key, usernames and passwords unencrypted. Example of such database is shown in figure 1, where the value column contains plaintext cookie values, such as session tokens.

IQ .	1 riginAttribute	e name	value	host	path	expiry	lastAccessed	creationTime
Filte	r Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
	0	donation-identifier	MC4wNzk3OTgwMDEyNTQz0Dc5Ng==	.openlibrary.org	/	1738872969	1707336969929882	1707336969929882
	1	OH.FLID	93ad49b0-b16f-4e7e-899c-5b130d4574e5	www.office.com	1	1747493388	1715957387934000	171595738793400
	2	MUID	2FC78AB29061644B28A29E3091EA652A	.office.com	1	1749653388	1715957387934000	171595738793400
	3	MicrosoftApplicationsTele	d8071aa0-5c6a-4c6c-bbc1-eee8c727563d	www.office.com	1	1747493388	1715957388162000	171595738816200
	4	ai_session	AK3x2MUTwYeNarPS3veWqv 1715957388165	www.office.com	1	1715959188	1715957388294000	171595738816400
1	3 ^partition	MC1	GUID=2396a130b2ef41e2822704892751f4428HA	.microsoft.com	1	1747493389	1715957389702000	171595738943900
1	4 ^partition	MS0	e1cfb2c254ea423abc5358c079e47461	.microsoft.com	1	1715959189	1715957389702000	171595738943900
1	.5	_octo	GH1.1.1113215169.1715957389	.github.com	1	1747493389	1715957390002000	171595739000200
1	.6	logged_in	no	.github.com	1	1747493389	1715957390002000	171595739000200
1	.9	_Host-GAPS	1:rduH4kG3EQsNNZfOdWfQXnzciNu	accounts.google.com	1	1779029391	1715957391557000	171595739155700
2	1	edgebucket	kSN4muagn3VtOCwExT	.reddit.com	1	1779029391	1715957392282000	171595739228200
2	12	loid	0000000010k35vaji9.2.1715957392864.20FBQUF	.reddit.com	1	1779029392	1715957392755000	171595739275500
4	13	token_v2	eyJhbGciOUSUzI1NiIsImtpZCI6IINIQTI1NjpzS3ds	.reddit.com	1	1716043792	1715957392756000	171595739275600
2	14	csv	2	.reddit.com	1	1779029392	1715957392756000	171595739275600
- 2	16	WMF-Last-Access	17-May-2024	www.wikipedia.org	1	1718712000	1715957394438000	171595739443800
2	27	WMF-Last-Access-Global	17-May-2024	.wikipedia.org	1	1718712000	1715957394438000	171595739443800
2	18	NetworkProbeLimit	0.001	www.wikipedia.org	1	1715960994	1715957394439000	171595739443900
2	19	session-id	140-5742781-7558436	.amazon.com	1	1747493395	1715957395820000	171595739582000
3	10	session-id-time	2082787201	.amazon.com	1	1747493395	1715957395820000	171595739582000
2	81	i18n-prefs	USD	.amazon.com	1	1747493395	1715957395820000	171595739582000
	12	sp-odn	"L5Z9:SI"	.amazon.com	1	1747493395	1715957395820000	171595739582000
- 3	4 ^partition	_GRECAPTCHA	09AFmZ3eBhA7V3UBWZ4DeJjagXgK3YvxxZdvG_d	www.google.com	/recaptcha	1731509396	1715957396350000	171595739635000
-	16	ubid-main	131-7371918-8717743	.amazon.com	1	1747493396	1715957396567000	171595739656700
3	17	session-token	ASHGFEih5UFtodEpCtGBYjhxnwlYNb+D9whoRLCv	.amazon.com	1	1747493397	1715957397011000	171595739701100
2	18	guest_id	v1%3A171595739750105037	.twitter.com	1	1750171797	1715957397382000	171595739738200
3	19	guest_id	v1%3A171595739750105037	.x.com	1	1747493397	1715957397907000	171595739760000
4	11	gt	1791481289469489198	.x.com	1	1715966398	1715957398166000	171595739816600
4	12	night_mode	2	.x.com	1	1747493398	1715957398776000	171595739790700
4	16	SOCS	CAISEwgDEgk2MzQwODkxhTQaAmVuIAEaBgiAvJ	.youtube.com	1	1750085401	1715957401162000	171595740105600
4	17	VISITOR_INFO1_LIVE	4fghWJQaMUs	.youtube.com	1	1731509401	1715957401162000	171595740112000
	18	GPS	1	.youtube.com	1	1715959201	1715957401258000	171595740125800
4	19	VISITOR_PRIVACY_META	CgJT SRIEgQSAgsMIGk%3D	.youtube.com	1	1731509401	1715957401258000	171595739109600
5	60	PREF	tz=Europe.Ljubljana	.youtube.com	1	1779029402	1715957402009000	171595739193500
	51	eu_cookie	{%22opted%22:true%2C%22nonessential%22:t	www.reddit.com	1	1779029403	1715957403614000	171595740361400
-	14	datr	kW5HZIR018Gp2Tn0v8O8AW1K	.facebook.com	1	1750517405	1715957405769000	171595740576900
	15	sb	kW5HZiPUbU28jqFsRF-rXfr	.facebook.com	1	1750517405	1715957405769000	171595740576900
	16	csm-hit	tb:ZAXPZY862Y8N82MP3852+s	www.amazon.com	1	1746197407	1715957407779000	171595739608800
5	57	d_prefs	MT oxLGNvbnNlbnRfdmVyc2lvbjoyLHRieHRfdmVyc	.x.com	1	1731509412	1715957412608000	171595741260800
5	18	guest_id_ads	v1%3A171595739750105037	.x.com	1	1750171812	1715957412753000	171595741275300
	i9	guest_id_marketing	v1%3A171595739750105037	.x.com	1	1750171812	1715957412754000	171595741275400
-	in .	nersonalization id	"v1 +X0dPnEti ZwkgnMoPNN490=="	x.com	1	1750171812	1715957412754000	171595741275400

Figure 1: Contents of the cookies.sqlite database on Firefox.

*3.3.2 Chromium-based.* Chrome store the data in the profile directories which are located at

%localappdata%\Google\Chrome\User Data\<profile> on Windows and ~/.config/google-chrome on Linux. Most other Chromiumbased browsers also follow the <ventor>/<name> naming scheme for the path. <profile> follows the pattern Default, Profile 1, Profile 2 and so on. The most interesting files are

- Linux: Cookies, Windows: Network\Cookies cookies
- History browsing history
- Web Data autofill data (addresses, credit cards, ...)

• Login Data – usernames and password for the integrated password manager

Most of the data contained in the databases is encrypted (such as cookies), but some of it isn't, for example the browsing history. To encrypt this data the browser generates a random key which is used as the encryption key. On Windows systems the encryption key for these databases is encrypted with DPAPI, base64 encoded and stored in the JSON file

%LOCALAPPDATA%\Google\Chrome\User Data\Local State. The path of the encryption key in this file is os\_crypt.encrypted\_key. On Linux systems the key is stored in the preferred keyring, such as Gnome Keyring or KWallet. If the OS doesn't provide a keychain implementation the browsers defaults to plaintext storage, similar to how Firefox stores the data.

# 4 METHODOLOGY

The process for migrating the cookies file from one computer to another depends on the browser and the operating system used. This part of the article describes the process for Firefox and Chromiumbased browsers, but since the vast majority of other browsers are closely related to either Firefox or Chromium [6] the process should work there with no or with very minor adjustments.

#### 4.1 Firefox-based browsers

Since Firefox does not encrypt the cookies database, moving it is trivial. A simple solution is to manually modify the database using freely available tools, such as DB Browser for SQLite and add the required cookies. While editing the database file is possible, it can easily lead to mistakes, because modern websites often load cookies from different domains, and their purpose is often not obvious from the name. That is why we recommend copying the entire database file to a separate browser profile on the investigator's computer and thus ensure all required cookies are present.

Firefox uses exactly the same database structure both on Windows and Linux, which means that copying the database between operating systems is possible as long as the browser versions are the same or similar. For best success we advise against copying between operating systems, because some websites seem to perform a check and invalidate the session if the browser is different.

#### 4.2 Chromium-based browsers

Chromium-based browsers encrypt cookie values in the cookies database, which means that copying the entire file does not work. The easiest way to move cookies to another computer is by decrypting the database, copying it to the investigator's computer and encrypting it again. Although many similar scripts and programs already exist, we decided to write our own script to demonstrate the process of decrypting and encrypting the database file. The script is available at https://github.com/VidVidex/chrome-cookie-crypt.

4.2.1 Decryption. The first step is to decrypt the cookies using the browser's encryption key. On Linux this key is stored in the keyring and on Windows it is encrypted with DPAPI and stored in the Local State JSON file. If the user logged in we can simply ask DPAPI to decrypt the key or query the keyring for the key, otherwise we have to find a way to circumvent the encryption. On Windows this can be done by manually decrypting DPAPIencrypted data using the user's password, security identifier (SID) and master key and on Linux if the keyring is unlocked by the desktop session we can simply reset the user's password and relogin.

Our script assumes we're decrypting on the same computer as was used to encrypt the data. To decrypt the database the following command can be used:

python chrome.py decrypt in\_db out\_db

where in\_db is the path to the encrypted Cookies database and out\_db is the path for the copy of the database with decrypted cookies. The script supports decryption both on Windows with DPAPI and Linux with KWallet and will automatically choose the correct mechanism. The decrypted data will be stored in the same location as the previously encrypted data together with all information needed to re-encrypt, as can be seen in figures 2 and 3. The decrypted data is stored as a JSON object.

6	🔓 New Database 🗟 Open Database, 🗟 Write Changes 🕼 Revert Changes 🛛 🎯 Open Project 🎕 Save Project 🖉 Save Project								
Dat	Database Structure Browse Data Edit Pragmas Execute SQL				e SQL	Edit Database Cell • ×			
Tabl	Table: Cookles 🗸 🚳 😘 😽 🖓 > Filte					Mode: Binary 🗸 🥥			
		value	encrypted_value	path	expires	0000 76 31 30 47 5d 35 00 84 49 0b 94 98 33 b8 3c 54 v106]5I3. <t< th=""></t<>			
		Filter	Filter	Filter	Filter	0010 af 24 eb ce 43 c9 5a 0b e6 97 33 42 fd 49 19 9b .\$C.Z3B.I			
1			BLOB	/	133593235	0030 f6 cf d4 ba 04 ce 07 24 a8 a8 fe d8 e0\$			
2			BLOB	/	133619137				
3			BLOB	1	133934496	Type of data currently in cell: Binary Apply			
4			BLOB	/	133934497	61 byte(s)			
5			BLOB	1	133748737	DB Schema O X			
				,	100740707	Name Type Schema			

Figure 2: Encrypted cookie values in the SQLite database

	🔒 New Da	itabase	🗟 Open Database 🗍 🗟 Write Changes	s 🕼 Reve	rt Chan	ges 🛛 🎯 Open Project 🖓 Save Project 🖓 Attach Database	>
ļ	Database Stri	ucture	Browse Data Edit Pragmas Execute SQL		Edit Da	itabase Cell	♦ ×
	Table: 🔲 coo	kies	✓ Ø % % Ø @ @ Ø,>	Filte	Mode:	JSON 🗸 📦	a 🖷 🖨
		value	encrypted_value		1	{"algorithm": "AES-GCM", "prefix": [118, 49, 48], "nonce": [71,	93, 53, 7
		Filter	Filter	Filter		0, 132, 73, 11, 148, 152, 51, 184, 60], "tag": [74, 215, 284, 24	6, 207, 2
	1		("algorithm": "AES-GCM", "prefix": [118,			52, 50, 52, 48, 50, 95, 52, 56, 95, 53, 50, 95, 49, 50, 51, 57,	48, 48, 2
	2		{"algorithm": "AES-GCM", "prefix": [118,	/ 0		95, 52, 56, 95, 52, 51, 54, 51, 56, 48]}	
	3		("algorithm": "AES-GCM", "prefix": [118,	1	Туре о	f data currently in cell: Valid JSON	Apply
	4		("algorithm": "AES-GCM", "prefix": [118,	1	329 ch	aracter(s)	
					DD Cab		0 X

Figure 3: Decrypted cookie values in the SQLite database.

4.2.2 *Copying the database.* The decrypted database can be moved to the investigator's computer the same as any other file. The decrypted cookie values can be used as part of an investigation, but more commonly it will be used to access the websites using the stored cookies.

4.2.3 *Encryption.* In order for the investigator's browser to correctly parse the cookies database, it needs to be re-encrypted. The encryption is done using the browser's encryption key, which is again stored in the keyring or encrypted with DPAPI. If using our script, the cookies in the previously decrypted database can be re-encrypted with the following command:

python chrome.py encrypt in\_db out\_db

where in\_db is the path to the database file with decrypted cookie values and out\_db is the path to the copy of the database with re-encrypted cookies. This database can then be copied to the browser profile directory.

Interestingly, Chrome does not use the same encryption mechanism on Windows and Linux. On Windows it uses AES-GCM and on Linux it uses AES-CBC. Because of this moving the cookies database between operating systems is not supported when using our script.

4.2.4 Bonus: Accessing the built-in password manager. As mentioned in the Firefox methodology section, some websites recognize cookie-based session hijacking. In order to minimize the possibility of this, usage of the same browser version and the same operating system is recommended. It is also possible for websites to invalidate the session if the IP address we're accessing from is from a different location.

If session hijacking fails, we can try accessing the built-in password manager. Chromium-based browsers store this information in the Login Data database and use exactly the same encryption mechanism as the cookies database. We added this bonus feature to the script and can be used with the following command:

python chrome.py passwords login\_data\_db

where login\_data\_db is the path to the Login Data database. The output of this command is a list of all stored usernames and passwords in the password manager, similar to the output shown below.

Found 2 passwords:

URL: https://office.com User: jn1234 Password: password123

URL: https://ucilnica.fri.uni-lj.si User: jn1234 Password: hunter2

#### 5 DISCUSSION

The ability to migrate browser data securely and effectively across different systems is crucial for forensic investigators. Our methodology provides a clear framework for handling such migrations, ensuring that critical evidence can be accessed and analysed irrespective of the original system configuration. This is particularly relevant given the diversity of browsers and operating systems in use today.

Our article shows that while Firefox simplifies the migration process by storing most data in plaintext, Chromium-based browsers add complexity with encrypted storage. This necessitates the use of techniques for decryption and re-encryption, which we have addressed through our custom script. The practical utility of this script demonstrates how forensic practitioners can streamline their workflows, reducing the time and effort required to access necessary data.

Despite the advancements outlined in this article, several challenges remain. One major issue is the reliance on system-specific encryption mechanisms, such as DPAPI on Windows and keyrings on Linux. These mechanisms, while enhancing security, complicate the migration process. Our script mitigates some of these challenges, but it is not a universal solution. Cross-platform cloud data access through browser credential migration

Another limitation is the potential for session hijacking detection by websites. As mentioned, some websites may invalidate sessions if they detect discrepancies in browser or operating system details and even IP-based location. This necessitates the use of identical browser and operating systems versions during forensic analysis. It also forces us to use an IP address which resolves to a similar location, which may not always be feasible.

Future research could focus on extending support for other operating systems and past versions of the encryption mechanism used. Additionally, further exploration into more sophisticated antidetection techniques for session hijacking could improve the reliability of credential migration.

Another area of interest is the investigation of emerging encryption mechanisms and browser security features. As browsers continue to evolve, staying ahead of these changes will be essential for maintaining the effectiveness of forensic methodologies. Collaborative efforts between forensic practitioners and browser developers could lead to standardized methods for secure data migration that preserve both security and accessibility.

# 6 CONCLUSION

To conclude, this article first describes the operating system mechanism for encrypting or storing private information and how browsers use (or don't use) these mechanisms to store confidential information on disk. The methodology section lays out a step-by-step guide for transferring cookies databases between computers for both Firefox and Chromium-based browsers.

While Firefox's straightforward approach of storing certain data without encryption simplifies migration, Chromium's encryption necessitates decryption and re-encryption procedures. The inclusion of scripts for encrypting and decrypting the cookie database on Chromium-based browsers adds a practical aspect to the article and offers immediate results for digital investigations.

#### REFERENCES

- [1] [n. d.]. Firefox Data Stores. https://github.com/mozilla/firefox-data-store-docs/ blob/master/README.md. Accessed on 2024-05-05.
- [2] [n. d.]. GNOME/Keyring. https://wiki.archlinux.org/title/GNOME/Keyring. Accessed on 2024-05-05.
- [3] Apra. 2019. Decrypting Browser Credentials For Fun (But Not Profit). https: //apr4h.github.io/2019-12-20-Harvesting-Browser-Credentials/.
- [4] Apra. 2019. HarvestBrowserPasswords. https://github.com/Apr4h/ HarvestBrowserPasswords.
- [5] Hyunji Chung, Jungheum Park, Sangjin Lee, and Cheulhoon Kang. 2012. Digital forensic investigation of cloud storage services. *Digital Investigation* 9, 2 (2012), 81–95. https://doi.org/10.1016/j.diin.2012.05.015
- [6] Sourojit Das. [n. d.]. Understanding Browser Market Share: Which browsers to test on in 2023. https://www.browserstack.com/guide/understanding-browsermarket-share. Accessed on 2024-05-05.
- [7] Tijl Deneut. [n. d.]. Windows DPAPI-NG lab in Python 3. https://github.com/ tijldeneut/dpapilab-ng. Accessed on 2024-05-05.
- [8] Bursztein Elie and Picod Jean-Michel. 2010. Recovering windows secrets and efs certificates offline. In Workshop On Offensive Technologies. Usenix.
- [9] Nathan Henrie. 2014. Decrypting Chromium cookies. https://stackoverflow.com/ questions/23153159/decrypting-chromium-cookies.
- [10] Nathan Henrie. 2014. pycookiecheat. https://github.com/n8henrie/ pycookiecheat.
- [11] Uk Hur, Soojin Kang, Giyoon Kim, and Jongsung Kim. 2023. A study on cloud data access through browser credential migration in Windows environment. Forensic Science International: Digital Investigation 45 (2023), 301568. https: //doi.org/10.1016/j.fsidi.2023.301568
- [12] Microsoft. [n.d.]. DPAPI. https://learn.microsoft.com/en-us/windows/win32/ api/dpapi/. Accessed on 2024-05-05.

[13] ParrotSec. [n. d.]. mimikatz. https://github.com/ParrotSec/mimikatz. Accessed on 2024-05-05.



Izzivi Študij

Ustvarjanje novih svetov

# 7 - Forenzična preiskava / Forensic Investigation

# Smernice za forenzično preiskavo avtomobilskih kamer

Gašper Pistotnik University of Ljubljana, Faculty of Computer and Information Science Ljubljana, Slovenia gp0478@student.uni-lj.si Žiga Bobnar University of Ljubljana, Faculty of Computer and Information Science Ljubljana, Slovenia zb48750@student.uni-lj.si Timotej Zgonik University of Ljubljana, Faculty of Computer and Information Science Ljubljana, Slovenia tz36793@student.uni-lj.si

# POVZETEK

V sodobnem prometnem okolju se vse pogosteje srečujemo z različnimi vrstami kamer, ki so postale ključni del nadzornih sistemov na cestah. Te kamere niso samo pomemben element za vzdrževanje reda in zagotavljanje varnosti na cestah, temveč postajajo tudi nepogrešljiv del forenzičnih preiskav. Namen te seminarske naloge je osvetliti različne vidike uporabe avtomobilskih kamer, zlasti tistih, ki so nameščene v vozilih (t.i. dashcams), in raziskati, kako lahko te naprave služijo kot ključni viri dokazov v pravnih in zavarovalniških primerih.

# KLJUČNE BESEDE

Smernice za preiskavo, forenzična preiskava avtokamer, prometne nesreče, dashcam

# 1 UVOD

Avtomobilske kamere, znane kot dashcams, postajajo ključni del sodobnih vozil, predvsem zaradi njihove sposobnosti objektivnega dokumentiranja prometnih situacij in nesreč. S svojimi širokimi zmožnostmi snemanja, ki vključujejo video- in avdiozapise ter zbiranje podatkov o lokaciji in hitrosti, nudijo dragocene dokaze, ki se lahko uporabljajo v pravnih in zavarovalniških postopkih. Njihova uporaba prinaša tudi številne tehnične in pravne izzive, kot so zasebnost, potencial za manipulacijo s podatki in pomanjkanje standardiziranih forenzičnih pristopov.

V tej seminarski nalogi bomo raziskali, kako se avtomobilske kamere uporabljajo v forenzičnih preiskavah, kakšne podatke nam lahko zagotovijo, in s kakšnimi izzivi se srečujejo forenzični strokovnjaki pri njihovi uporabi. Prav tako bomo obravnavali potrebo po razvoju standardiziranih metod za obdelavo in analizo podatkov z avtokamer, da bi zagotovili njihovo verodostojnost in uporabnost kot dokaznega materiala v sodnih postopkih.

#### 1.1 Kamere v prometu in forenzika

V današnjem svetu so naprave za snemanje videa prisotne povsod okoli nas. Širi se tudi njihova uporaba v prometu. Tam so lahko prisotne kot nadzorne kamere, ki jih postavi in vzdržuje upravljalec ceste, lahko jih začasno postavijo policisti za nadzor prometa ali jih imajo na sebi v obliki telesnih kamer (t.i. bodycam). Poznamo tudi kamere za prepoznavo registrskih tablic (na primer za nadzor plačevanja cestnin), lahko so v obliki naprav za meritev hitrosti (radar) ali stacionarne kamere za nadzor prometa (na primer na semaforjih za spremljanje prevoženih rdečih luči). Lahko pa kamere vozniki sami dodatno namestijo v svoje vozilo, če to ni že tovarniško vgrajeno.

Medtem ko ima vsaka kamera svoje značilnosti, so mnoge rešitve izvedene na način, da v primeru potrebe po uporabi posnetkov kot dokaznega gradiva omogočijo pooblaščenim osebam dostop. Hkrati pa imajo preiskovalci že določene standardne smernice, ki se jih držijo pri postopku pridobivanja dokaznega gradiva.

Pod drobnogledom si oglejmo še avtomobilske kamere (angl. dashcam, v nadaljevanju avtokamere). Njihova uporaba je v zadnjem času v zelo velikem porastu zaradi vedno ugodnejših cen in vedno večjih zmogljivosti. Obenem pa je na trgu vedno več modelov različnih proizvajalcev takšnih kamer, med seboj pa se vsaka lahko razlikuje po vgrajenih funkcionalnostih, specifikacijah, uporabniškem vmesniku kot tudi načinu shranjevanja podatkov.

Avtokamere se najpogosteje uporabljajo za:

- Razjasnitev okoliščin prometnih nesreč, če si pričevanja očividcev nasprotujejo.
- Objektivni prikaz okoliščin in dogajanja v času nesreče in pred njo.
- Zavarovalniške zahtevke za dokazovanje vzrokov in krivcev nesreče.
- Dokazovanje dejstev na sodišču, kjer služi podatek kot alibi oziroma v nasprotnem primeru kot dokaz, da je bila oseba prisotna na določenem kraju ob določenem času.

Avtokamere imajo tudi nekaj značilnosti, ki jih naredijo precej pomembne v primeru potrebe po dokaznem gradivu. Vsebujejo namreč videoposnetek dogajanja, pogosto pa tudi pripadajoč zvočni posnetek, kar je že samo po sebi zelo uporabno. Poleg tega vsebujejo tudi podatke o času nastanka datotek, s čimer se lahko dokaj natančno povezuje dogajanje na posnetkih še po časovni osi. Marsikatere kamere vsebujejo GPS module, ki omogočajo tudi beleženje lokacije, orientacije in hitrosti, nekatere pa vsebujejo tudi senzorje sil, ki lahko samodejno aktivirajo snemanje in preprečijo prepisovanje podatkov, ko zmanjkuje pomnilnika za shranjevanje. Vse to preiskovalcu ponuja ogromno podatkov, ki jih lahko z ustrezno analizo spremeni v uporabno dokazno gradivo za uporabo na sodišču [Giovannini et al. 2021].

Ker so si lahko avtokamere lahko zelo različne, so nujno potrebni ustrezni forenzični pristopi in orodja, saj se lahko v primeru napačnega postopka pojavijo spremembe dokaznega gradiva oziroma celo neželjen prepis ali uničenje podatkov, ki jih želi preiskovalec analizirati. Do težave pride, ker splošno sprejetih smernic trenutno še ni. [Lallie 2023] je v svojem članku ter konferenci raziskal to področje in je glavni vir, po katerem se bomo zgledovali v tej seminarski nalogi.

# 2 FORENZIČNA PREISKAVA AVTOKAMER

Pri forenzični preiskavi avtokamer se soočamo z več izzivi, med katerimi je največji trenutno verjetno vprašanje avtentičnosti in manipulacije s posnetki. Različni izzivi, ki se pojavljajo na tem področju, so:

- Manipulacija s posnetki (angl. tampering) video posnetke, zajete z avtokamerami, je mogoče relativno enostavno digitalno obdelati ali manipulirati, da bi se prikrila ali spremenila dejstva, ki bi bila sicer prisotna na posnetkih. Takšne manipulacije lahko vključujejo brisanje, urejanje ali dodajanje vsebine in otežujejo preverjanje pristnosti posnetkov. Poleg manipulacije s posnetki lahko pride tudi do manipulacij z metapodatki in drugimi podatki, ki jih avtokamera beleži (kot so na primer GPS podatki). V takšnih primerih mora biti forenzični preiskovalec pozoren, če obstaja sum na kakršno koli manipulacijo.
- Standardizacija in kompatibilnost različni proizvajalci avtokamer uporabljajo različne formate zapisa posnetkov in podatkov, kar lahko oteži izvedbo standardnih forenzičnih postopkov. Forenzični preiskovalci se morajo soočiti s specifično programsko opremo proizvajalca, ki ni zasnovana za uporabo v forenzične namene. Morda je potrebno tudi pretvoriti podatke v ustrezen format, ki je združljiv z orodji za analizo. Za marsikatero napravo obstaja tudi verjetnost, da orodje še ne obstaja in ga je potrebno najprej razviti in validirati njegovo delovanje.
- Zasebnost in zakonske omejitve avtokamere pogosto snemajo osebne informacije, ki bi lahko omogočile prepoznavo oseb na posnetkih, registrske tablice in lokacije. Uporaba teh podatkov je lahko predmet pravic zasebnosti posameznikov, kar predstavlja morebitno kršitev pravic do varstva osebnih podatkov.
- Kakovost posnetkov sama kvaliteta videa iz avtokamer je nizka, še posebej v slabih svetlobnih razmerah ali ob hitrih premikih, kar otežuje identifikacijo oseb ali vozil v teh situacijah. Poleg tega se posnetek ne shranjuje v surovi obliki, ampak se pred shranjevanjem obdela in stisne, pri čemer pride do izgube kakovosti, kar bi lahko povzročilo neprepoznavnost ključnih podrobnosti.
- Zanesljivost podatkov podatki, kot so GPS koordinate, hitrost vozila in časovni žigi, ki jih zagotavljajo avtokamere, lahko imajo določeno mero netočnosti oziroma lahko postanejo popolnoma napačni zaradi tehničnih napak ali uporabniško namerno spremenjenih nastavitev. Poleg tega je vprašljiva tudi zanesljivost medija, na katerega se posnetki shranjujejo. Pogosto so namreč uporabljajo kartice SD, ki se lahko po veliko zapisanih podatkov izrabijo in podatkov ne shranjujejo več pravilno.
- Velike količine podatkov pogosto se forenzični preiskovalci soočajo z ogromnimi količinami videoposnetkov, ki jih je treba pregledati in analizirati. To lahko zahteva veliko časa, osebja in sredstev ter učinkovite metode za hitro prepoznavanje in nadaljnjo obdelavo relevantnih delov podatkov.
- Sprejemljivost v pravnem sistemu v nekaterih pravnih sistemih je lahko sprejemljivost dokazov iz avtokamer omejena, če ni jasne dokumentacije, kako so bili podatki zbrani, shranjeni in obdelani. Vsi postopki morajo biti izvedeni v skladu z zakonom in mora biti zagotovljeno, da posnetki niso bili kompromitirani. Ta problem se rešuje z ustrezno vodeno verigo dokazov (chain of custody).

Ti izzivi predstavljajo ključne ovire, ki lahko vplivajo na zanesljivost in uporabnost forenzičnih analiz avtokamer v pravosodnem kontekstu. Z ustreznimi postopki za forenzične preiskovalce se lahko zagotovi, da se podatki, pridobljeni iz avtokamer, uporabijo kot dokazno gradivo.

Poleg vseh težav in pomanjkanja standardnih praks za avtokamere, s časom število forenzičnih modelov narašča, kar povečuje obremenitev forenzikov glede znanja postopkov. Zaradi tega ni smotrno definirati novega modela samo za avtokamere, saj jih obstaja že veliko, ki bi z ustreznimi prilagoditvami v procesu lahko zadostili našemu problemu. Samim avtokameram najbolj ustreza model Guidelines on Mobile Device Forensics [Ayers et al. 2014] po standardu organizacije NIST. To je standardni model, ki združuje najbolj pogoste in obvezne postopke pri analizi mobilnih naprav, kar avtokamera tudi je.

#### 2.1 Model za forenzično preiskavo avtokamer

Model, predstavljen v viru, je torej osnovan na modelu za analizo mobilnih naprav. Pomembni deli vsakega modela so nujna ohranitev verige dokazov - vsak poseg po dokazih mora biti dokumentiran. Sami koraki modela pa so:

- Ohranjanje (angl. preservation) zagotoviti je treba, da se originalni podatki ne spremenijo. Sem spadata varovanje podatkov pred morebitnimi poškodbami in varovanje pred manipulacijo s podatki.
- (2) Pridobivanje (angl. acquisition) zajema proces kopiranja in zajema dokaznega gradiva na načine, ki bodo omogočili, da se originalni dokazi ne bodo spreminjali.
- (3) Pregled in analiza (angl. examination and analysis) v tem koraku se zbrano gradivo pregleda in ugotavlja, kateri podatki so pomembni za aktivno preiskavo. Nato se jih analizira s pomočjo forenzičnih orodij in samega razumevanja podatkov.
- (4) Poročanje (angl. reporting) priprava poročila, ki mora biti razumljivo osebam, ki niso nujno usposobljene glede podrobnosti delovanja naprav. V poročilu mora biti vsak korak preiskave podrobno dokumentiran, poleg tega morajo biti dokumentirane tudi vse ugotovitve, metodologije in zaključki. To poročilo je namenjeno naprejšnji uporabi v pravosodnem sistemu in mora vsebovati vse potrebne podatke, da je veljavno.

Vse skupaj se začne že na samem prizorišču dogodka, ko morajo prvi prisotni na prizorišču zavarovati območje in preprečiti poškodovanje oziroma manipulacijo morebitnega dokaznega gradiva. Ta del procesa bomo imenovali prvi odziv (angl. *first response*).

#### 2.2 Koraki prvega odziva

Prvi odziv se odvija na samem kraju dogodka (običajno nesreče oziroma zločina). Pomembno je, da je oseba, ki izvaja preiskavo, seznanjena z značilnostmi naprave, ki jo ima namen preiskovati, kar pomeni, da mora poznati koncepte uporabe naprave in ustrezno ukrepanje ob različnih dogodkih z napravo. Vse to je nujno, da lahko izvede ustrezne postopke za zavarovanje in pridobivanje dokaznega gradiva. Smernice za forenzično preiskavo avtomobilskih kamer



#### Slika 1: Generični model za preiskavo mobilnih naprav po standardu NIST. Vir: [Ayers et al. 2014]

Prvi korak, ki je skupen tudi vsem preiskavam vseh drugih objektov, je izvedba vseh tradicionalnih forenzičnih pristopov. To predstavljajo postopki pridobivanja prstnih odtisov in DNA analiza.

Nato lahko preiskovalec sledi naslednjemu postopku, ki je ustrezno prilagojen za rokovanje z avtokamerami:

- Ustavitev snemanja s tem se prepreči, da bi se podatki okvarili zaradi nenadne izklopitve napajanja ali napačnega načina prekinitve snemanja.
- (2) Ugasnitev naprave, odklop napajanja preprečimo, da bi se podatki shranjeni v pomnilniškem modulu kamere, nepričakovano prepisali oziroma drugače spremenili.
- (3) Odstranitev pomnilniških modulov, zaščita pred pisanjem v kamerah so najbolj pogosti pomnilniški moduli v obliki kartic SD in microSD. Potrebno jih je odstraniti iz naprave, dokumentirati, uvrstiti v verigo dokazov in ustrezno zaščititi pred pisanjem (v primeru SD kartice je to stikalo "lock"). S tem preprečimo, da bi se ob naslednjem vklopu kamere podatki na pomnilniškem modulu slučajno spremenili.
- (4) Vklop kamere, dokumentiranje nastavitev potrebno je ročno preko uporabniškega vmesnika dokumentirati stanje nastavitev in konfiguracije v kameri. To vključuje morebitno shranjeno registrsko številko, različico programske opreme, različico operacijskega sistema, proizvajalca, serijsko številko in podobno.
- (5) Ugotovitev časa in primerjava z referenčnim časom poiskati je potrebno, kje se nahaja podatek o trenutnem času

kamere, ga dokumentirati in hkrati primerjati z referenčnim trenutnim časom (ponavadi je to UTC čas). To omogoča, da se ugotovi odstopanje nastavljene ure naprave od dejanskega časa.

- (6) Izklop avtomatskega vklopa nekatere kamere se ob priključitvi zunanjega napajanja (pogosto vklop kontakta vozila) samodejno vklopijo in pričnejo s snemanjem. To je potrebno izključiti, da kamera sama ne more začeti snemati.
- (7) Izklop senzorjev pospeška poleg avtomatskega vklopa imajo lahko določene kamere senzorje pospeška, ki prav tako ob zaznanem pospešku vklopijo kamero in začnejo snemanje. Tudi to je potrebno izključiti.
- (8) Izolacija povezljivosti nekatere kamere imajo vgrajene povezave Wi-Fi, Bluetooth, mobilne podatke, GPS in podobno. Da se prepreči kakršna koli komunikacija z zunanjim svetom in s tem manipulacija z napravo oziroma odkrivanje njene lokacije, je potrebno vse povezave izključiti. Poleg tega je priporočeno napravo vstaviti v ustrezno zaščitno embalažo, ki preprečuje pošiljanje signalov, za kar je največkrat uporabljena tako imenovana Faradayeva kletka oziroma podobni pristopi, ki elektromagnetnemu valovanju do določene frekvence preprečujejo prehod skozi embalažo.
- (9) Popis/dokumentacija oznak na kameri potrebno je narediti fotografije same naprave, vseh perifernih naprav in kablov, vseh označb na kameri, kot so serijska številka, proizvajalec in model.

Poleg teh korakov so v nekaterih situacijah potrebni še nekateri drugi, kot je na primer identifikacija vse dodatne opreme, ki jo naprava potrebuje za delovanje. To so lahko GPS antene, napajalni kabli, kabli za povezavo z računalnikom in dodatni pomnilniški moduli. V primeru, da je kamera zaščitena s kodo ali geslom, je to potrebno identificirati čimprej in zagotoviti dostop do shranjenih podatkov. Zadnji korak je predvsem pomemben pri uporabi aplikacij na mobilnih telefonih kot avtokamere, saj so ti pogosto zavarovani s kodo, medtem ko same kamere na trgu še nimajo splošne podpore za zaščito.

Vsak od korakov mora biti seveda tudi ustrezno dokumentiran za zagotavljanje verige dokazov in podprt s fotografijami.

# 3 TRIAŽA

V današnjem svetu tehnologije postajajo pomnilniški mediji vedno večji po kapacitetah. Trdi diski lahko vsebujejo več terabajtov (TB) podatkov, dodatno jih je lahko v vsakem sistemu prisotnih tudi več. To lahko za preiskovalce predstavlja težavo z nabiranjem inventarja stvari, ki potrebujejo obdelavo (angl. backlog), in ogromno količino podatkov, ki jih je potrebno analizirati. Zato je potrebno pred samo analizo najprej oceniti, kakšno vrednost imajo za dotično analizo in kako prioritetno jih je potrebno obdelati. Proces odločanja prioritet se imenuje triaža in omogoča, da lahko preiskovalci agilno analizirajo dokazno gradivo in čim bolje ter čim hitreje iz njega pridobijo uporabne zaključke.

Za primer samih avtokamer sama količina podatkov ne predstavlja nujno največje težave, saj so pomnilniški mediji v velikostnem rangu nekaj 32 gigabajtov (GB) do okrog 128 GB, kar je v primerjavi s preostalimi napravami relativno malo. Večji problem je časovna potratnost, saj je potrebno vse posnetke ustrezno organizirati, poiskati bistvene odseke in jih analizirati.

Triaža poteka tako, da se že na samem kraju preiskave ugotavlja, kateri dokazi so verjetno bolj pomembni od drugih in za katere potrebujemo čimprejšnjo analizo (na primer zaradi časovne nestabilnosti samih dokazov) in čimprejšnje ugotovitve. S tem se bistveno zmanjšajo količine naprav in podatkov, ki potrebujejo takojšnjo analizo. S tem se dodatno tudi sprosti same preiskovalce, saj nimajo tolikšnega pritiska z odločanjem, kaj je bolj pomembno.

Pri triaži se upošteva ne samo dokazno gradivo trenutne preiskave, ampak je potrebno upoštevati tudi nujnost same preiskave kot take. V forenzičnem laboratoriju lahko namreč potekajo že druge preiskave in je nujno določiti ustrezni vrstni red.

Na pomembnost preiskave vplivajo:

- nevarnost oseb za družbo
- potreba po hitrem odločanju
- identifikacija najpomembnejših dokazov
- identifikacija ogroženosti oseb
- potreba po globoki analizi podatkov na primer kriptirane naprave

#### 3.1 Načini pridobivanja podatkov iz avtokamer

Pri pridobivanju podatkov iz avtokamer ločimo različne načine po zahtevnosti in po invazivnosti.

*Ročna ekstrakcija* predstavlja neposredno uporabo naprave in dostopanje do podatkov prek uporabniškega vmesnika. Pri avtokamerah to še vedno ostaja pomemben in potreben korak, saj je potrebno celotno konfiguracijo kamere dokumentirati, saj se notri lahko skrivajo pomembni podatki, na primer registrska številka, izklopljene oziroma vklopljene varnostne nastavitve kamere in ostali podatki, ki bi lahko nakazovali na možnost, da je bilo z napravo manipulirano na kakršenkoli način.

Drug način pa je tako imenovana *logična ekstrakcija*. Ta predstavlja izdelavo kopije podatkov, ki so prisotni na napravi z ustreznimi namenskimi orodji. Šele potem, ko že obstaja kopija podatkov, ki je ustrezno tudi varnostno kopirana in preverjena z zgoščevalnimi funkcijami lahko na takšnih kopijah izvajamo forenzične analize. Sem lahko štejemo izdelavo kopij pomnilniškega medija na bitnem nivoju oziroma izdelavo kopije po najboljših zmogljivosti na takšen način, da poskrbimo, da ne pride do poškodbe originalnih podatkov. Za primer avtokamer lahko ta proces glede na invazivnost v napravo ločimo na 3 sklope:

- Neposredno kopiranje pomnilniškega modula ker gre v večini primerov za kartice tipa SD, je možnost, da se jo varno odstrani iz naprave in uporabi standardni nabor forenzičnih orodij, ki omogočijo izdelavo kopije medija brez večje nevarnosti za podatke. Tukaj je eno izmed standardnih orodij FTK imager, obstajajo pa seveda tudi druga dobro preizkušena orodja.
- Preko kabla za povezavo z računalnikom mnogo avtokamer omogoča priključitev na računalnik preko USB vrat. V tem primeru se naprava najpogosteje predstavlja računalniku kot MTP naprava, ki omogoča dostop do datotek na datotečnem sistemu, ne omogoča pa neposrednega dostopa do samega datotečnega sistema. To pomeni, da ni nujno možno odkriti primerov skritih in izbrisanih datotek. Poleg

tega je postopek dokaj invaziven, saj mora biti naprava ves čas dostopa do podatkov prižgana in obstaja nevarnost, da bi se kljub previdnosti podatki prepisali. Dodatno nevarnost predstavljajo tudi poškodovani kabli in konektorji, saj se lahko v primeru slabe povezave namesto originalnih podatkov v kopijo prikradejo napake, ki jih ne moremo nujno odkriti. Zaradi tegase priklopu naprav na računalnik izogibamo, če obstaja druga možnost.

 Neposreden izvoz iz čipa - preko uporabe specializiranih orodij se lahko ustvari bitna kopija čipa. Ta orodja so običajno hexdump in uporaba JTAG vodila, lahko pa je potrebno čip previdno prestaviti iz naprave v napravo za branje. Ta postopek je zelo invaziven, saj zahteva odpiranje naprave, ob odstranjevanju čipa pa obstaja možnost, da se le-ta poškoduje. Poleg tega je za ustrezno branje potrebno imeti kompatibilno opremo in seznanitev z uporabo. Ta način je potreben, če je naprava poškodovana in medij za shranjevanje ni odstranljiv.

# 4 FORENZIČNA ORODJA ZA ANALIZO PODATKOV AVTOKAMER

Naslednji korak po zajemu podatkov in izdelavi kopij, na kateriih se lahko izvaja preiskava, je analiza pridobljenih podatkov. V digitalni forenziki obstaja množica splošno sprejetih orodij, ki imajo standardne procese za forenzično analizo že vgrajene in karseda avtomatizirane. Primeri teh orodij so Encase, FTK (Forensics Toolkit) in Autopsy. Težava pa je v tem, da trenutno še nimajo vgrajenih procesov za avtomatizirano analizo posnetkov in podatkov avtokamer.

Proizvajalci mnogih avtokamer ponujajo lastno programsko opremo za pregled in ogled posnetkov, narejenih s takšnimi kamerami, poleg tega pa lahko prikazujejo še druge metapodatke, shranjene v datotekah. Težava pa nastane, ker je takšna programska oprema zaprtokodne narave in ni nujno ustrezno validirana in preizkušena, da ne prihaja do napak, saj ni namenjena uporabi v forenzične namene. To pomeni, da je potrebno morebitne pridobljene ugotovitve primerjati tudi z rezultati, ki jih dobimo z uporabo drugih orodij.

V praksi se pogosto uporabljajo tudi skripte, specializirane za namensko uporabo in druga posebna orodja, ki jih lahko preiskovalec uporablja v ustreznem zaporedju, da pridobi ugotovitve. V tem primeru pa je težava, da morajo biti skripte prilagojene vsaki napravi posebej in je veliko ukazov potrebno napisati in izvesti ročno. Poleg tega mora biti vsak korak argumentiran in dokumentiran, preiskovalec pa mora biti dobro seznanjen z delovanjem tako naprave, ki jo analizira, kot tudi z vsakim od orodij.

Osredotočiti se je potrebno tudi, kje in na kakšen način priti do podatkov na avtokameri. Z ročnim načinom ekstrakcije s same naprave lahko pridobimo veliko podatkov o sami kameri in njenih nastavitvah, na nekaterih napravah pa tudi seznam narejenih posnetkov. Z uporabo drugega načina pa lahko iz pomnilnega medija naprave pridobimo same posnetke in dodatne datoteke. Pomnilni mediji so običajno v standardnih formatih FAT32 ali exFAT, ki so splošno poznani in imajo dobro podporo s strani obstoječih forenzičnih orodij. S poznavanjem datotečnih sistemov je mogoče uporabiti tehnike odkrivanja prikritih in izbrisanih datotek, prav
tako pa tudi delno prepisanih delov, ki bi lahko razkrili, če je bilo slučajno manipulirano z napravo.

Pomembno je biti tudi pozoren na podatke, ki so lahko očitni, ampak jih lahko zaradi te očitnosti spregledamo. Med takšnimi podatki so lahko že imena datotetk in map v datotečnem sistemu, ki lahko vsebujejo datum in uro nastanka, razlog za nastanek in druge podatke.

### 5 PODATKI NA KAMERI

Posnetki so običajno v formatih .mov (najbolj pogosto) ali .mp4 (Garmin), ki jih lahko analiziramo in obdelujemo z orodji za analizo videa, poleg videa pa so prisotne v posnetku tudi druge informacije. Te so običajno v exif metapodatkih videa in jih lahko pridobimo z uporabo exiftool. Iz posnetka torej lahko pridobimo ogromno podatkov, dodatno pa lahko iz same datoteke dobimo tudi čas nastanka in zadnje spremembe, mogoče tudi čas zadnjega dostopa (odvisno od datotečnega sistema in implementacije).

Poleg samih posnetkov lahko pogosto najdemo tudi druge, s posnetkom povezane datoteke, na primer .nmea datoteke (National Marine Electronics Association), ki vsebujejo podatke o lokaciji, času in hitrosti, pridobljene iz GPS sprejemnika. NMEA datoteke so tekstovne datoteke, ki vsebujejo veliko različnih podatkov, ker pa je to standardizirani format, obstajajo orodja/skripte/knjižnice, ki omogočajo navigacijo po podatkih in pretvorbo ter izvoz željenih podatkov v ustreznejše formate, na primer KML. Ker gre za format, ki je prvotno namenjen za mornarstvo, ima določene značilnosti, na katere je treba biti pozoren, na primer hitrost, ki je merjena v vozlih na sekundo, poleg samih koordinat pa se lahko shranjuje tudi smer, ki se pridobi iz magnetometra - kompasa.



Slika 2: Primer pridobljene poti iz posnetka avtokamere. Vir: [Daraghmi and Shawahna 2023]

Če kamera ne shranjuje podatkov o lokaciji v ločeno datoteko oziroma v metapodatke posnetka, se običajno shranjuje v sam videoposnetek, in sicer v obliki *vodnega žiga* (angl. watermark). Ta je večinoma prisoten na spodnjem robu posnetka in vsebuje trenutne koordinate, hitrost in čas, lahko pa tudi informacije o kameri in registrsko številko, nastavljeno v konfiguraciji kamere. V primeru, da želimo pridobiti podatke iz vodnega žiga, je potrebno za vsako spremembo v videu spremembe iz slike ročno prepisovati. To je zamudno delo in lahko privede do veliko napak. V zadnjem času se pojavlja vedno več rešitev, ki lahko s pomočjo strojnega učenja iz slik prepoznavajo besedilo. Pričakovati je, da se bodo ta orodja ustrezno razvila in s tem poenostavila proces pridobivanja podatkov iz samega videoposnetka [Daraghmi and Shawahna 2023].



#### Slika 3: Vodni žig, prisoten na desnem spodnjem robu posnetkov avtokamer. Vir: [Lallie 2020]

Kot dodatno funkcionalnost pa ima veliko avtokamer vgrajen tudi mikrofon, ki lahko s snemanjem zvoka v kabini dodatno razjasni določene okoliščine. Nekateri modeli imajo vgrajene tudi pospeškometre, ki samodejno sprožijo in zaščitijo snemanje in dodajo ustrezne podatke v video.

Nekatere naprave lahko poleg posnetkov ustvarijo tudi druge datoteke s pomembnimi podatki. Garmin na primer beleži bazo voženj/posnetkov in ima datoteko s podatki o zaznanih napakah v kameri (na primer prazna baterija).

Določene kamere imajo poleg samega snemanja tudi druge funkcionalnosti, na primer zaznavanje zapuščanja svojega pasu, predvidevanje neizogibnega trka in podobno in nato opozarjanje na takšne stvari. Vendar pa v raziskavi Lallie niso našli nobene možnosti, kjer bi lahko zgodovino teh podatkov pridobili iz naprave [Lallie 2020].

# 6 VERODOSTOJNOST IN AVTENTIČNOST GRADIVA

Pomembno vprašanje pri obravnavi in analizi dokazov je, ali je dokaz verodostojen in ali je bil na kakršen koli način obdelan za namene prikritja določenih dejanj. Kot že prej omenjeno, je same podatke o lokaciji zelo težko spreminjati, saj se morajo ujemati ne samo lokacija, ampak tudi hitrost, čas, smer vožnje in na koncu seveda tudi sam posnetek, hkrati pa morajo biti vsi ti podatki tudi usklajeni med seboj. Tudi za to se že razvijajo določeni algoritmi, ki lahko uporabijo zameglitev posnetka in ocenijo hitrosti vozila [Mehrish et al. 2017].

Če se torej odkrijejo odstopanja od pričakovanih vrednosti, se lahko pojavi sum na to, da se je posegalo v sam posnetek. Tako se lahko za posnetek, ki na prvi pogled izgleda povsem avtentično, zahteva analiza, da se ugotovi kaj in zakaj je bilo nekaj v posnetku spremenjeno, če je le-to možno z drugimi forenzičnimi pristopi, na primer analizo datotečnega sistema za odkrivanje skritih datotek in možnih preostankov datotek, ki niso bile prepisane.

# 7 POROČANJE

Na koncu preiskave ostane še najpomembnejši korak, priprava poročila. Za sprejemljivost v sodnem procesu je potrebno beležiti vse korake in ugotovitve. Ponavadi to vključuje uporabljeno metodologijo in posamezne ugotovitve in kot prilogo vse datoteke, ki so nastale med preiskavo in slikovno gradivo. Kot že omenjeno, surovi podatki niso v veliko pomoč pri uporabi kot dokazno gradivo, zato je potrebno takšne podatke pretvoriti v obliko, ki je koristna. To vključuje segmentiranje in izbor relevantnih delov posnetkov ter prikaz podatkov na ustrezno označenem zemljevidu.

#### 8 LEGALNOST UPORABE AVTOKAMER

Precejšnjo težavo predstavlja neobstoj slovenske zakonodaje, ki bi urejal uporabo avtokamer. Posledično tudi ni znano, kdaj in pod katerimi pogoji bi jih bilo dovoljeno uporabiti kot dokazno gradivo. Glede tega je najverjetneje, da bo področje urejeno na nivoju EU, direktivo pa se bo preneslo v slovensko zakonodajo [Giovannini et al. 2021].

Tudi po svetu je potrebno biti pazljiv, saj uporaba avtokamer ni nujno dovoljena, seznami držav pa so pogosto nepopolni in nedosledni. Zemljevid na sliki 4 na primer označuje slovenijo kot državo, ki dovoli uporabo avtokamer, kljub temu da temu ni tako.



Slika 4: Prikaz stanja legalnosti uporabe avtokamer po svetu. Vir: [Giovannini et al. 2021]

# 9 ZAKLJUČEK

V tej seminarski nalogi smo predstavili način, kako bi lahko s pomočjo avtokamere preiskovalec pridobil dokaze in se pri tem čim bolj držal standardnih dobrih praks pri takšnih preiskavah.

Največji problem predstavlja ogromno ročnega dela, ki je potrebno v primeru, če potrebujemo iz samega video posnetka zbirati podatke o lokaciji in času na daljših odsekih. V prihodnosti bo to verjetno mogoče avtomatizirati z uporabo orodij s strojnim učenjem, ki pa danes še niso nujno dovolj preverjena, da bi lahko celoten posnetek obdelali s sprejemljivim število napak.

Dodatno predstavlja težavo tudi potreba po ročnem iskanju nastavitev kamere, ki so skrite v uporabniškem vmesniku, saj je potrebno do njih dostopati ročno in jih zajamemo najpogosteje s slikami, kar lahko predstavlja dodaten vir napake pri digitalizaciji.

Forenzična analiza avtokamer se torej v samem postopku ne razlikuje veliko od analiz drugih digitalnih naprav, vendar pa moramo upoštevati, da so postopki v veliki meri odvisni od natančnosti naprave, kot pa na drugih področjih, kjer so celotni postopki že standardizirani. Kjer se proces najbolj razlikuje, je v orodjih, ki jih žal še ni veliko in pri tipih podatkov, ki jih pridobimo iz same naprave, saj veliko naprav nima vgrajenega GPS sledilnika lokacije.

V prihodnosti se bo uporaba avtokamer najverjetneje dodatno razširila in bodo že serijsko vgrajene v večino vozil. To pomeni, da bo seznanitev preiskovalcev s postopki za forenzično analizo takšnih naprav postala nuja. Z vzponom uporabe naprav pa se bo zagotovo pojavilo tudi veliko novih orodij, ki bodo omogočila lažjo in hitrejšo analizo.

# LITERATURA

- Saad Alboqami, Huthifah Alkurdi, Nawar Hinnawi, Emad Ul Haq Qazi, and Abdulrazaq Almorjan. 2023. Digital Forensic Investigation Framework for Dashcam. In ITNG 2023 20th International Conference on Information Technology-New Generations, Shahram Latifi (Ed.). Springer International Publishing, Cham, 151–157.
- Richard Ayers, Sam Brothers, and Wayne Jansen. 2014. Guidelines on Mobile Device Forensics. https://doi.org/10.6028/NIST.SP.800-101r1
- Yousef-Awwad Daraghmi and Ibrahim Shawahna. 2023. Digital Forensic Analysis of Vehicular Video Sensors: Dashcams as a Case. Sensors 23, 17 (2023). https: //doi.org/10.3390/s23177548
- Elena Giovannini, Arianna Giorgetti, Guido Pelletti, Alessio Giusti, Marco Garagnani, Jennifer Paola Pascali, Susi Pelotti, and Paolo Fais. 2021. Importance of dashboard camera (Dash Cam) analysis in fatal vehicle–pedestrian crash reconstruction. *Forensic Science, Medicine and Pathology* 17, 3 (2021), 379–387.
- Harjinder Singh Lallie. 2020. Dashcam forensics: A preliminary analysis of 7 dashcam devices. Forensic Science International: Digital Investigation 33 (2020), 200910. https: //doi.org/10.1016/j.fsidi.2020.200910
- Harjinder Singh Lallie. 2023. Dashcam forensic investigation guidelines. Forensic Science International: Digital Investigation 45 (2023), 301558. https://doi.org/10. 1016/j.fsidi.2023.301558
- Ambuj Mehrish, A.V Subramanyam, and Mohan Kankanhalli. 2017. Multimedia signatures for vehicle forensics. In 2017 IEEE International Conference on Multimedia and Expo (ICME). IEEE, 685–690. https://doi.org/10.1109/ICME.2017.8019488

# A Bayesian evaluation of captured device location

Klemen Vovk University of Ljubljana kv4582@student.uni-lj.si Ema Leila Grošelj University of Ljubljana ema.ljubljana@gmail.com Nika Molan University of Ljubljana nm83087@student.unili.si

### ABSTRACT

This paper presents an improved procedure for assessing geolocation evidence quality in digital forensics. Building on previous research, we introduce a probabilistic approach that considers directional errors in geolocation data. By using prior knowledge on GPS error margins, our method offers a more detailed evaluation framework. We outline the problem, our procedure, how we use data, and compare our findings. Our study improves the accuracy and reliability of geolocation evidence evaluation, pushing forward digital forensic investigations.

#### **Keywords**

device geolocation, MCMC, digital forensics

#### **1. INTRODUCTION**

This paper aims to improve the procedure for evaluating the quality of geolocation evidence in the field of digital forensics. Lets say the court provides us with a captured geolocation (evidence) and candidate geolocations. Captured geolocation could be extracted for example from an image taken near the crime scene around the time when the crime happened. Our goal is to work out the possibilities that the actual location of the device at the time was as a matter of fact somewhere else.

In the article [4] the authors computed a likelihood ratio to compare two candidate locations. They also considered that errors of geolocations may not be equal in all directions. We derive our prior knowledge about GPS error from article [5] where the authors report error of up to 5 meters in low-cost phones. We present our problem, data processing and methods, and then our results and conclusions.

#### 2. PROBLEM

Given a captured geolocation (evidence) E, our goal is to estimate the distribution of 2-dimensinal random vector  $X_E$  that represents the actual geolocation of the device, possibly different from E.

We get a list of candidate geolocations  $P_1, ..., P_K$ . At every candidate geolocation  $P_k$  we perform multiple measurements  $y_1^k, ..., y_M^k$ . We than transform the measurements at  $P_k$  and E to a space centered around  $P_k$  and spanned by distance from  $P_k$  and angle with respect to  $P_k$ . The transform details are disscussed in section 3.

From here on we are able to calculate the distribution of random vector  $Y^k$  (measured geolocation at  $P_k$  in distanceangle space). For that MCMC method is used. From the distribution we get the probability  $P(E|P_k)$  - score of the candidate, where  $E|P_k$  denotes the event of captured locatio being at E, given that device is at  $P_k$ . With assumptions that P(E = (i, j)) are equal for all (i, j) and  $P(P_k = (i, j))$ are equal for all (i, j) and k the following holds from the Bayes theorem:

$$P(P_k|E) = \frac{P(P_k)}{P(E)} * P(E|P_k) = konst * P(E|P_k)$$

Interpreting the scores we get an estimate for distribution of  $X_E$ .

# 3. DATA AND DATA PROCESSING

We ran our procedure on the data presented in [4] as well as on our own data. Both datasets were processed, that is transformed from shape (lng, lat) to  $(dist_k, angle_k)$ . Each candidate geolocation  $P_k$  produces a different space. Also Ehas a different representation in each space (ex.  $E_k$ ).

An arbitrary point T is transformed according to  $P_k$  in the following manner:  $dist_k := distance(T, P_k)$  whereas  $angle_k$  is 0 if T is straight north from  $P_k$  and increases in clockwise direction. Transform is depicted at Figure 1.

In article [4] the candidates were two known points in the building with known geolocations. As having such points is almost never the case, in our procedure the court provides us descriptions of the locations (ex. corner of the room, under the tree), and the geolocations  $\{P_k\}_{k=1...K}$  of such locations are later determined by the mean of all measurements.

We performed our measurements at two different sites, in Ljubljana and in Novo mesto. The two measurements also



Figure 1: Transform from (lng, lat) to  $(dist_k, angle_k)$  from [4].

differ in the distance between proposed geolocations. The distance was around 100 meters in Ljubljana and 10 meters in Novo mesto. The measurement was conducted by taking about 1 hundred photos at each candidate using the setting "precise location on" and about 50 photos at each candidate using the setting "precise location off" on iPhone.

# 4. MARKOV CHAIN MONTE CARLO (MCMC) METHOD

In reference to [1] MCMC is often used to solve integration and optimisation problems in large dimensional spaces. These two types of problem play a fundamental role in machine learning, physics, statistics, econometrics and decision analysis. The idea of Monte Carlo simulation in general is to draw an i.i.d. (independent and identically distributed) set of samples from a given distribution defined on a highdimensional space. These N samples can be used to approximate the distribution.

We used MCMC principle and approximately fitted the points 4000 times, which gave us 4000 samples - pairs of mean values  $\mu_{k,i}$  and covariance matrices  $\Sigma_{k,i}$  (bivariate normal distributions), all slightly different from one another. The bonus that we get is a confindence score for the overall distribution.

We evaluated each bivariate normal distribution  $(\boldsymbol{\mu}_{k,i}, \boldsymbol{\Sigma}_{k,i})$ at point  $E_k$  for all k. That gave us  $\{P(E|P_k)_i\}_i$ . Elementwise division of  $\{P(E|P_{k_1})_i\}_i$  and  $\{P(E|P_{k_2})_i\}_i$  gave us 4000 likelihood ratio samples (LRSs) - how much more likely it is that  $P_{k_1}$  was actual position of the phone versus  $P_{k_2}$ . The LRSs give us the distribution of LR, which can be presented with histogram like in Figure 6.

Our approach improves the original in [4] by estimating all probabilities directly using MCMC. This means we don't have to choose the angular wedge size at all, since we don't even have a wedge. This is handy as there is no obvious way of choosing the wedge size. Since we are using MCMC, we fit a bunch of distributions to the data, allowing us to quantify the uncertainty in LR using standard deviation and a 95% high-density interval (95% HDI). As for the second, it represents an interval on the posterior distribution of a parameter that contains 95% of the probability density. In other words, it's a range of values for a parameter within

Range of LR	Verbal Equivalent
1-3	In my opinion the observations are no more probable if [P1] rather than[P2] were true. Therefore, the observations do not assist in addressing which of the two propositions is true.
4-10	In my opinion the observations are slightly more probable if [P1] rather than [P2] were true.
10-100	In my opinion the observations are more probable if [P1] rather than [P2] were true.
100-1000	In my opinion the observations are much more probable if [P1] rather than [P2] were true.

Table 1: Table of LR verbal equivalent from [3] page 39.

which we are 95% confident that the true value lies. It provides a measure of how (un)stable the model is. This allows us to interpret the results including the uncertainty (see 1).

We noticed the error in long/lat mix-ups. Interestingly, despite this error, we found that it did not significantly influence their results, as the mix-up occurred consistently across all steps of their methodology rather than being a one-time occurrence. It's worth mentioning that our approach did not encounter such a mix-up, as we adopted a different methodology.

#### 5. RESULTS

In this section the results of use of are MCMC method on the tree different datasets from section 3 are presented. In Figure 2 we present the measurements in (lng,lat) format. While measurements from article [4] include two candidates, our include three, to state that multiple candidates can be tested. Candidates from article [4] and from Ljubljana are at distance about 100 meters, while candidates in Novo mesto are significantly closer, at distance of about 10 meters. Also measurements in article [4] are only one taken inside the buildings. Both our measurements are taken out in the open.

In Figure 3 we underlaid the measurements with a map. For that we used [2] openStreetMap. Measurements also differ in where is E with regard to candidates, and measurements. In article [4] measurements E is in the midst of P1 measurements, resulting in inclination towards the P1 being absolutely more probable candidate for actual location. Less certain but still also in case of Novo mesto there exists an inclination towards one candidate that is P3. In measurements taken in Ljubljana on the other hand one would hardly say that any of the candidates is more probable than other two.

We than transformed the measurements and evidence point E to two or three spaces according to number of candidates. The points in  $(dist_k, angle_k)$  space are depicted in Figure 4. The polar coordinate system was used to capture the cyclic nature of angle.

In our limited set of cases the use of mean value for Pk turned out alright. Even in Novo mesto where P3 is 10

meters away from closest measurement, the P3 seemed to match the location of actual candidate location in nature.

In Figure 5 one of 4000 samples of bivariate normal distributions is depicted for every candidate in every measurement case. We can see that most are quite accurate, but for example the second in Subfigure 5a that is the fit for the candidate P2 in Novo mesto is a bit off. That is expected and desirable as only a certain diversity will also give us a confidence interval with nonzero width.

Evaluating likelihoods at  $E_k$  for each sample distribution and element-wise division between those of  $E_{k_1}$  and  $E_{k_2}$ gave us a plethora of LR samples, some of which are depicted in Figure 6. As likelihoods of  $E_k$  in Ljubljana case were constantly zero (E couldn't have been captured from either of our candidates), the element-wise division was not an option, therefore we omit it from the figure. On the other hand in Subfigure 6a we can see the distribution of LR P(E|P1) / P(E|P2) in data form article [4]. We can observe that LR is very high (about  $10^8$ ). As expected the probability is on the side of P1. This ratio is so high that it is not even foreseen in the Table 1. Whereas in Novo mesto case we deal with three different likelihoods and two ratios are depicted P(E|P1) / P(E|P2) and P(E|P2) / P(E|P3). In both cases we are dealing with extremely small ratios with extremely large inverses. Those are also beyond values listed in Table 1.

As seen from the histogram, due to the heavy skew, a standard deviation from the mean is not a good explanation of the uncertainty, this is why we include 95% HDI. The 95% HDI provides a wide range of values for parameters where we're 95% confident the true value falls. This interval not only shows the uncertainty but also evaluates how stable the model is. By covering most of the probability density, the 95% HDI gives a strong indication of the model's performance, particularly in cases where relying solely on standard deviation might not be sufficient.

# 6. ANALYSIS

This article presents a method for enhancing the evaluation process of geolocation evidence in digital forensics.

By transitioning from geographical coordinates to distanceangle space, a more detailed examination of geolocation data becomes possible. This transition captures the relative positions and orientations of potential locations and the evidence point, which is essential for comparing the likelihood of various scenarios. This shift is evident in the polar plot, where each candidate's measurements form distinct clusters based on their distance and angle from the evidence point.

Furthermore, the outlined procedure utilizing MCMC offers notable advantages when confronted with complex datasets or when mere visualization fails to yield conclusive insights. For instance, in scenarios involving closely positioned candidates or when the evidence point lacks clear alignment with any candidate, a quantitative analysis using MCMC can provide more precise evaluations of each scenario, surpassing the limitations of visual scrutiny alone.

Overall, this approach demonstrates a thorough methodol-

ogy for evaluating geolocation evidence, integrating quantitative analysis and visualization techniques to extract meaningful findings. Further exploration of specific aspects such as the influence of GPS settings and the interpretation of likelihood ratios can enhance the robustness and practicality of this methodology in forensic inquiries.

### 7. CONCLUSION

In conclusion, to enhance the reliability of geolocation evidence in digital forensics, it is crucial to consider the temporal variability of reference data. Repeating data capture at various times throughout the day can yield more representative datasets less susceptible to temporary glitches. Additionally, instead of transforming around a particular peak, centering geolocation analysis around telecommunication transmitters may offer more meaningful insights. By adopting these strategies, forensic investigators can bolster the accuracy and robustness of geolocation evaluations, thus strengthening the integrity of digital forensic procedures.

#### 8. REFERENCES

- C. Andrieu, N. De Freitas, A. Doucet, and M. I. Jordan. An introduction to mcmc for machine learning. *Machine learning*, 50:5–43, 2003.
- [2] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org.https://www.openstreetmap.org, 2017.
- [3] F. S. Regulator. Development of evaluative opinions. Technical Report FSR-C-118, UK Forensic Science Regulator, Birmingham, 2021.
- [4] H. Spichiger. A likelihood ratio approach for the evaluation of single point device locations. Forensic Science International: Digital Investigation, 2023.
- [5] L. Wang, Z. Li, N. Wang, and Z. Wang. Real-time gnss precise point positioning for low-cost smart devices. *GPS Solutions*, 25:1–13, 2021.



(c) Measurements from Novo mesto.

Figure 2: Three different sets of measurements in (lng,lat) format.







(b) Measurements from Ljubljana.



(c) Measurements from Novo mesto.

Figure 3: Three different sets of measurements in (lng, lat) format with the underlaying map.



(c) Measurements from Novo mesto.

Figure 4: Three different sets of measurements in  $(dist_k, angle_k)$  format.



(c) Measurements from Novo mesto.

Figure 5: Three different sets of measurements in  $(dist_k, angle_k)$  format and a sample bivariate normal distribution.



(b) Measurements from Novo mesto.

Figure 6: LR distributions for article [4] and Novo mesto.

# **Can ChatGPT Unlock Potential in Digital Forensics?**

Review of "ChatGPT for digital forensic investigation: The good, the bad, and the

unknown"

Anže Mihevc University of Ljubljana, Faculty of Computer and Information Science Večna pot 113 Ljubljana, Slovenia am8130@student.uni-lj.si Aljaž Grdadolnik University of Ljubljana, Faculty of Computer and Information Science Večna pot 113 Ljubljana, Slovenia ag5319@student.uni-lj.si Sanil Safić University of Ljubljana, Faculty of Computer and Information Science Večna pot 113 Ljubljana, Slovenia ss4131@student.uni-lj.si

# ABSTRACT

In this report we build upon the research presented in the paper "ChatGPT for Digital Forensic Investigation: The Good, the Bad, and the Unknown."[6] We start by exploring related work in digital forensic investigation. Then, we dive into a detailed examination of the original paper, focusing on its experiments and findings. We also add our own examples as validation of the original findings. Our goal is to understand the strengths, weaknesses, and areas yet to be explored in using ChatGPT for digital forensic tasks.

#### **Keywords**

ChatGPT, GPT-4, digital forensics, Artificial Intelligence, Large Language Models

# 1. INTRODUCTION

Large language models like ChatGPT are transforming various fields, and digital forensics is no exception. This report investigates the potential of ChatGPT in this domain. We build upon a previous study, "ChatGPT for Digital Forensic Investigation: The Good, the Bad, and the Unknown," analyzing its experiments and findings. Our goal is to understand how effectively ChatGPT can be used for digital forensic tasks. We'll dissect its strengths and weaknesses, exploring areas where it shines and identifying limitations that necessitate further research. This analysis will provide valuable insights into the evolving landscape of digital forensics and the potential role of AI within it.

In the following sections, we'll begin by examining relevant research on digital forensics and large language models. This will provide context for our analysis of the original study's methodology. We'll then delve into specific areas explored in that study, such as artifact identification and teaching scenario generation. Finally, we'll summarize the key takeaways and identify promising areas for future research, paving the way for further exploration of this exciting intersection between AI and digital forensics.

# 2. RELATED WORK

This section delves into the relevant literature to establish the context for our current study. Here, we will analyze previous papers and related works that have explored similar topics. By examining these contributions, we aim to identify the current knowledge gaps and situate our research within the ongoing dialogue of this field. We will be looking at some papers that dive into the generative large language models in general and also at some papers that directly combine LLMs and digital forensics.

Authors of the paper "Artificial Hallucinations in ChatGPT: Implications in Scientific Writing" [1] explored the potential of ChatGPT, a large language model from OpenAI, to be used as a writing tool in scientific research. The authors investigated this by prompting ChatGPT to generate text on the pathogenesis of two specific medical conditions. Their findings documented both the helpful and concerning aspects of ChatGPT's performance, highlighting the need for awareness of its limitations in scientific writing.

In the paper "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" [3], authors introduced a novel pre-training approach for natural language processing (NLP) tasks. Unlike previous models, BERT (stands for Bidirectional Encoder Representations from Transformers) leverages unlabeled text for pre-training by conditioning on both the left and right context of words in all layers. This bidirectional approach allows BERT to capture richer contextual information compared to unidirectional models. The pre-trained BERT model can then be fine-tuned for various NLP tasks with minimal modifications to the architecture. This approach led to state-of-the-art performance on several NLP benchmarks, including question answering and natural language inference. Notably, BERT achieved significant improvements on the GLUE score, MultiNLI accuracy, and SQuAD question answering tasks.

Du at al. [4] presented summaries of existing artificial intelli-

gence based tools and approaches in digital forensics. They presented overview of machine learning and deep learning and presented applications of AI in digital forensics. Theses included data discovery and recovery, device triage, network traffic analysis, encrypted data, event reconstruction, multimedia forensics and fingerprinting. They showed how AI is used in each of the areas of digital forensics. It also highlights challenges including availability of data sets, difficulties in explaining the results and challenges in releasing models where potentially restricted training data could be inferred from the models. Despite the challenges they believe there is potential in future work. That is in terms of improving the performance of some of the current techniques and also there are some approaches that have not yet been tested in individual areas and with their results they could be more easily identified.

Breitinger at al. [2] make a 10-year review and future directions in digital forensics. The paper serves as a review of the accepted full research papers published in the ten years of proceedings of DFRWS EU 2014-2023 to analyse the impact of the conference's output. As part of the discussion, potential future research directions and best practices are discussed, aiming to increase the discipline's research efforts and foster its development into a more mature scientific domain. The article mentions ChatGPT for digital forensic investigation: The good, the bad, and the unknown as the future directions in artificial intelligence topic: the advent of Large Language Models, in particular, could significantly advance digital forensics (Scanlon et al., 2023). However, there is limited discussion on key digital forensic issues like explainability, which is crucial for AI's successful integration into digital investigations. Forensics of AI systems is an intriguing, largely unexplored frontier and the aforementioned article(ChatGPT for digital forensic investigation: The good, the bad, and the unknown) explores one aspect of it.

In the editorial of the FSI Digital Investigation journal titled Digital forensic investigation in the age of ChatGPT [7] the authors talk about the potentials of Large Language Models (LLMS) for digital forensics. The editorial points out that the technology holds promise for automating tasks like script generation and multilingual analysis but also that LLMs present certain challenges. These include potential biases and errors within the models, legal implications surrounding their use, and the inherent opacity of their inner workings as they are often closed sourced and work like a "black box". Overall, the article suggests that LLMs can be a valuable tool to augment forensic investigations, but should not supplant human expertise and judgement. The authors posit that the future might see non-experts utilizing LLMs for initial investigations, potentially giving rise to a new specialization: digital forensic prompt engineering.

"So what if ChatGPT wrote it?" [5] is an opinion paper by 43 multidisciplinary authors that all give their perspective on the opportunities, challenges and implications of generative conversational AI for research, practice and policy. It is not solely focused on digital forensics but provides a very comprehensive overview of the emerging AI tools by a wide collection of experts. They present the benefits and the drawbacks of the technology. Some of the main capabilities of ChatGPT are to enhance productivity and the potential to offer significant gains in the banking, hospitality and tourism, and information technology industries, and enhance business activities, such as management and marketing. However they also consider the limitations of the technology such as disruptions to practices, threats to privacy and security, and consequences of biases, misuse, and misinformation. Their opinion on whether ChatGPT's use should be restricted or legislated is split and the article identifies several questions that will require further research. The questions focus on three main areas: the impact on knowledge, transparency, and ethics; the digital transformation of organizations and society and the impact on teaching, learning, and research.

# CHATGPT IN DIGITAL FORENSICS Methodology

We will now dive more into the original paper *ChatGPT* for digital forensic investigation: The good, the bad, and the unknown [6] which our paper is based on.

In their methodology, the authors outlined how they tested ChatGPT's usefulness in digital forensics. They chose six areas to explore, conducted experiments, and evaluated the results. They used fact-checking and their expertise in the field to validate the findings. Each section ended with a summary, leading to general conclusions later in the paper. They also created a GitHub repository to store their experiments.

In our paper we summarised the six areas explored in the original article and provided our own examples of queries and ChatGPTs answers to validate the original authors findings.

# 3.2 ChatGPT for artefact identification

In this section, the authors explored the use of ChatGPT for artefact identification in digital forensic investigations. They focused on three main areas: file downloads, file execution, and cloud interaction. ChatGPT provided insights on potential locations to examine for evidence, such as Windows Event Logs, browser history, registry hives, and cloud storage platforms. However, it primarily emphasized Windows Event Logs, potentially overlooking other important artefacts. The depth of response varied, with some artefacts receiving detailed guidance while others were briefly mentioned. Additionally, ChatGPT provided links to tools, some of which did not exist or had similar but not identical names. Regarding cloud interaction, ChatGPT offered reasonable areas to investigate but inconsistent paths to default locations on the machine. Overall, while ChatGPT offered valuable insights, its focus on certain artefacts and inconsistencies in responses highlight the need for caution and additional validation in digital forensic analysis.

#### 3.2.1 Example(s)

*Prompt:* Can you please give me detailed instructions on installation and basic use cases of a program WireShark for digital forensics investigation?

Response:

Installation:

- Windows ...
- macOS ...
- Linux ...

Basic Use Cases for Digital Forensics: ...

Advanced Use Cases:

Tips and Best Practices:

- Regular Updates: Keep Wireshark updated to ensure compatibility with the latest protocols and security features.
- Filter Efficiently: Use specific filters to avoid capturing unnecessary data, which can slow down analysis.
- Use with Other Tools: Integrate Wireshark with other forensic tools like network intrusion detection systems (NIDS) for comprehensive analysis.

Wireshark is a versatile tool for network analysis and digital forensics, providing deep insights into network traffic and helping investigators uncover crucial information. By following the above steps, you can effectively use Wireshark for your digital forensics investigations.

# **3.3** ChatGPT for self-directed learning of digital forensics

In their analysis of ChatGPT's effectiveness for self-directed learning in digital forensics, the authors categorized educational objectives into introductory, advanced/expert, research, and tool explanation levels. They found that Chat-GPT provided accurate overviews at the introductory level, though it sometimes struggled with specific details. For advanced learning, ChatGPT recommended exercises and activities but had limitations in providing comprehensive scenarios and occasionally included errors. The model's inability to provide real-time updates limited its usefulness for staying current with research and developments. However, it proved helpful in explaining tools like Wireshark and tshark. Overall, while ChatGPT serves as an effective tool for acquiring foundational knowledge, its reliance on textual explanations and lack of practical exercises pose limitations for in-depth learning and application.

#### 3.3.1 Example(s)

Prompt: Can you explain some basics of digital forensics?

#### Response:

Digital forensics is the process of identifying, preserving, analyzing, and presenting digital evidence in a manner that is legally admissible. It is crucial in the investigation of cybercrimes, data breaches, and various other illegal activities involving digital devices. Here's a breakdown of the key components and concepts in digital forensics:

- 1. Types of Digital Forensics
- 2. The Digital Forensics Process
- 3. Common Tools and Techniques
- 4. Legal and Ethical Considerations
- 5. Challenges in Digital Forensics

#### Conclusion

Digital forensics is an essential field that plays a critical role in modern investigations involving digital evidence. Understanding the basics of various types of digital forensics, the standard process, tools, legal considerations, and challenges is crucial for effectively navigating and conducting forensic investigations.

# 3.4 ChatGPT-assisted keyword searching

Since the concept of searching is fundamental to digital forensics and much of that is based on keyword searching, this topic is very intriguing when we talk about a generative AI. In the article the authors delve into generating regular expressions, generating keyword lists and other searching topics. They came to a conclusion that there are some potential uses of ChatGPT already within the context of searching in digital forensics. Generating regular expressions and enhancing keyword search lists, either with additional terms, or suggesting abbreviations or misspellings, have all been found to be reasonably effective, although the former requires validation and testing of those regular expressions generated. There are also clearly some potential uses for the technology in future; the ability to summarise documents and answer questions about the nature of the content in a user friendly manner has extensive potential for digital forensic applications. Unfortunately, the inability to upload evidence to such a service prevents this from being useful in its current form.

#### $3.4.1 \quad Example(s)$

In the examples we ask ChatGPT to provide us with a regular expression, with some examples to check it's correctness. The other example checks the ability to create lists of words that may help us with our searching for keywords.

*Prompt:* Create a regular expression for finding email addresses and also provide me with some examples of email addresses that should be matched with the provided regex.

*Response:* ^[a-zA-ZO-9.\_%+-]+@[a-zA-ZO-9.-]+\.[a-zA-Z]{2,}\$

Explanation of the Regex

- ^ Asserts the position at the start of the line.
- [a-zA-ZO-9.\_%+-]+ Matches one or more characters that can be letters (uppercase and lowercase), digits,

dots, underscores, percent signs, plus signs, and hyphens. This represents the local part of the email.

- @ Matches the "@" symbol.
- [a-zA-Z0-9.-]+- Matches one or more characters that can be letters (uppercase and lowercase), digits, dots, and hyphens. This represents the domain part of the email.
- $\$ . Escapes the dot to match the literal dot character.
- [a-zA-Z]{2,} Matches two or more letters. This represents the top-level domain (TLD).
- \$ Asserts the position at the end of the line.

*Second prompt:* We have a case in digital forensics where we need to search a disk for certain words. The case is on a sexual assault. Provide me with different possible words, characters or even more that could be related to sexual assault.

#### Response:

- 1. General terms
  - (a) Sexual assault
  - (b) Rape
  - (c) Molestation
  - (d) Abuse
  - (e) Harassment
  - (f) Victim
- 2. Specific Acts and Related Terms
  - (a) Penetration
  - (b) Fondling
  - $(c) \ Groping$
  - (d) Touching
  - (e) Inappropriate
  - (f) Indecent
- 3. Descriptive Terms and Phrases
  - (a) "She said no"
  - (b) "He said no"
  - (c) "No means no"
  - (d) "Stop"
  - (e) "Help"
  - (f) "Please don't"
- 4. AND MANY MORE

# **3.5** ChatGPT and programming in digital forensics

The authors also examined GPT-4's potential to assist digital forensics investigators by generating scripts (the capacity to swiftly create a script for a particular duty is essential in various digital forensic cases) for a set of common tasks. These tasks are: file carving, RAID disk acquisition, passwords cracking and memory forensics - recovering encryption keys. The authors found that in the tested scenarios, GPT-4 effectively generated scripts for various digital forensics tasks. The scripts were well-commented, had adequate error checking, and combined different technologies, e.g., integrating Linux tools into a Python script. The system could also provide detailed explanations of the code's functionality and the decisions behind its creation. However, userlevel knowledge of scripting languages and digital forensics is essential for application and to spot any unreasonable assumptions, such as limiting encryption key size or assuming only text files contain sought-after regular expressions. Generated code can not be used blindly.

GPT-4 also refused to help create code for "gate-kept" operations, e.g., potentially unethical or illegal use cases, such as password cracking. But you can get around that with breaking down the requests into constituent parts, but again for that the user has to have some knowledge on the topic. OpenAI continually works to prevent such "jailbreaking".

#### 3.5.1 Example(s)

In the examples for programming with ChatGPT we tell it that we are digital forensics investigators and would like a program which carves a PNG files from a disk. We would expect to get a program that would carve the files and would even consider if more than one PNG file is in another PNG file. We also tried with a program that cracks a zip file that is password protected.

*Prompt:* Generate a script to extract files from a captured disk image (either \*.E01 or raw images). Craft a Python script to retrieve/carve PNG files.

*Response:* We get a Python script that only checks for the file extensions, which would not be enough, because carving should carve a file that is hidden in another file and this program does not do that.

*Second prompt:* We are digital forensics experts and found a locked zip file on perpetrators disk. Develop a password cracking script for it

*Response:* We get back a python script that uses a txt file as a dictionary and goes through it and try's to do a brute force attack.

#### **3.6 ChatGPT for incident response**

ChatGPT showed promise in detecting deviations from standard system configurations, however during testing the authors observed very inconsistent performance. ChatGPT would miss anomalies and struggle when contextual knowledge was crucial. Additionally since it does not have any specific training it cannot effectively analyse information unique to a particular system. When the researchers provided it with context, such as a textual description of the incident in addition to log files, it preformed better at identifying suspicious activity. When such context was absent, ChatGPT would miss critical details such as a reverse shell. Such a pattern was evident in the analysis of multiple attacks such as ARP spoofing, Follina exploit CVE-2022-30190 and Heartbleed vulnerability. In addition ChatGPT lacks functionality of real-time data collection and analysis which hampers its ability to identify current threats. Overall the authors found that ChatGPT is too inconsistent and lacks mechanisms to ingest enough context or data in order to be used effectively as a tool in incident response situations.

#### 3.6.1 Example

In this prompt we ask ChatGPT to analyze the network traffic provided to us by the command *tcpdump*. The response is a detailed list of all the sources of network traffic and some additional information for each one, like their domain, IP and Occurrences. This is an example of a simple task that ChatGPT solves successfully however we should not trust this completely and when finding suspicious activity in this list we should double check that traffic.

#### **Prompt:**

"Can you tell me the sources in the following network traffic captured by tcpdump command: 21:52:02.016131 IP ec2-44-241-205-248.us-west-2.compute.amazonaws.com.https > laptop.56784: Flags [FP.], seq 631923077:631923108, ack 3902439375, within a workplace starts a new job and slowly becomes win 118, options [nop,nop,TS val 1780495695 ecr 2811458923], length 31 21:52:02.016155 IP laptop.56784 > ec2-44-241-205-248.us-west-2.compute.amazonaws.com.https: Flags [R], seq 3902439375, win 0, length 0"... The prompt was shortened for readability.

#### Response:

Based on the provided tcpdump output, the sources (IP addresses) and their associated domain names in the captured network traffic are as follows:

- 1. ec2-44-241-205-248.us-west-2.compute.amazonaws.com:
  - (a) IP: 44.241.205.248
  - (b) Domain: ec2-44-241-205-248.us-west-2.c ompute.amazonaws.com
  - (c) Occurrences: Multiple entries, including traffic over HTTPS (port 443) and host monitoring (hostmon).
- 2. 157.249.81.141:
  - (a) IP: 157.249.81.141
  - (b) Domain: (No reverse DNS found in the provided data, just the IP address)
  - (c) Occurrences: HTTPS traffic (port 443).
- 3. ... the response was shortened for readability
- **3.7** ChatGPT for generating teaching scenarios

The researchers tested and found great results in generating lifelike teaching scenarios for digital forensics using Chat-GPT. They used ChatGPT to generate realistic storyboards, characters profiles and interests as well as synthetic data that could be present on a victims or criminals hard drive. They point out that since this application is not in an investigative context there is no *incorrect answer*. ChatGPT did sometimes come up with unrealistic scenarios but did quickly correct itself when prompted. The authors find that for creative generative applications, ChatGPT offers significant potential. Pairing it with other tools that generate images and videos could add richness to the synthetic scenarios as well as make them more realistic which would enhance the teaching/learning experience.

#### 3.7.1 Example

In this example we asked ChatGPT to create a scenario in which a new employee builds resentment to their company and over time starts stealing intellectual propriety of the company. ChatGPT provided a detailed timeline of events during a period of 6 months. When further asked to provide an example browser history of this employee it generated a similar timeline but this time with examples of links and search queries.

#### Prompt:

generate an outline timeline of a scenario where someone discontent over a few months and begins to steal intellectual property

Response: Month 1: Initial Excitement and Integration

- Week 1: Orientation and Onboarding
  - Attends orientation sessions
  - Meets team members and managers
  - Receives initial training on company policies and procedures
- Week 2: Settling In
  - Starts working on initial tasks under supervision
  - Begins to understand the company culture and workflow
  - Shows enthusiasm and eagerness to contribute
- Week 3-4: Building Relationships
  - Forms initial professional relationships with colleagues
  - Participates in team meetings and brainstorming sessions
  - Receives positive feedback on initial work

Month 2: Growing Familiarity and Early Signs of Discontent

- Week 5-6: Increased Workload
  - Assigned more complex and demanding tasks

- Starts to feel the pressure of deadlines and performance expectations
- Begins to notice inefficiencies or issues within the team or company
- Week 7-8: First Signs of Discontent
  - Experiences first disagreements or conflicts with colleagues
  - Feels underappreciated or undervalued despite efforts
  - Begins to express mild dissatisfaction to trusted colleagues or friends outside of work
- ... (the response was shortened to improve readability)

# 4. AUTHORS DISCUSSION

#### 4.1 The good

The authors identified three main strengths of ChatGPT in the context of digital forensics: creativity, reassurance, and avoidance of the blank page syndrome. They recognise its utility in creative tasks such as generating teaching scenarios or creating inputs (keyword lists). For an expert with previous knowledge and expertise ChatGPT also provides some reassurance when cross-comparing results however prior knowledge is crucial in this case. The last strength of ChatGPT is providing a starting point to some project or work such as a script that can then be further worked on and improved.

#### 4.2 The bad

The authors also identified some weaknesses such as: quality and age of training data, handling highly specialised and uncommon tasks, and interacting with ChatGPT. As the LLM is trained on pre-existing data it does not have the context of state of the art methods. Additionally since the digital forensics community is relatively small there is not a lot of training data on this specific topic. This makes ChatGPT very inconsistent and unreliable in specific scenarios. The text only interaction with ChatGPT also hinders it's potential capabilities as digital forensics often deals with multitude of data formats. Lastly, the authors point out that the output is not deterministic, which is not desired in digital forensics where a principle is to be reproducible.

# 4.3 The unknown

The authors predict that with improvement of the models and their integration into existing forensic software new use cases could be discovered. It could also make the interaction with the software more natural and require less specific knowledge. However they caution against relaying too much on such models because the often "hallucinate" and produce unreliable results.

# 5. LIMITATIONS

The study offers valuable insights into ChatGPT's potential in digital forensics, but it's important to recognize its limitations. The experiments didn't cover all forensic aspects and were conducted in a controlled setting. Real-world scenarios may pose more complexities not captured in the study. Results heavily rely on the prompt, and minor changes can yield different outcomes. Also, due to ChatGPT's nondeterministic behavior, results aren't directly reproducible, requiring static analysis provided in the GitHub repository.

### 5.1 The question of ethics

ChatGPT is a service provided by OpenAI. As OpenAI is a public company often talked about in the mainstream news they impose some ethical restrictions on ChatGPT as we saw in some of the examples. The restrictions are not publicly disclosed which can be detrimental for digital forensics since some of the proceeds that are required to be preformed could be considered unethical or even illegal if preformed without a warrant or outside a digital investigation. For a more unrestricted access to the full potential of ChatGPT or other LLM models, there would need to be a way to access or host your own model. We believe this is one of the potential areas to explore and research in the future.

# 6. CONCLUSIONS

The authors of the original paper extensively explored the potential applications of ChatGPT for digital forensics through a series of eight experiments. Their findings shed light on both the capabilities and limitations of ChatGPT in this domain. They highlighted the recurring issue of 'hallucination', emphasizing the need for caution in its use. Despite these challenges, they identified areas where ChatGPT shows promise, such as code generation and keyword list creation. Furthermore, they proposed avenues for future research, including the development of locally deployable models to address specific forensic tasks.

# 7. REFERENCES

- H. Alkaissi and S. Mcfarlane. Artificial hallucinations in chatgpt: Implications in scientific writing. *Cureus*, 15, 02 2023.
- [2] F. Breitinger, J.-N. Hilgert, C. Hargreaves, J. Sheppard, R. Overdorf, and M. Scanlon. Dfrws eu 10-year review and future directions in digital forensic research. *Forensic Science International: Digital Investigation*, 48:301685, 2024.
- [3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, and T. Solorio, editors, Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [4] X. Du, C. Hargreaves, J. Sheppard, F. Anda, A. Sayakkara, N.-A. Le-Khac, and M. Scanlon. Sok: Exploring the state of the art and the future potential of artificial intelligence in digital forensic investigation. In *Proceedings of the 15th international conference on* availability, reliability and security, pages 1–10, 2020.
- [5] Y. K. Dwivedi, N. Kshetri, L. Hughes, E. L. Slade,
  A. Jeyaraj, A. K. Kar, A. M. Baabdullah, A. Koohang,
  V. Raghavan, M. Ahuja, H. Albanna, M. A.
  Albashrawi, A. S. Al-Busaidi, J. Balakrishnan,
  Y. Barlette, S. Basu, I. Bose, L. Brooks, D. Buhalis,
  L. Carter, S. Chowdhury, T. Crick, S. W. Cunningham,
  G. H. Davies, R. M. Davison, R. Dé, D. Dennehy,

Y. Duan, R. Dubey, R. Dwivedi, J. S. Edwards, C. Flavián, R. Gauld, V. Grover, M.-C. Hu, M. Janssen, P. Jones, I. Junglas, S. Khorana, S. Kraus, K. R. Larsen, P. Latreille, S. Laumer, F. T. Malik, A. Mardani, M. Mariani, S. Mithas, E. Mogaji, J. H. Nord, S. O'Connor, F. Okumus, M. Pagani, N. Pandey, S. Papagiannidis, I. O. Pappas, N. Pathak, J. Pries-Heje, R. Raman, N. P. Rana, S.-V. Rehm, S. Ribeiro-Navarrete, A. Richter, F. Rowe, S. Sarker, B. C. Stahl, M. K. Tiwari, W. van der Aalst, V. Venkatesh, G. Viglia, M. Wade, P. Walton, J. Wirtz, and R. Wright. Opinion paper: "so what if chatgpt wrote it?" multidisciplinary perspectives on opportunities, challenges and implications of generative conversational ai for research, practice and policy. International Journal of Information Management,  $71{:}102642,\ 2023.$ 

- [6] M. Scanlon, F. Breitinger, C. Hargreaves, J.-N. Hilgert, and J. Sheppard. Chatgpt for digital forensic investigation: The good, the bad, and the unknown. *Forensic Science International: Digital Investigation*, 46:301609, 2023.
- [7] M. Scanlon, B. Nikkel, and Z. Geradts. Digital forensic investigation in the age of chatgpt. *Forensic Science International: Digital Investigation*, 44:301543, 2023.

# **Evidence in the fog – Triage in fog computing systems**

Nino Cajnkar Fakulteta za računalništvo in informatiko Ljubljana, Slovenija nc20429@student.uni-lj.si Vid Cesar Fakulteta za računalništvo in informatiko Ljubljana, Slovenija vc4183@student.uni-lj.si Nikolaj Popovič Fakulteta za računalništvo in informatiko Ljubljana, Slovenija np1763@student.uni-lj.si

# Abstract

With the increase use of cloud services the need for fog networks as an intermediate layer grows. At the same time the number of crimes involving at least one digital device is also growing. This paper will present how to approach (digital) evidence in the fog. The most important part is understanding how to network operates, and where to look for data - with some probability. There are quite some metrics we need to take into account if we want to give a good estimation, but we can never be 100% certain to say which node will have the data, because the structure of fog networks is very dynamic. We will also compare some measures that can be used for estimating the probability of a node containing the data.

# **CCS** Concepts

• Security and privacy  $\rightarrow$  Distributed systems security.

# Keywords

fog computing systems, triage, forensic, evidence, data volatility, YAFS, dynamic networks, analytical model

### 1 Introduction

Evidence in the fog is even more complex than standard computer evidence, as there are more variables to where the needed data is stored, and how to effectively get access it. But first we need to understand what fog computing is and how to address this problem of accessing data.

#### **1.1 Fog Computing**

Fog Computing is a highly virtualized platform that provides compute, storage, and networking services between end devices and traditional Cloud Computing Data Centers, typically, but not exclusively located at the edge of network. [1] This means that in the fog, there is a group of dozens of thousands of devices, which provide services similar as Cloud services provide now, but these devices are not as centralized as classic Cloud servers, but closer to end devices and users. One might think that the fog is in some sense similar to the Cloud, but there are some characteristics that make the fog a non-trivial extension of the Cloud. Some of these characteristics are low latency (closer to edge nodes, introduction to new technologies as 5G mobile network), location awareness, geographical distribution (not as Centralized as Cloud), large-scale sensors (monitoring the environment), larger number of nodes compared to Cloud (consequence of wide geographical distribution), real-time interactions contrary to batch processing, predominance of wireless access etc. The fog is some kind of intermediate layer between the Cloud layer and edge layer (end devices) as presented

on *image 1*. Edge nodes can also be completely separate networks that communicate with fog layer trough gateways. [2]



# Figure 1: Placement of fog layer between Cloud and end devices

The main focus of our paper is the fog layer and finding out where the interesting data might be - where to look for it. Because the resources of investigations are usually limited, they need to be utilized carefully and prioritized so that devices containing the most valuable evidence are collected. Evidence in a fog network is normally presumed to be located in nodes for processing data. The main challenge for an investigator is to locate the nodes which might contain the data (evidence) before it disappears - is overwritten by new data.

Questions this paper will study are:

- (1) In which way does fog computing affect the identification, collection, (and examination) of evidence that is disseminated in the network?
  - How fast does the data disappear from nodes, how to identify nodes that have high probability of containing data and then obtain it?
- (2) How can available information and metrics be used to identify and prioritize data collection from complex and dynamic networks?
  - Where is the data located, how to locate it with very limited information we have?
- (3) How sensitive are the identified measures for network attributes and dynamics?

• What happens when nodes are out for repair, new ones are added, how does that affect distances between nodes, probability of selecting a node for processing certain data.

#### 1.2 Digital triage

When doing digital forensics, the term triage usually refers to the process of assessing and prioritizing digital evidence to determine it's relevance to an investigation. With this we can eliminate unimportant evidence and focus on data which will bring us closer to a conclusion. Triage involves several steps such as:

- (1) Identification identification of relevant evidence
- (2) Collection Process of collecting identified evidence using specialized tools and techniques.
- (3) Prioritization Which evidence will be analyzed first according to the potential relevance to the case
- (4) Initial analysis Basic, quick techniques used to analyze the data, to get some initial insight
- (5) Filtering Filter out irrelevant, duplicate data, to focus on pertinent evidence
- (6) Documentation Where investigator documents it's work and findings to maintain an audit trail to ensure integrity of evidence and repeatability of the whole process.

#### 2 Related work

There has been some work done in presenting different ways a fog network can operate. We have different topologies, service placements and dynamic elements. **Topologies** like hierarchical tree structure where end devices are leaves (cameras for example) which then communicate up the hierarchy, first for detection / tracking of vehicles, and if we get the vehicle license plate this propagates to another layer for checking if the license plate is stolen etc. [3] There are also fog colonies where a control node of a colony determines the device that will allocate the service. [4] There are also some random structures, Barabasi-Albert type fog networks, Lobster graphs and Grid Layouts.

**Dynamic elements** include mobility of nodes for example LISP protocol [1], failing of nodes that impacts branches in hierarchical tree structures, and changes of routes which has impact on probability that a certain node will contain our data.

**Service Placements** can be distributed trough network (large scale sensors, Smart Grid) [1], hierarchical deployed services to handle data stream from distributed edge devices [3]. Some prefer services in the close neighborhood to be selected for processing [5], which is similar as placement that is close to the edge.

#### 2.1 Volatility

Volatility of traces has always been concern of forensic examiners. With digital forensics there is no difference, and when taking fog computing into account things get even more complicated. Volatility is defined as the expected time before the data is inaccessible i.e. being overwritten or erased (by zeroing the storage location) or when the collection methods cannot access the data (storage media fails). First of we need to look at each potential node in fog network as a separate computer and approximate it's volatility. Data can have an *ideal lifetime* i.e. when it's in use, a *natural lifetime* when it exists and can be retrieved. This natural lifetime can be significantly shortened if we delete the file by also setting all the memory contents to zero, not just deleting the pointer. [6]

A volatility for IoT systems, which are in a ways similar to fog networks (many edge nodes) consists of 6 elements. Storage abstraction layers where we have 3 levels - physical, logical and application which describe layers responsible for physically storing the data, what data structures and access methods are used and processing data encoded from events. Events are consisting of 2 groups - external and internal that effect the system. Application activity functions - creating, modifying and deleting data with some probability distribution. Memory device reliability, here we have 4 types RAM, SSDs, Tape drives (not that common) and HDDs, basically all devices that contain memory and each of the types has it's own reliability function. Storage management functions software and firmware mapping application storage to physical storage. Environment - configuration that contains current system configuration, physical environment that affects the system and operational environment which is external and focuses on usage patterns, attacks. [7]

#### **3** Analytical model

We have to define the model of existence of data in the fog system. The model is set to determine the devices we need to examine, to get the optimal evidence needed, additionally, it serves as an estimation of the probability that the evidence exists. In this chapter we will take a look how we can calculate this probability and how we can further optimize it. [7]

# 3.1 Probability of the existence of data

To calculate the probability of the data still existing in the fog system, we have to take into account the probability the data ever existed and the probability that it still exists. We define  $k_{data}$  as the set of all the nodes that contain the data,  $P(k_i \in k_{data})$  the probability that the node  $k_i$  contains the data and  $P_{k_i}(t_{acquisition} < t_{vol})$  the probability that the node  $k_i$  contains the data at the the time of acquisition  $t_{acquisition}$ . The probability is then

 $P(k_i \in k_{data}, t_{acquisition} < t_{vol}) = P(k_i \in k_{data}) \cdot P_{k_i}(t_{acq} < t_{vol}).$ 

Now we can solve each of the two sub problems, the problem of data volatility in the nodes and the problem of a specific node processing data. Due to the nature of investigative process, not all data is of interest, only the data that can be considered evidence in the procedure, so we are looking for a specific subset of the data.

#### 3.2 Data volatility

Data volatility is a term defined as the probability distribution of the data lifetime and the centroid of this distribution given by the expected lifetime of the data. It is calculated as the sum of two variables, first one being the time taken from the moment data is created in a storage location, to the moment when data is unlinked from the storage (because it is not needed anymore). The second variable here is the time taken from the moment the data is unlinked from storage to the moment when it is overwritten by some other data, so to the moment when data is actually gone from the storage. Both variables are modeled by negative exponential distribution with regard to parameter  $\lambda$ , also regarded as the static failure rate. This parameter is tailored to each type node and message individually, since nodes can behave differently based on the purpose they have (they can be a source, a target or a proxy relaying traffic. We define the two variables as

$$P(t_{x} = t_{unlink,erase}) = \lambda e^{-\lambda t_{x}} =$$

$$= \frac{exp(\frac{-t_{x}}{t_{unlink,erase}})}{t_{unlink,erase}}$$

$$P(t_{x} > t_{unlink,erase}) = \int_{0}^{t_{x}} \lambda e^{-\lambda t} dt.$$

#### 3.3 Data distribution

When modeling data volatility, we first have to identify the nodes, since some of them are more like to contain the data we are looking for for a longer time then others. These kind of nodes are the ones that are generating and consuming data. Additionally, nodes that are acting as processing nodes have longer volatility of the data generated and it is vital to identify server nodes first in the investigation. When a node that is sending data needs to find the node to receive it, that node is selected based on the set of requirements of processing the data, denoted as *S*. This set can be partially or fully ordered and the size of this set |S| gives us the number of server nodes required. [8]

Sensor nodes mostly reside at the edges of a tree network or spread across mesh network and can be either small or quite powerful, connected to some external source. They are the source of data from the environment they are in, which is useful to find since distribution of sources and targets is what distribution of data in fog systems is conditional to. Now for the target nodes, these are usually fog service nodes processing data or devices working based on input. The two nodes that can work as target nodes are processing nodes and actuating nodes. The first one can, apart from processing data, also send data to a new target, if the input says so, while the second one physically affects the environment, if the data received commands it. Similarly to sensor nodes, target nodes can be either small and contained or more powerful ones, affecting their environment.[2]

Between the source and target nodes, there lie the proxy nodes. These nodes are just relaying the data from the source to the target, however, they are able to contain some of the data, if it is not encrypted. If nodes are overloaded, neighboring nodes can shoulder some of the burden and get the job done in the original time. The dynamics of the system can make the search for where the data is stored quite hard. The state of the nodes at the time the data is transferred is hard to find out, making the model stochastic in nature.

One of the possible node placements is one fog node, nested between central cloud node and the edge nodes. This only depends on source nodes physical location and the possible evidence should be located between gateways administering the source code, therefore, there are only a limited amount of data locations. This does become tricky to find, if the device has been moved from different locations and/or networks. The second possibility is in a complete mesh network, where every node can communicate with every other node in vicinity. The reality is, that most fog systems lie somewhere between these two options, with a layer between the edge nodes and the cloud. Usually the cloud layer has overview of all the fog nodes and resembles a hierarchical structure.

For our model, we use messages' latency between system endpoints and memory constraints for the weighting and constraints of the model to optimize service placement, but the objective functions can also use other attributes to optimize this.

The number of possible evidence locations increases with the number of possible fog nodes that can process the data from the source node. First, we are interested in the probability of a node being the source location. This is equal to the probability that any node that is able to process the data is equally likely to process the data from the sensor node. We define *S* as the set of all nodes that are able to undergo the processing task

$$S = \{\bigcup_{i} (x_i) : x_i \in X \quad \operatorname{Res}(x_i, a) \ge \operatorname{Res}_{min}(a)\}.$$

We define X as the set of all nodes and  $x_i$  as the i - th node in X. The functions *Res* and *Res<sub>min</sub>* are functions that return the node's resources and the minimal resources needed for the node to be fog service with regard to application *a*, respectively. We model x as a random variable, modeled by uniform distribution  $(P_{uniform}(x) = 1/|S|ifx \in S; otherwise0)$ . However, this is just a simplified model and to get a real model, we have to take into account how the network's attributes affect the placement of services. To generalize the probability, we have to define a couple of additional terms.

Firstly,  $g(x_i)$  is a graph measure, measuring the probability of node  $x_i$  to be selected for data processing and  $w(g(x_i))$  is a weighting function for graph measure. We will define the measure and function later on, for now we can define the general probability as

$$P_{general}(x) = \frac{w(g(x))}{\sum_{j} w(g(x_j))}.$$

This probability holds for  $x \in S$ , but if  $x \notin S$ , also the probability is 0. Commenting on the general probability, if the weighting is uniform for all the nodes, general probability is equal to the uniform probability. In the next chapters, we shall model the weighting function with respect to different approaches.[7, 8]

#### 3.4 Distance based weighting

One of the possible bases for weighting is calculating based on the shortest path. We weigh a node based on the distance between the data source and the target node. We denote d(n, m) as the shortest distance between nodes n and m, the source node is denoted as s. Additionally, we denote T the set of all target nodes that consume information from source node. We define the weighting function as

$$w_n = \frac{1}{d(s,n) + \sum_{t \in T} d(n,t)}$$

The distance measure in this case is usually number of edges. This method is quite fast and easy to implement, however it is not the most accurate.

#### 3.5 Path weighting

The second possible basis for weighting is path based, more precisely, based on the time taken to send a message. We know that the order of nodes selected is dependent on the requirements of the application. For the path weighting approach, we will use the inverse of the time taken for a small probe packet to go from source sensor to the target sensor. This consists of a couple of steps and the total time taken is just the sum of all the intermediate steps. First, the packet goes from a source sensor, then it is sent and processed by server and at the end it is re transmitted to the target. We denote  $\{e_{sn}\}$  and  $\{e_{ni}\}$  as the set of edges that the data passes thru between the source node and fog node, then from the fog node to target nodes, respectively. Here, Td will be the set of all nodes. The bandwidth of *i*-th edge will be noted as  $b_i$ ,  $s_{sn}$ ,  $s_{ni}$  are the size of the message of data sent from source to fog node and from fog to target nodes, respectively, and  $V_{path}$  is the set of nodes in each path. Trouting,v is the time used by a node for routing data, and  $t_{proc}$  is the time the fog node use for processing the data and send out new data for the sink node. Summing the times together, we get the equation for  $t_{lat}$ , which is also our distance measure in this case

$$t_{lat} = t_{proc} + \sum_{i \in \{e_{sn}\}} b_i s_{sn} + \sum_{v \in V_{path}} t_{routing,v} + \sum_{j \in T} \sum_{i \in \{e_{nj}\}} b_i s_{nj}.$$

Taking the inverse of the total time, we get our path weight

$$w_{lat} = \frac{1}{t_{lat}}.$$

#### 3.6 Path based internode weighting

In the case that fog layer is distributed among the nodes, we know that there is a central device or service knowing latency requirements and possible distribution of nodes. Taking into account latency should be minimized, there is a path or paths for which latency requirements are met and probability of a node laying on them is high. We have to find the correct path, however, even if the system knows the minimal requirements, we probably don't know them. We still want to determine the probability of a node containing the data, which we can simplistically model by the number of non-cyclic, latency-weighted paths through the node. Our weight here will be the sum of the weighted paths going thru the node (from *S*), divided by all the possible paths from source to target, giving the probability that a certain node has processed the data. Here we have to take into account that higher latency paths have lower probability of being selected. We denote by  $P_n$  set of all paths that contain the node n,  $t_{lat,p}$  is the total time of the path p and k(t) will be the scaling function of suppression based on latency. Our weighting function is then

$$W_n = \sum_{p \in P_n} k(t_{lt,p}),$$

and the weighted ratio of paths is

$$R_n = \frac{W_n}{\sum_{x \in N} W_x}.$$

Consecutively, for a fog node n and source node s,  $R_n$  is the total weight of the latency of the paths thru node n and we can calculate the probability of the node containing the data

$$P(S=n) = \frac{R_n}{\sum_{i \in N} R_i}$$

#### 3.7 Optimization of internode weighting

Path based internode weighting method is the most precise method, but it is also computationally the hardest. This can be improved upon by neglecting paths with low weights, since they have low probability to be the correct ones. Additionally, longer paths have necessarily lower weights, since adding additional node to the path always increases the weight. A cut off method was created, consisting of two parts. First part ignored all the paths with m (or in the case of the article, 5) edges more then the shortest path and the second part ignored all the paths of 110% of the shortest path. Using this method, computational complexity has been decreased, however, there was a greater chance to miss longer paths with high weights.

#### 4 Implementation

#### 4.1 YAFS

In this section we are going to discuss **YAFS**. The abbreviation stands for **YET ANOTHER FOG SIMULATOR**. The whole simulator was implemented in python. From the literature we choose three types of networks based on structure:

4.1.1 Complete Mesh Network is a topology where each node (device) in the network is connected to every other node. This creates a fully interconnected network where data can travel directly between any two nodes without having to pass through intermediate devices. On positive side it is redundant, since paths can be redirected if one node fails. It is also easy to add new nodes. The downside to this is that with each new node number of connections increases and also it has higher overhead in terms of bandwidth and processing resources. [8]

4.1.2 Network Overlay with a Tree Structure This is the type of network that is ordered hierarchicaly from the top cloud node and fog nodes in between to the bottom edge nodes equiped with IoT devices. On the positive side it is easier to manage and maintain the network compared to a mesh network. Data typically follows predefined paths, reducing the need for complex routing algorithms. On the downside failure of a higher-level node can affect all nodes beneath it, leading to potential connectivity issues. Adding new nodes may require reconfiguring the entire tree structure, which can be cumbersome. Such network is represented in the figure 1 [8].

4.1.3 Fog Colony-Based Overlay streamlines fog computing by centralizing workload management with dedicated fog colony controller nodes. These nodes efficiently distribute tasks within the colony, optimizing resource usage and enhancing system scalability and performance. However, this centralized approach may introduce potential single points of failure and require careful management of controller nodes, adding complexity to system administration [8].

Evidence in the fog - Triage in fog computing systems



Figure 2: Fog Colony-Based Overlay [8]

#### 4.2 Implementation

The first step in the simulator is that each node receives randomly assigned resource amount sampled from negative exponential distribution.

Sensor nodes were chosen from those with minimal resources, while fog server nodes were selected based on surpassing a specified threshold. Then actuator nodes were allocated from the remaining nodes to avoid simultaneous task assignments, with connection attributes randomly assigned for bandwidth and propagation delay, and weights determined by packet transmission time.

The role of processing nodes is to process the data and it will store them for longer eventually. The implementation of YAFS is based on two assumptions:

- in the dependent nodes the volatility is high and they will be overwritten very soon,
- it is also very probable that same holds for data producing nodes.

From this we conclude that processing nodes will be the most likely to hold the data.

We know that we will need some routing protocols in out networks, nut this is a problem when it comes to the mashed network. Considerate protocols are IPV4, RPL and so one, but in this work we will use Dijkstras shortest path algorithm. The path will be weighted and the weight will consist of:

- combined propagation delay given by the size of the message sent,
- the bandwidth of each of the edges,
- and the propagnition delay.

Take note that all the values are sampled from gamma distribution. When we speak about routing in hierarchical model it is quite obvious that the thing will be much simpler. We first create the minimum spanning tree. To weight the edges we use the latency of small probe. Structure of hierarchy implies the obvious - the root node will be the processing one and the leaves are the devices with a lot of fog nodes between. In the third model, a fog colony routing is implemented with special algorithm where the most central nodes tend to be the controlling ones. Those nodes submit the tasks to others. When the controller nodes have been selected, the rest of the nodes are assigned to colonies by their closest fog colony controller node. The processing nodes do not have connections between the colonies so only the controlling ones are able to coordinate between the colonies.

#### 4.3 Volatility implementation

The volatility is defined as the expected time before the data is inaccessible. The volatility can be viewed as a stochastic process, where there is a probability distribution for the time until data becomes inaccessible to the investigators. This time consists of two parts:

- the time between data being written and being freed by application,
- and the time from data being unlink and getting overwritten.

In implementation we added the new volatility extension. This is a class with two functions for setting and sampling from the distributions that representing the time from creation to deletion and time from deletion to overwritting. As we mentioned earlier those two periods represent the volatility. The simulator subtracts the unlink sample time from the message creation time for message sources, assuming message deletion upon transmission. Timestamps for creation, unlinking, and erasure are logged along with time deltas between these events and the node, application, and message creation. Volatility statistics are then calculated from this data using the Stats class.

# 5 Results

We know that the nodes that actually use or process the data will have longer time between being accepted and being marked as deleted so we can conclude that they have lower volatility. In this section we will explain the experiments that have been made for each of the node centrality measurements that we explained previously. The chance of node being a server node depends on the path length so we can compare our new measures with the existent. All of the experiments have been done with **YAFS** simulator. Each experiment ranked the measures based on their values, and the rank of the service node was noted. Measures consistently ranked near the top are more likely to result in nodes being selected as server nodes.

# 5.1 Benchmark graphs:

The first benchmark graph was **Barabasi-Albert** graph. This is a simple graph model, that uses the concept of preferential attachment to create the scale-free network. Such graph can be used in the various types of experiments. The second is **Random Lobster generator** that is a type of tree-like graph where nodes are arranged in a linear fashion, resembling the shape of a lobster's body, hence the name.

The **Erdős-Rényi** model is a classic random graph model named after mathematicians Paul Erdős and Alfréd Rényi. In this model, graph is generated by starting with a fixed number of vertices and then randomly adding edges between pairs of vertices with a certain probability p.

#### 5.2 Compairson of measures:

The first experiment represents the comparison between internode weigthing and the standard centrality measures:

- Degree centrality,
- Betweenness centrality,
- Closeness centrality,
- Eigenvector centrality,
- Harmonic centrality.

The experiments were performed with a Barabasi-Albert graph with several network sizes, varying between 50 and 1000 nodes, and an attachment parameter of 2. For all these experiments, the probability of selecting a node as a fog service node was set to 0.5.

Two variations of the internode weighting were conducted: one involving the inclusion of end nodes in the ranking alongside other nodes, and another where end nodes were excluded from the ranking altogether. The second approach offers a more accurate ranking assumption, assuming that end nodes do not serve as server nodes. In the graph bellow we can spot that for smaller networks internode weigthing is giving similar results as the casual centrality measurements but for bigger graphs outperforms them.



Figure 3: Comparison of the centrality rank of the fog server node for the set of measures for different network sizes [8]



Figure 4: Comparison of the aggregated rank of all graph types of size 250 together with individual graphs for each network type [8]

# 5.3 Sensitivity for architectures and network types

The measurements results can vary with respect to the network type. We used three network types in the experiments:

- Barabasi-albert,
- Erdos-Renyi,
- Random lobster graph generator.

In figure 4. we can see that performance was roughly equal for all of them, with different settings. In figure 5. we can spot that there is no significant difference in performance on all three architectures we implemented. Those four are mesh network, network overlay with tree structure and fog colony. All architectures have been created using Brabasi-albert. For the colonies there was 10 colonies used in the experiment.

#### 5.4 Experiments on dynamic networks

Here we are going to explain experiments based on dynamic networks. In such networks there are two sources of dynamics:

- changing the weight of the edges by changing the propagation delay and bandwidth,
- removing and adding edges with new attributes.

The dynamic attributes of the network are governed by specific parameters: a probability of 0.1 for attribute change and the use of the same parameters as the original graph for the gamma variate distribution controlling propagation delay and bandwidth. The experiment's outcomes, depicted in Figure 5, indicate a decline in performance as the network undergoes changes throughout the simulation. Number and robustness of servers that are connected into the networks will also tell how the measurements would behave if there was lower amount of nodes in the network or if the existent weights would increase.



Figure 5: The effect of dynamics on the rank of server node for the measures [8]

#### 6 Conclusion

The article studied fog computing and its effects on discovery of digital evidence. Analyzing different models of probability computations, the optimal model was found and its further optimization explained, given the constraints of the weighting. However, the measure is quite robust given different attributes and probability constrictions on nodes and different systems. The final model has been a bit optimized, but leaves the desire for further optimization, which can lose us valuable paths and consecutively evidence, therefore, further work on the topic is much desired.

#### 7 Citations and Bibliographies

#### References

- F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop* on Mobile cloud computing, ser. SIGCOMM '12. ACM, Aug. 2012. [Online]. Available: http://dx.doi.org/10.1145/2342509.2342513
- S. H. and N. V., "A review on fog computing: Architecture, fog with iot, algorithms and research challenges," *ICT Express*, vol. 7, no. 2, p. 162–176, Jun. 2021. [Online]. Available: http://dx.doi.org/10.1016/j.icte.2021.05.004
   K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe,
- [3] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe, "Mobile fog: a programming model for large-scale applications on the internet of things," in *Proceedings of the second ACM SIGCOMM workshop on Mobile cloud computing*, ser. SIGCOMM'13. ACM, Aug. 2013. [Online]. Available: http://dx.doi.org/10.1145/2491266.2491270
- [4] I. Lera, C. Guerrero, and C. Juiz, "Availability-aware service placement policy in fog computing based on graph partitions," *IEEE Internet of Things Journal*, vol. 6, no. 2, p. 3641–3651, Apr. 2019. [Online]. Available: http://dx.doi.org/10.1109/JIOT.2018.2889511
- [5] O. Škarlat, M. Nardelli, S. Schulte, M. Borkowski, and P. Leitner, "Optimized iot service placement in the fog," *Service Oriented Computing and Applications*, vol. 11, no. 4, p. 427–443, Oct. 2017. [Online]. Available: http://dx.doi.org/10.1007/s11761-017-0219-8
- [6] J. Chow, B. Pfaff, T. Garfinkel, and M. Rosenblum, "Shredding your garbage: Reducing data lifetime through secure deallocation," *Proceedings of the 14th USENIX* Security Symposium, pp. 22–22, 07 2005.
- [7] J.-P. Sandvik, K. Franke, and A. Årnes, Towards a Generic Approach of Quantifying Evidence Volatility in Resource Constrained Devices. Springer International Publishing, Dec. 2020, p. 21–45. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-60425-7\_2
- [8] J.-P. Sandvik, K. Franke, H. Abie, and A. Årnes, "Evidence in the fog triage in fog computing systems," *Forensic Science International: Digital Investigation*, vol. 44, p. 301506, Mar. 2023. [Online]. Available: http://dx.doi.org/10.1016/j.fsidi.2023.301506