Univerza v Ljubljani

Fakulteta za računalništvo in informatiko

University of Ljubljana

Faculty of Computer and Information Science

Zbornik

Digitalna forenzika

Seminarske naloge, 2022/2023

Ljubljana, 2023

Zbornik

Digitalna forenzika, Seminarske naloge 2022/2023

Editors: Andrej Brodnik, Tom Fiette, Jon Selič

Template author: David Klemenc

Ljubljana : Univerza v Ljubljani, Fakulteta za računalništvo in informatiko 2023.

Kazalo / Contents

1 Uvod / Introduction

Digital forensics is a branch of forensic science that covers the recovery and investigation of material found in digital devices and is often associated with computer crime. The term digital forensics was originally used as a synonym for computer forensics, but has expanded to include the investigation of all devices, which can store digital data. The roots can be traced to the personal computer revolution of late the seventies and early eighties. In the 1990s, the development of the discipline took place without real organization until national guidelines emerged in the 21st century.

Digital forensic investigations have different tasks. The most common are to support or refute the hypothesis before criminal or civil courts. Criminal cases involve an alleged violation of the laws it establishes legislation such as murder, theft and assault on the person. Civil cases deal with the protection of rights and property of individuals (often related to family disputes), but they can also deal with contractual ones disputes between economic entities, in which a form of digital forensics, which is called electronic, is involved discovery.

The collection contains the seminar papers of master's students at the Faculty of Computer Science and Information, University of Ljubljana 2022/2023. Within the Digital Forensics course, each group of students chose one article, which served as a starting point for the seminar work. This year's articles were selected from four research areas: Forensics of communication systems, mobile forensics, AI in forensics and miscellaneous; i.e. articles that do not fall into any of the aforementioned categories. In the following section, we have compiled summaries of each of the research papers written in this school year, while the papers themselves are compiled at the end in the order corresponding to their theme.

Ljubljana, 2023 Tom Fiette

2 Povzetki / Summaries

2.1 Forensic Investigation of Instant Messaging Services on Linux OS: Discord and Slack as Case Studies

Instant messaging applications have experienced significant user base growth and have become widely used around the globe in the last decade. Due to the low barrier of entry, these applications have attracted criminals that use them to conduct criminal activity. In this paper, we present the work conducted to investigate the forensic potential of instant messaging applications and closely look at the forensic investigation of volatile memory of Discord and Slack messaging applications. It was shown that login information, phone numbers, chat messages, attachments, reactions and app related data could be acquired from the volatile memory and used as forensic evidence.

Ključne besede / Keywords : Memory Forensics, Instant messaging applications, Discord, Slack

2.2 Retrieving Deleted Records From Telegram

Mobile applications utilize their own mechanism of storing data on a local mobile device. One interesting instant messaging platform that provides a mobile client is Telegram. In this paper, we focus on some of the research done on the matter of how Telegram encodes and stores data. We present results of a forensic analysis made on two devices that used the application.

Ključne besede / Keywords : Mobile forensics, Telegram, SQLite, Write-Ahead log

2.3 Forensic investigation of Google Meet for memory and browser artifacts

Since the outbreak of the COVID19 pandemic, video conferencing applications experienced an abrupt boom in their usage due to the new work-from-home norm. Video conferencing was used not only for work or study purposes but it also allowed to connect people that were not able to meet physically. However, with the rise of these technologies, numerous malicious agents were attracted, searching for security vulnerabilities. In this paper, we will focus on digital forensic analysis of a web-based video conferencing application - Google Meet. We will explore the approaches introduced in the reviewed paper [2]. These include experiments targeted on how various browsers and Random Access Memory (RAM) sizes of client devices affect the format and informational value of extracted memory artifacts. The aim is to explore, whether user data can be extracted from Google Meet artifacts and used as potentional digital evidence in criminal investigations.

Ključne besede / Keywords : Digital Forensics, Memory Forensics, Video Conferencing, Google Meet

2.4 Oddaljeno upravljanje in sledenje mobilnih naprav

Izguba ali kraja mobilnih naprav lahko predstavlja veliko tveganje za osebno in/ali organizacijsko varnost, saj so lahko občutljive informacije shranjene v napravi, ogrožene. Z vidika mobilne forenzike imajo te metode pomembno vlogo ne le pri zaščiti občutljivih informacij, ampak tudi pri pridobivanju dokazov v primeru kazenske preiskave. Oddaljeno brisanje zagotavlja varnejšo metodo brisanja podatkov, vendar lahko forenzičnim preiskovalcem tudi ovira zmožnost obnovitve pomembnih podatkov. Lokalizacija na daljavo omogoča takojšnje sledenje lokaciji naprave, kar lahko preiskovalcem pomaga pri prepoznavanju morebitnih osumljencev ali določanju zadnje znane lokacije naprave. Ta članek se osredotoča na raziskovanje in razlago področja sledenja in brisanja mobilnih naprav na daljavo ter bo sledil ideji izvirnega postopka in drugih mobilnih naprav.

Ključne besede / Keywords : Oddaljeno sledenje, oddaljeno brisanje, UWB, mobilne naprave

2.5 Forenzična analiza mobilnih aplikacij za vzdrževanje avtomobilov

S pomočjo modernih tehnologij v avtomobilih se mobilne aplikacije za sledenje vožnji razvijajo vse bolj napredno. Te aplikacije zagotavljajo ključne podatke o vožnji, kot so hitrost vožnje, pozicija, zdravje avtomobila in zaviranje, ki so lahko uporabljeni v digitalnih preiskavah. Kljub temu se s pojavom teh podatkov poveča količina neuporabnih podatkov, kar lahko vpliva na procesirni čas. Clanek poudarja pomembnost uporabe definiranega okvira za razlikovanje stopenj podatkov, vključno z opisom korakov, ki so potrebni za izluščenje uporabnih podatkov. Avtorji primerjajo različne mobilne aplikacije za sledenje vožnji avtomobila ter jih med seboj primerjajo po razširnosti.

Ključne besede / Keywords : Forenzika vožnje avtomobila, mobilne aplikacije, forenzična triaža

2.6 Using deep learning to detect social media 'trolls': A review

In this paper, our main goal is to provide a comprehensive summary of a recent article that showcases the effectiveness of deep learning techniques in detecting trolls based on text content. Additionally, We explore similar articles in the existing literature, examining various approaches and their efficacy in identifying trolls and toxicity in both textual and visual content on social media platforms. Moreover, We discuss the crucial correlation between social media content and criminal activities, underscoring the significance of digital forensics in the analysis of social media data. By delving into this topic, We aim to shed light on the broader implications and consequences of troll behavior and toxic content within online communities. Furthermore, We propose several enhancements to the implementation of deep learning techniques. These include incorporating computer vision to analyze visual elements, examining the impact of custom emotes, utilizing topic sampling for a more comprehensive analysis, and emphasizing the importance of regularly updating the dataset. Overall, this paper provides a detailed exploration of the current state of detecting trolls and toxic content on social media platforms using deep learning techniques. Through an extensive review of related literature and a primary focus on summarizing a recent article, We aim to contribute to the understanding of this field and highlight potential avenues for future research.

Ključne besede / Keywords : Deep learning, Trolls, Social media

2.7 Deepfake Noise Investigation and Detection

In recent years, Deepfake technology has been rapidly improving, becoming more nuanced and harder to detect for the naked eye. This imposes challenge for Deepfake detection models: to stay up-to-date and refined in order to stay effective. Authors of this paper introduce new state-of-the-art Deepfake detection model that relies finding altered forensic noise traces.

Ključne besede / Keywords : Deepfake, Neural network, Forensics, Noise traces, Face, Siamese model

2.8 Razložljiva Umetna Inteligenca za Digitno forenziko: Proti ublažitvi nezaupanja v Digitalno forenziko, ki temelji na Umetni inteligenci z intepretabilnimi metodami

Ta članek raziskuje pomembnost razložljivosti pri analizi forenzike na podlagi umetne inteligence ter preučuje potencial razložljivih modelov pri reševanju skrbi glede zaupanja in transparentnosti. Ker digitalna forenzika postaja vse bolj odvisna od tehnik umetne inteligence, nejasnost teh algoritmov povzroča dvome o njihovi zanesljivosti. Z vključitvijo razložljivih modelov preiskovalci pridobijo vpogled v odločitvene procese sistemov, kar povečuje zaupanje in sprejemanje dokazov, ki jih generira umetna inteligenca. Clanek poudarja koristi razložljivih modelov pri zmanjševanju pristranosti, zagotavljanju pravičnosti ter mostu med naprednimi tehnikami umetne inteligence in človeško razumljivostjo, kar končno povečuje zaupanje v analizo digitalne forenzike.

Ključne besede / Keywords : Umetna inteligenca, razložljiva digitalna forenzika AI, zmanjševanje nezaupanja, analiza na podlagi umetne inteligence, razložljivi modeli.

2.9 Analysis of Alt-Right Social Media Platforms

This paper examines the increasing popularity of alt-right platforms from two angles: a forensic analysis of the digital artifacts they produce and an exploration of their sociopolitical impacts. Previous investigations uncovered that significant private and secure information can be extracted from these platforms. Vulnerabilities concerning user privacy, exposing opportunities for unauthorized data access were identified. In addition, we review the ASNAAT tool designed to aggregate this data for digital forensic investigations. The study further explores the broader societal consequences of these alt-right platforms. We delve into their influence on political discourse. The challenges of regulating these platforms, balancing free speech with the prevention of harm, are also discussed. The paper concludes with the understanding that these platforms, while promoting diverse discourse, also pose potential risks to societal cohesion and democratic processes. While also lacking secure communication and data transfer protocols.

Ključne besede / Keywords : Forensic analysis, social media, disinformation

3 - Forenzika komunikacijskih sistemov / Communication Systems Forensics

Forensic Investigation of Instant Messaging Services on Linux OS: Discord and Slack as Case Studies

Mihael Trajbarič
University of Ljubljana, Faculty of Mathematics and Physics
Jadranska ulica 19
Ljubljana, Slovenia
mt3714@student.uni-lj.si

Nikolay Vasilev
University of Ljubljana, Faculty of Computer and Information Science
Večna pot 113
Ljubljana, Slovenia
nv7834@student.uni-lj.si

Jan Krivec
University of Ljubljana, Faculty of Computer and Information Science
Večna pot 113
Ljubljana, Slovenia
jk1637@student.uni-lj.si

ABSTRACT

Instant messaging applications have experienced significant user base growth and have become widely used around the globe in the last decade. Due to the low barrier of entry, these applications have attracted criminals that use them to conduct criminal activity. In this paper, we present the work conducted to investigate the forensic potential of instant messaging applications and closely look at the forensic investigation of volatile memory of Discord and Slack messaging applications. It was shown that login information, phone numbers, chat messages, attachments, reactions and app related data could be acquired from the volatile memory and used as forensic evidence.

Keywords

Memory Forensics, Instant messaging applications, Discord, Slack

1. INTRODUCTION

Instant messaging applications (IMAs) have been around for a long time, and while their usage was fairly limited at the start, they have since expanded their capabilities to support direct messaging, group messaging, dedicated chat rooms, the ability to upload files, integrating bots, embedded gifs, voice messaging and direct voice-over IP communication. The IMAs have gained popularity in all areas of human life and have inadvertently become an area for criminal activity to take place. This means, that IMAs have become an area that can provide crucial investigatory data for forensic investigators.

To uncover important data, many different works have researched the possibility of searching both volatile and non-volatile data of different IMAs across multiple operating systems and have shown that forensic investigation of IMAs can prove helpful during crime investigation. This work will focus on the IMAs Discord and Slack, which have only recenlty gained popularity.

Discord is an application that has been traditionally defined as an application to support communication around gaming, but has since broadened its consumer base to sports, entertainment, education and even work related purposes. As of 2021, Discord boasted 140 million active users and the increased adoption of the software has also attracted hate groups and criminal enterprises. To try and combat the illegal use, Discord noted that they had removed thousands of servers including those that were related to "child harm material, cybercrime, doxxing, exploitative content, and extremist or violent content".

On the other hand Slack tried to position itself as a more professional, work oriented application, designed for use in traditional businesses. Its main focus has been to gain market share against its main competitor, Microsoft Teams. As of 2019, Slack has 12 million active users and noted that they banned accounts due to illegal activity on the platform.

Both Discord and Slack are built using the Electron framework that enables developers to remove the knowledge required to build native applications and instead build the applications using web technologies like HTML, CSS, and JS. As such, both applications store its data in the volatile memory of the device, which is used to access the application.

This paper presents a forensic examination of the volatile memory of both Discord and Slack using snapshots of the memory taken after IMA-specific activities. The results show that sensitive application information such as usernames, emails, passwords, messages, conversations, and files can be acquired from system, which may be useful for forensic investigators. There is a difference in authorization needed to access stored files between Slack and Discord. Discord uses file URLs that can be accessed without authentication, while Slack requires authentication, specific to the workspace the file was uploaded to.

2. RELATED WORK

IMAs have become a subject of forensic investigations through numerous studies, which have been conducted to investigate the forensic potential of IMAs on different operating systems, including both their volatile and non-volatile data. These applications are volatile in nature, and the investigation must be done as soon as possible. The system must not be rebooted or turned off before investigation, and the tools used must be as small as possible to reduce the probability of data compromise.

Most of the research has been done on one of the following operating systems: *1)* iOS/Android, *2)* Windows, *3)* Mac, *4)* Linux. However, the primary operating systems used for investigating IMAs in forensic examinations is Android.

Salamh et al. [11] utilized Magnet ACQUIRE to generate a memory dump from a rooted Android device, in order to try to recover deleted data from instant messaging applications. In order to perform the analysis, the authors used data carving and extraction tools HxD, WinHex, and Autopsy. They successfully located an SQLite file called Write-Ahead-Log (WAL), which is created by WeChat before it transmits data remotely. This paper is considered as a valuable contribution to the hard-disk forensic investigation of IMAs, since the WAL file contains the latest queries, which allow the recovery of deleted messages or conversations using the 'delete for everyone' feature.

Anglano et al. [2] presented a forensic analysis of the ChatSecure instant messaging application on Android smartphones, which adopts strong encryption for transmitted and locally stored data. The authors used various forensic tools and techniques such as Android Emulator, Android Device Bridge, LiME, Volatility, to extract and analyze digital artifacts from the application, including message logs, metadata, and encryption keys. The findings showed that ChatSecure provides a high level of security and privacy for its users, but also reveals potential vulnerabilities in the application's security model. The most interesting discoveries are the possibility of identifying and extracting the passphrase from the volatile memory of the device and the fact that the data stored in the database cannot be recovered after being deleted, because of a secure deletion technique by SQLCipher.

Wu et al. [14] presented a forensic analysis of the WeChat instant messaging application on six different Android smartphones. The authors used various forensic tools and techniques such as Apktool, dex2jar, JD-GUI to extract and analyze digital artifacts from the application, focusing on its security and privacy mechanisms. They were able to extract the user data by utilizing the backup command of the Android Debug Bridge (ADB), even though typically root privileges are needed to access the WeChat directories. This way the study revealed potential vulnerabilities in WeChat's security model, which could be useful for forensic investigators looking to identify and mitigate security risks, including the use of weak encryption algorithms and the storage of sensitive user information in plain text. Overall, this study provided useful insights into the forensic investigation of instant messaging applications on mobile devices, particularly those used for sensitive communications. Its findings could be relevant to the forensic investigation of similar applications on Linux operating systems as well.

Sgaras et al. [12] did a forensic analysis of several instant messaging and Voice over IP (VoIP) applications, including WhatsApp, Skype, Viber and Tango on iOS and Android platforms. The focus of their forensic analysis was data, saved by the apps on the file system. The analysis was done in a simulated environment in a post-mortem fashion with the assumption that devices with simulated traffic had been previously seized. Using logical extraction, they were able to obtain traffic and content data from Skype and WhatsApp only on Android, while this did not work on iOS. However installation, traffic and content data were obtained for all apps using filesystem extraction analysis on iOS. Even richer data was obtained using manual filesystem analysis with various success among different applications.

Dezfouli et al. [7] investigated volatile memory, internal memory as well as network traffic artefacts, produced by using Facebook, Twitter, LinkedIn and Google+ on Android and iOS platforms. Usernames, account info as well as posts with metadata, messages and comments could be recovered from RAM or internal memory from both Android and iOS devices, while no passwords could be retrieved. Since all apps use TLS protocol for network communicating, only IP address, domain name and session timestamps could be retrieved using network forensic tools.

Chu et al. [5] were investigating Skype and MSN messaging between a Windows computer and an Android smartphone. Memory dump images were taken after every message from a forensic workstation, connected to the device over a USB cable using an Android Software development kit. Images were later analyzed with string search using Access Data FTK tool. Using this technique, researchers were able to find traces of communication even after logging out of IMAs and rebooting the phone.

Nisioti et al. [10] showed a volatile memory is an important source of IMA forensic data on Android devices. Although Android has Out-Of-Memory based task killer, which starts killing processes, when memory is nearly full, it usually keeps apps in RAM as long as possible. IMAs are even loaded on every boot to shorten response time, which makes volatile memory full of data. Researchers used UNIX strings tool as well as Volatility framework[1], a swiss army tool of memory forensics and Volatilitux, which has better support for Android. Analysis was done by known data identification and decoding, deriving patterns, which are then used to create regular expressions. Using this technique, researchers investigated WhatsApp, Viber and Facebook Messenger messages. The results showed messages stay in the RAM for a long time, even after a reboot of the device and removal of the battery. They were even able to recover older messages, up to 16 months old.

Of course, there are also papers, which investigated other operation systems as well as multiple operation systems. Choi et al. [4] demonstrated a forensic analysis of how the chat data from IMAs is stored and encrypted. The authors fo-

[1]https://www.volatilityfoundation.org/

cused on three popular IMAs, which are running on Windows OS and are most popularly used in China and South Korea: Kakaotalk, Nateon, and QQ Messenger. With the usage of tools CheatEngine and Process Monitor, they discovered that in the case of KakaoTalk and NateOn it is possible to retrieve user's chat messages without requiring any confidential data (e.g. secret, password). When it comes to QQ Messenger, they found out that the service provider generates the encryption key for each client application and transmits it to them. This means that users' personal data can be accessed by the service provider at any time, which by itself raises other privacy concerns.

Bashir et al. [3] did a forensic investigation of LinkedIn's desktop application on the Windows 10 operating system. The authors created fake data for their users and obtained the memory dump by using DumpIt. Using the forensic tools WinHex, HexEditor and manual analysis techniques, they were able to locate various artifacts from the application, such as system logs, registry entries, and memory dump files.

Kazim et al. [8] focused on analyzing volatile memory dump images and tried to reconstruct Google Hangout chat messages with the goal of proving communication between the suspect of online harassment and its victims. Memory dump was obtained using *dumpIT* and *Volatility* tools. A keyword search was performed using words, that usually appear in messaging apps (Skype, Hangout, MSN Messenger, email accounts etc.). Since messages are not stored in volatile memory in their natural order, they need to be manually ordered. Researchers were also able to obtain some other data, including a clipboard, where they found copied passwords and traces of opening FTP channels between victims and attacker.

Yang et al. [15] did a forensic analysis of Facebook and Skype in virtual machines running Windows 8.1 OS. After simulated communication between victims and attackers, evidence was acquired. The researchers used Wireshark to obtain a network capture file, FTK Imager to create a copy of the virtual disk and dd to make a bit-stream image of virtual memory. A wide variety of forensic tools was then used for evidence analysis, including Autopsy, HxD and Volatility. Memory dumps did not contain user's passwords but contained plenty of useful information, downloaded files, parts of databases and rich metadata in JSON format from the use of Facebook IMA, as well as payload headers from the use of Skype, all of which could be used as proofs of contact.

Thantilage and Khac[13] created a framework for retrieval of IMA evidence from *volatile memory* (RAM), running Windows OS and Mac OS. While the usual approach is to use a tool for creating memory dump, this technique has some issues. The proposed framework consists of running application and passing known data (strings) through it. After that a memory acquisition is made, using one of the tools for memory dumping. After known data is identified and extracted from memory dump, patterns and similarities can be drawn, which lead to obtaining other data as well using REGEX expressions. The framework was able to obtain a great variety of data across a range of applications, including Facebook, Twitter, Viber, Whatsapp, Gmail and others.

Motylinski et al. [9] is the closest to Davis et al. [6]. It presented the tool DiscFor on the Linux operating system, which automates the evidence-collection process and presentation of Discord data from local directories. The authors revealed that it is possible to retrieve recent messages with precision and speed from all Discord sources without having to access the individual's account, but by utilizing tools that can decode the data, which is stored in the cache storage (disk cache on Windows and simple cache on Linux), as well as an activity log.

Overall, the above studies provide valuable insights into the forensic investigation of instant messaging applications on different operating systems. However, as we can see, there is a lack of research investigating these applications on Linux OS. Therefore, the current study investigates the forensic potential of Discord and Slack on Linux, contributing to the existing body of knowledge on the topic.

3. FORENSIC INVESTIGATION OF DISCORD AND SLACK

In this section, we will present the study on volatile memory forensics of Discord and Slack instant messaging applications.

3.1 Volatile memory forensics

To better understand the works in the original paper [6], we must first establish what the forensics of volatile memory is. Volatile memory (also known as RAM) is a form of computer memory that can be read and changed in any order, typically used to store working data and machine code. It works faster than other types of storage (such as hard disks, CDs, DVDs etc.) but the data (or the state) is lost when power is removed from the system. The beginnings of memory forensics reach back to the year 2005 when DFRWS issued a Memory Analysis Forensics Challenge [1]. As a response to this challenge, many tools designed to analyze memory dumps were created and have since then become irreplaceable tools in a digital forensics arsenal. These tools had knowledge of the operating system's internal data structures and were thus capable of reconstructing the operating system's process list and process information. This proved that operating system-level volatile memory forensics is possible. Later, many forensics tools were developed to offer insight into volatile memory. One such tool is Volatility, also used in the original paper.

Because memory forensics is time sensitive, either a live system analysis (capture and preserve the volatile data in RAM of the running device before the system is shut down) or a cold boot attack (if the system was only recently shut down, we can reboot the system and preserve the data stored in RAM before it dissipates completely) can be performed. Some operating systems use memory paging to extend the capacity of RAM and certain pages can be retrieved from the secondary storage.

3.2 Forensic examination method

First, the authors of the paper created three Gmail accounts, which they used to create corresponding Slack and Discord accounts. The second step was setting up three virtual machines, which were running on the Ubuntu 20.04.3 LTS

operating system, using VMware Workstation. Each machine had a RAM constraint of 4GB (4096MB) to reduce the amount of data to analyze in the memory dump. Each machine was also assigned one of the user accounts, which means that the Discord and Slack applications were installed and direct messaging and server interactions between them took place.

During this process a total of 54 snapshots were taken using Volatility 2.7 to dump the process memory for the applications. They created a Volatility profile for the Linux version they were using, which helped them run the command `linux_pslist` to collect the process ids for Discord and Slack. Take in mind that the process ids remained the same for the duration of the tests, since Discord and Slack generate new process ids only when application is restarted. They then took snapshots using a modified `linux_dump_map` command. The command was also modified, in order to get rid of large empty spaces consisting of only zeros due to `zread()` call within the `linux_dump_map` returning an all zero buffer when a page is missing in the memory. We can see the the order of snapshots taken in Table 1 for Discord and in Table 2 for Slack:

Snapshot	Machine
Base Snapshot	All
Install	All
Login	All
Profile image, status update	All
Friend request	Machine 1
Friend request accepted	Machine 2, 3
Direct messaging	Machine 1, 2
Created server	Machine 1
Joined server	All
Assigned Roles	All
Added role to private channel	All
Joined voice chat	All
Quit discord	All

Table 1: Memory snapshots taken in regards to Discord

Snapshot	Machine
Install Snapshot	All
Install	All
Login & create workspace	Machine 1
Login & join workspace	Machine 1, 2
Added Giphy bot to workspace	Machine 1
Updated profile	All
Workspace chat message	All
Private chat (machine 1, 2)	Machine 1,2
Machine 3 added to private chat	All

Table 2: Memory snapshots taken in regards to Slack

They performed a string search for keywords like *Discord, discordapp, gmail, txt, username, password, Slack* and names of users in the files generated by the `linux_dump_map`, using `grep` to generate a list of relevant files. This list was further filtered using `strings` command with a limit of n-4 (n being the length of the shortest keyword) to remove files that didn't contain useful information. Because some snapshots were taken, when the applications weren't running (For example the snapshot after quitting both applications), the `WxHexeditor` was used to inspect the snapshots and search for keywords.

3.3 Implementation

In order to validate the results from the paper, we tried to recreate the experiment following the instructions. We created three Gmail accounts (identities), one for each virtual machine, and their corresponding profiles in Discord and Slack.

1. Machine1 represents the identity with the following accounts:

   Email: victorchapmandf123@gmail.com

   Discord: VictorChapmanDF123#3057

   Slack: Victor Chapman

2. Machine2 represents the identity with the following accounts:

   Email: erinbullockdf123@gmail.com

   Discord: ErinBullockDF123#4638

   Slack: Erin Bullock

3. Machine3 represents the identity with the following accounts:

   Email: scarlettboltondf123@gmail.com

   Discord: ScarlettBoltonDF123#6876

   Slack: Scarlett Bolton

As part of the preparation, we also provided the files that would be used for the testing. Since Machine1 and Machine2 are supposed to upload files, we prepared text documents and .jpg images presenting each of the Harry Potter books. After setting up the three virtual machines running on the operating system Ubuntu 20.04.3 LTS, we uploaded the needed files into the machines, downloaded Discord and Slack and started generating the needed snapshots.

We generated all 54 snapshots, the same way they did in the paper, and the experiment progressed smoothly until this point. When we tried to dump the process memory using Volatility 2.7, an unexpected challenge arose. Unfortunately, the process memory dumping procedure encountered difficulties, hindering further progress in the experiment. The possible reasons behind this issue could be:

- Process integrity protection – Discord's and Slack's current version might employ additional security measures or process integrity protections that interrupt the memory dumping process. These protective mechanisms could include code signing, process memory encryption, or anti-tampering measures, which may prevent direct access to the process memory.
- Version incompatibility – While the paper may have indicated successful use of Volatility 2.7, it is possible that subsequent updates or changes to Discord, Slack, or even Python, may have made them incompatible with the version of Volatility being used.

- Anti-forensic techniques – Discord and Slack, being popular communication platforms, may have incorporated anti-forensic techniques to prevent unauthorized access or tampering with user data. These techniques could include obfuscation, memory anti-forensic methods, or countermeasures to prevent memory extraction, making the memory dumping process more challenging or infeasible.
- Outdated or incompatible libraries: Volatility 2.7 relies on various external libraries to perform its forensic analysis tasks. Over time, these libraries may undergo updates or changes that could result in compatibility issues with this version of Volatility. It is possible that some of the implemented libraries used by Volatility in the article may no longer be compatible or fully functional with the current version of Volatility or the target Linux OS.

Due to time constraints, we weren't able to fully investigate or address the reason behind the occurrence of this problem. Nonetheless, we believe that the obtained results from the paper provide us with valuable insight into the forensic investigation of instant messaging services on Linux OS. This is why we will put our focus on the evaluation and analysis of the results, which were already presented to us in the paper.

3.4 Findings

The authors found a lot of data in the memory snapshots, such as usernames, user roles, uploaded textual files, images, messages between users, passwords used for login etc. The summary of artefacts obtained during the investigation can be found in the table 3 below:

Artifact	Discord	Slack
Avatar Image	x	x*
Discord Server	x	N/A
Email	x	x
Emojis	x	x
Friend Request	x	N/A
Gifs	x	x
Image	x	x*
Message	x	x
Password	x	-
Restricted Channels	-	-
Status	x	x
Slack Workspace	N/A	x
Text File	x	x*
User Roles	x	N/A
Username	x	x
Voice Chat	-	N/A

Table 3: Artifacts found in Discord and Slack.

Note that the "x*" indicates that the artifact is found but further authentication is needed to access it (Slack requires workspace authentication to access uploaded files URL's). The "-" indicates that the artifact may exist, but there may be limitations to its recovery. The N/A indicates that the artifact is not applicable to the specific platform.

It is expected to find sensitive information, such as passwords stored in the volatile memory, so finding the password for Discord was not unexpected. The login information was found in snapshots other than those created after the login, indicating that Discord stores the login information after the initial login (Figure 1). However, all snapshots were taken without restarting the app and it is possible that we would find a login token instead of a password after a restart. Slack uses third party for account generation and authentication (gmail authentication was used in this case), but no reference to a gmail password nor oauth token was found in the snapshots (Figure 2).

Messages within restricted channels were visible only to users with required privileges to access the data, indicating that restricted data does not get automatically pushed to all users and privacy is respected.

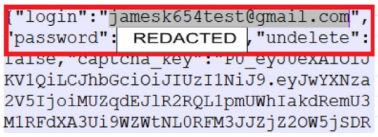


Figure 1: Snapshot right after Discord login process. Researchers searched for full @gmail address. Username, id, avatar id, channel ids, the phone number associated with the account, status, region, and geolocation were found as well as JSON with username and password.

Figure 2: Snapshot right after Slack profile-update process. Researchers were able to obtain a reference to uploaded image, which was not protected and was accessible. They also found custom status in JSON format.

4. CONCLUSIONS

As the instant messaging applications (IMAs) became an essential tool for day-to-day communication for individuals and businesses, they have also attracted criminals that used the benefits of the applications to conduct criminal activity and act as a hidden communication channel. This paper presents the numerous studies conducted to investigate the forensic potential of different types of IMAs. Because different applications differ in their architecture and abilities they can provide to the users, many different tools and approaches are used to conduct forensic investigations. The original paper has focused on forensic investigation of the volatile memory and has shown that sensitive and forensically significant data can be reconstructed from the memory snapshots after using the Discord and Slack messaging applications. It was shown that login information, phone

numbers, chat messages, attachments, reactions, bot ids, and app related data could be acquired from the volatile memory and used as forensic evidence. Differences between Discord and Slack were presented, mainly the lack of security restriction of files uploaded to Discord.

5. REFERENCES

[1] Dfrws 2005 forensics challenge.

[2] Anglano, C., Canonico, M., and Guazzone, M. Forensic analysis of the chatsecure instant messaging application on android smartphones. *Digital Investigation 19* (2016), 44–59.

[3] Bashir, S., Abbas, H., Shafqat, N., Iqbal, W., and Saleem, K. Forensic analysis of linkedin's desktop application on windows 10 os. In *16th International Conference on Information Technology-New Generations (ITNG 2019)* (Cham, 2019), S. Latifi, Ed., Springer International Publishing, pp. 57–62.

[4] Choi, J., Yu, J., Hyun, S., and Kim, H. Digital forensic analysis of encrypted database files in instant messaging applications on windows operating systems: Case study with kakaotalk, nateon and qq messenger. *Digital Investigation 28* (2019), S50–S59.

[5] Chu, H.-C., Lo, C.-H., and Chao, H.-C. The disclosure of an android smartphone's digital footprint respecting the instant messaging utilizing skype and msn. *Electronic Commerce Research 13*, 3 (Sep 2013), 399–410.

[6] Davis, M., McInnes, B., and Ahmed, I. Forensic investigation of instant messaging services on linux os: Discord and slack as case studies. *Forensic Science International: Digital Investigation 42* (2022), 301401. Proceedings of the Twenty-Second Annual DFRWS USA.

[7] Dezfouli, F. N., Dehghantanha, A., Eterovic-Soric, B., and Choo, K.-K. R. Investigating social networking applications on smartphones detecting facebook, twitter, linkedin and google+ artefacts on android and ios platforms. *Australian Journal of Forensic Sciences 48*, 4 (2016), 469–488.

[8] Kazim, A., Almaeeni, F., Ali, S. A., Iqbal, F., and Al-Hussaeni, K. Memory forensics: Recovering chat messages and encryption master key. In *2019 10th International Conference on Information and Communication Systems (ICICS)* (2019), pp. 58–64.

[9] Motyliński, M., MacDermott, A., Iqbal, F., Hussain, M., and Aleem, S. Digital forensic acquisition and analysis of discord applications. In *2020 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI)* (2020), pp. 1–7.

[10] Nisioti, A., Mylonas, A., Katos, V., Yoo, P. D., and Chryssanthou, A. You can run but you cannot hide from memory: Extracting im evidence of android apps. In *2017 IEEE Symposium on Computers and Communications (ISCC)* (2017), pp. 457–464.

[11] Salamh, F. E., Karabiyik, U., and Rogers, M. K. Asynchronous forensic investigative approach to recover deleted data from instant messaging applications. *2020 International Symposium on Networks, Computers and Communications (ISNCC)* (2020), 1–6.

[12] Sgaras, C., Kechadi, M.-T., and Le-Khac, N.-A. Forensics acquisition and analysis of instant messaging and voip applications. In *Computational Forensics* (Cham, 2015), U. Garain and F. Shafait, Eds., Springer International Publishing, pp. 188–199.

[13] Thantilage, R. D., and Le Khac, N. A. Framework for the retrieval of social media and instant messaging evidence from volatile memory. In *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)* (2019), pp. 476–482.

[14] Wu, S., Zhang, Y., Wang, X., Xiong, X., and Du, L. Forensic analysis of wechat on android smartphones. *Digital Investigation 21* (01 2017).

[15] Yang, T. Y., Dehghantanha, A., Choo, K.-K. R., and Muda, Z. Windows instant messaging app forensics: Facebook and skype as case studies. *PLOS ONE 11*, 3 (03 2016), 1–29.

Retrieving deleted records from Telegram

Žan Počkar
Faculty of Computer and Information Science
Večna pot 113
Ljubljana, Slovenia
zp68409@student.uni-lj.si

Tom Sojer
Faculty of Computer and Information Science
Večna pot 113
Ljubljana, Slovenia
ts22842@student.uni-lj.si

ABSTRACT
Mobile applications utilize their own mechanism of storing data on a local mobile device. One interesting instant messaging platform that provides a mobile client is Telegram. In this paper, we focus on some of the research done on the matter of how Telegram encodes and stores data. We present results of a forensic analysis made on two devices that used the application.

Keywords
Mobile forensics, Telegram, SQLite, Write-Ahead log

1. INTRODUCTION
Telegram is a cloud based messaging service. It was first released for Android in 2010 and by now works on every major operating system. By far the most usage represents Android, as 85% of all the users use this OS [2]. It supports different communication options, such as so-called normal chats, secret chats, video calling, chat rooms, etc. Normal chats do not support end-to-end encryption, but instead use an internal protocol MTProto that encrypts data on Telegram's servers. For end-to-end encryption, the secret chat method is mostly used.

As claimed by Telegram in June of 2022 [1], they were one of the top 5 downloaded apps of that year with more than 700 million monthly users. This is why research is crucially needed to catch any potential flaw that could be exploited in a malicious manner.

In our article, the following will be presented: the background section (2), where we will go into details of Telegram Messenger and how the app stores data, then we will present some related work (section 3). The follows a review of how the authors of the discussed article prepared for the investigation (subsection 4.1) and analysis (subsection 4.2). We will show forensic tools useful for research (section 5) and discuss methods utilized in the reference article (section 6).

2. BACKGROUND
2.1 Database
To understand the behavior of the app's storage, we need to go into details about how it handles the database. The examined application version was 7.9.3, released in August, 2021. Telegram Messenger client stores data with SQLite. This database's method of storing is using a Write-Ahead log file (WAL) with a checkpoint of 1000 pages (a database in SQLite is divided into pages, each of size at least 512 bytes). Using the WAL file is a method of ensuring that data doesn't get lost. Before anything is written to a database, it is first logged. This ensures that any transaction can be repeated if some failure appears. The checkpoint of 1000 pages notes that all data before it reaches that point has been written to disk. In case of a crash, the recovery procedure would include finding the checkpoint and recovering anything until reaching the checkpoint. Meanwhile the main database is left intact and changed after that point has been reached. This means that recently deleted data can be found in the WAL file, but not in the main database file.

2.2 Telegram Data Structure
While Telegram stores some local data in clear text, such as the name of the sender/recipient, more volatile data is stored into complex data structures that appear in a binary serialized form. Fields in the structure are stored as a sequence of bytes, where they appear in specific positions. Retrieving information from such a structure demands first to deserialize it and then decode each useful field. The main problem is that only for a limited set of such structures, the structure and serialization scheme is known. The exact decoding scheme is not made public by the brand, which is why the process of figuring out the correct structure is left to the community. The structures are constantly redefined or removed after new features are released, which makes it harder to maintain tools that decode all the different variations.
Since a part of the application is open source, Anglano et al. [4] performed a source code analysis to identify all the different data structures used by the app as well as to reconstruct the structure and serialization schemes. The cited article named these structures *Telegram Data Structures* or *TDSs*. We can see mappings from the analysis that connect some TDSs with a Java class definition in Figure 1. We will take a look at TDSs later in the analysis.

TDS	Java class
Base types (e.g., int and string)	org.telegram.tgnet.NativeByteBuffer
Chat	org.telegram.tgnet.TLRPC.Chat
documentAttributeAudio	org.telegram.tgnet.TLRPC.TL_documentAttributeAudio
documentAttributeFileName	org.telegram.tgnet.TLRPC.TL_documentAttributeFilename
documentAttributeVideo	org.telegram.tgnet.TLRPC.TL_documentAttributeVideo
EncryptedChat	org.telegram.tgnet.TLRPC.EncryptedChat
FileLocation	org.telegram.tgnet.TLRPC.FileLocation
GeoPoint	org.telegram.tgnet.TLRPC.GeoPoint
Message	org.telegram.tgnet.TLRPC.Message
messageActionPhoneCall	org.telegram.tgnet.TLRPC.TL_messageActionPhoneCall
MessageMedia	org.telegram.tgnet.TLRPC.MessageMedia
messageMediaContact	org.telegram.tgnet.TLRPC.TL_messageMediaContact
messageMediaDocument	org.telegram.tgnet.TLRPC.TL_messageMediaDocument
messageMediaGeo	org.telegram.tgnet.TLRPC.TL_messageMediaGeo
messageMediaPhoto	org.telegram.tgnet.TLRPC.TL_messageMediaPhoto
messageMediaVenue	org.telegram.tgnet.TLRPC.TL_messageMediaVenue
Peer	org.telegram.tgnet.TLRPC.Peer
PhotoSize	org.telegram.tgnet.TLRPC.PhotoSize
User	org.telegram.tgnet.TLRPC.User
UserProfilePhoto	org.telegram.tgnet.TLRPC.UserProfilePhoto

Figure 1: Mapping between TDSs and Java classes from the source code.

3. RELATED WORK

Although many different tools and solutions were used in this paper, individual use cases, justifications for the choices of tools and procedural workflows were not documented in detail. The related work section helps to explain some of those uncertainties as it lists works that depict guidelines, techniques, protocols, and standardized procedures in the field of mobile forensics, records retrieval, and data carving. They also credit works explaining (then relevant) Telegram's folder and database structure and the possible forensic analysis approaches, as well as works on similar messaging applications.

Some articles have already dealt with TDSs on other platforms. One instance is a paper, written by Gregoio et al. [5], which deals with and investigates the app on a Windows phone. The importance of forensic analysis has also been introduced on other messaging platforms such as WhatsApp [3] and WeChat [8].

As it was mentioned in the paper, with each version of Telegram released, the structure of its database, data structures, objects and, in general, the way data is handled and processed by the app, change as a result of added or removed features. This means that most, if not all, of the previously designed protocols and tools will have reduced success rates of analyzing the data. There has however been some previous research on the SQLite structures for data carving [6].

Despite the app's rapidly changing internal structure, older tools, documentation and works remain relevant due to the cross validation of the results which has proven itself to be essential. Using a plethora of tools and various versions of the same tools could yield significantly better results.

4. METHODOLOGY

4.1 Preparation and collection

To prepare for an examination and analysis and get a sense of how deleted records are handled by the application using different forensic tools, an investigative environment was first established. Two Android phones were examined: an LG G6 with Android 9 and Samsung A50 with Android 11. Both phones had a SIM card activated and Telegram Messenger client installed. Root access was gained on both devices in order to collect data used by the app. All data on the two phones that was unrelated to the observed app was deleted. This allowed the researchers to focus on relevant data only.

After finishing with the experimental setup, messages were exchanged between the clients. The preparation for the analysis included the exchange of media files, such as images and videos. All of the media were later deleted both from the app and the trash bin folder. In total, around 2200 messages were exchanged during this process [7].

The research focused on retrieving data in different scenarios. Criteria included (1) elapsed time since the device had been acquired, (2) whether the device is powered on, off or in airplane mode and the state of the application (3), i.e. if messages are created or deleted. As many different conditions were tested, several images of the devices were produced. Conditions are stated in Figure 2, each state is denoted by status, time frame and action made on the device. For instance, in one of the images acquired, the phone was powered off and the researchers gained data from that state. By comparing different criteria, some potential real world scenarios were simulated.

Status	Time Frame	Action
Standby	As soon as possible	None
Standby	After 2/6/24 h	None
Standby	As soon as possible	Removing Sim
Airplane mode	After 2/6/24 h	None
After reboot	24 h	Removed Sim
After shutdown	48 h	None
Faraday box	After 2/6/24 h	None

Figure 2: Experiments with different device statuses.

4.2 Examination and analysis

As part of the forensic analysis, some open source and commercial forensic tools were used. This includes software for SQLite analysis and image acquisition software. Figure 4 shows a list of all the tools used. We will discuss other tools later in the article.

There are a few forensically relevant data sources on the device. Firstly, the main local database, named cache4.db. We can see the database's path on the device in Figure 3. In one of the scenarios, the size of cache4.db was 1336 kB, while the related WAL file had the size of 4475 kB. This is a consequence of using the Write-Ahead log. The local database included 52 tables, containing user data, messages, contacts and phone numbers. There is also a user configuration file that stores details of the account on the device, profile photos of user's contacts and also copies of files the user interacted with. The latter is stored in the media folder. From the normal chat messages table, several table entries could

be read in clear text, such as the name of sender/receiver, date sent, the message status and direction. Info and body entries of this table were in the form of the Telegram Data Structure, which as previously defined, is in a binary form. Using forensic tools, some data was queried and the initial conclusions were:

- The same message may appear in the WAL file several times.
- The Auto-delete feature of Telegram, that removes messages after a certain period, did not affect the ability to recover messages.
- The body of messages is in a TDS form and not easily interpreted.

Content	Directory/ File Name
Telegram Database	/data/org.telegram.messenger/files/**cache4.db**
Copies of exchanged picture-media files in normal chat	/media/0/Telegram/Telegram Images
Copies of outgoing picture-media files in normal chat	/media/0/Pictures/Telegram
Picture-media files and their thumbnails	/media/0/Android/data/org.telegram.messenger/cache

Figure 3: Location of the main database file and the media files.

Experiments concluded that while different states of the device result in the cache4.db and WAL file changes, they do not alter the data inside of the messages table. For instance, when a phone has a network connection, both files might be altered after a short amount of time. They will remain in the same state if a phone is in airplane mode or the SIM card is removed. While cache4.db and the WAL file change very frequently, the messages tables inside doesn't. The researchers then observed events when the messages table was modified, and this happened when:

- a message was created, even if not successfully sent
- a message was received
- a message was opened
- a message was deleted

Figure 5 shows how a hex editor was used to retrieve information. Now let's describe some other observations. Firstly, there was more storage needed to record an event when an outgoing message appeared compared to an incoming message.
Secondly, the main database file does not have the auto vacuum feature enabled which keeps the file size at the minimum. This potentially means that some of the deleted records might be retrieved. This was tested by the examiners on both devices. They deleted hundreds of records and attempted to recover them. Immediately after deletion, most of the messages were recoverable, but after new messages were created, all the deleted messages were not recoverable anymore. This was verified in the database file and the WAL file using a hex editor. Records are also not recoverable after the user has logged out.

Here, we can also make an observation with the WAL file. Since the file makes a commit to the database after a 1000 pages have been reached, a lot of data can be restored for a while if the commit doesn't occur for a longer period of time. In this sense, deleted records may be retrievable for a while, but here an element of luck is involved.
After deletion of media files that were used in any message exchange, some artifacts were still available on both the devices. This was observed by first deleting pictures and by comparing their artifacts with the records in cache4.db.

Lastly, we can ask ourselves and observe when records are not available anymore. This is the case on two occasions:

- when a user logs out and the database is deleted
- when the local database is deleted in the settings menu

5. TOOLS

Tools used for this research can be split into forensic and non-forensic tools. Non-forensic tools were used mainly for preparing the white box environment for the experiment, screen capturing and monitoring purposes. On the other hand, forensic tools were used mainly for extraction and analysis of both devices and Telegram app data.

5.1 Non-forensic tools

Among the non-forensic tools used for setting up the devices Magisk Manager was used for rooting and Team Win Recovery Project - TWRP for creating a custom recovery menu. In addition to the third-party solutions, an inbuilt android feature was used for creating additional user on each of the two devices. Dual apps or Dual messenger are features of contemporary android devices which enable creating a virtual copy of an app that can then be used with a separate account as an independent user with their own file system, databases and permissions.

5.2 Forensic tools

A total of 26 tools were used, of which 13 were commercial use tools, 9 either open source or free and 4 were commercial tools with some form of a free version. As it was previously mentioned, the version of the tool used often significantly impacted the quality of the obtained results. Which is why in addition to exploring different software solutions, a combined use of multiple versions of the same tools was utilized.

All of the tools used were categorized according to their use case into forensic analysis, SQLite analysis and image acquisition tools. But not all tools were equally successful. Due to the rapid changes to the Telegram Data Structure, most tools were not up to date with the latest changes and could not decode any data. Some tools were successful, but not in all cases, some could successfully decode only the normal chat messages while some worked only with the secret messages. As the changes in the TDS are unlikely to backtrack, newer versions of forensic tools had higher success rates as they were more compatible with the latest TDS encoding. Examples of the above mentioned cases are Cellebrite UFED Physical Analyzer 7.50, Oxygen Forensics 14.1, and AXIOM v5.7 and v6.0.

Usage	Tool	License
Forensics Analysis	Access Data (Forensic ToolKit-FTK) 7.4	Commercial
	Autopsy Suite 4.19.1	Open Source
	Belkasoft X 1.10	Commercial
	Cellebrite UFED Physical Analyzer 7.50	Commercial
	Magnet Axiom 5.7, 6.0	Commercial
	MSAB (XRY and XAMN)	Commercial
	Oxygen forensics (Oxygen Detective) 14.1	Commercial
	slo-sleuth/android blob cache (https://github.com/slo-sleuth/android_blob_cache.) (media files)	Open Source
	X-Ways Forensics 20.1 (& HEX analysis)	Commercial
SQLite Analysis	aLEAPP 1.9.9	Open Source
	Andriller 3.6.1	Open Source
	Belkasoft X 1.10	Commercial
	bring2lite (Meng and Baier, 2019)	Open Source
	DB Browser for SQLite 3.12.2	Open Source
	Forensic Toolkit for SQLite	Commercial
	FQlite 1.5.5	Open Source
	KS DB Merge Tools for SQLite	Free Tool & Commercial
	Oxygen forensics (SQLite Viewer) 14.1	Commercial
	Paraben E3 Forensic Platform 3.1	Free Tool & Commercial
	SQLECmd (https://github.com/EricZimmerman/SQLECmd.)	Open Source
	SQLite-Deleted-Records-Parser (https://github.com/mdegrazia/SQLite-Deleted-Records-Parser.)	Open Source
	SQlite Expert 5.4	Free Tool & Commercial
	SQLite Forensic Explorer 2.0.0	Free Tool & Commercial
	Teleparser (https://github.com/RealityNet/teleparser.)	Open Source
	undark 0.6 (https://github.com/alitrack/undark.)	Open Source
Image Acquisition	ADB Commands (CLI) & SDK Platform Tools 31.03	Open Source
	Belkasoft X 1.10	Commercial
	Cellebrite UFED 4 PC 7.48	Commercial
	Magnet Acquire 2.45	Free Tool
	MSAB (XRY and XAMN)	Commercial
	Oxygen forensics (Oxygen Extractor) 14.1	Commercial

Figure 4: A list of forensic tools used.

Cellebrite UFED Physical Analyzer is a commercial tool used for uncovering digital evidence, trace events, and examine digital data. It allows for import and decoding of a specific application through its selective decoding, and it speeds up many aspects of the investigation including automatized report generation. Despite all the quality of life features, in this case, it was able to decode only normal messages.

Oxygen Forensics tools used in this research are marketed as all-in-one solutions for extraction, decoding, and analysis of both data and artifacts for both computer and mobile forensic investigations. Although convenient, these tools were able to decode only the secret messages.

When it came to AXIOM, built for remote acquisitions, collection and evidence analysis from different sources including mobile devices, the older version could not decode any data while the newer could decode only normal messages.

On the other hand, tools on which the researchers had relied the most were SQLite Analysis tools. Especially tools for inspecting, querying, carving and parsing the database. Although these tools also followed the trend of newer versions yielding better results.

6. DISCUSSION

This attempt to retrieve the telegram data can't be considered as nothing but a proof of concept and even that done in extreme laboratory conditions. While the article uses some novel approaches to retrieve stored data and does achieve some success. We must raise some questions about the methodology and the fine result of retrieving the data. We can generally split our questions into two categories. The first categories deal with the methodology of the used device and acquired data. The second category deals with the final results and readability of said data.

We first must raise questions about the devices used. The team decided to use two completely wiped devices on which the first installed a custom recovery menu and gained root access. Secondly, they installed the same version of the telegram app, a version which is now discontinued. Lastly, while the team did simulate conversation with the telegram app no other application was running on the devices prior, during or after communications.

It is true that the article itself freely admits that the device setup wasn't forensically sound but more of a whitebox proof of concept. And yet even after this question arise if the test wasn't done in too many perfect conditions and if such results even carry any weight in real world practice.

Let's begin with the devices themselves. The rationale for not using other applications to reduce during the test is sound but in our opinion flawed. Multiple times the data was retrieved only because the team was able to identify changes after the messages were exchanged, in addition the article states that the data is in certain instances volatile

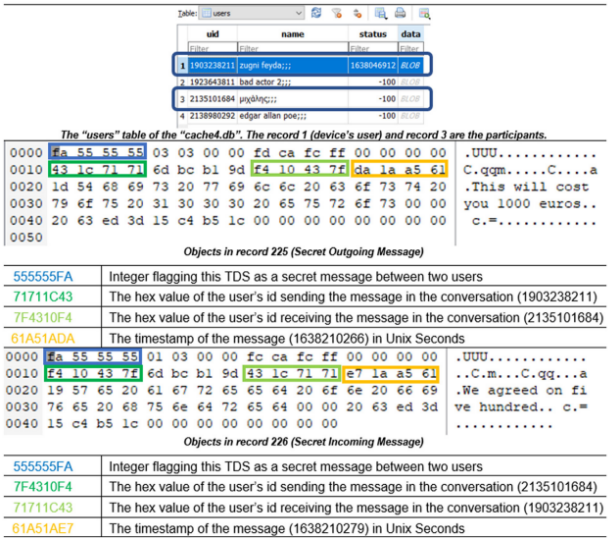


Table: users

uid	name	status	data
1 1903238211	zugni feyda;;;	1638046912	BLOB
2 1923643811	bad actor 2;;;	-100	BLOB
3 2135101684	μιχάλης;;;	-100	BLOB
4 2138980292	edgar allan poe;;;	-100	BLOB

The "users" table of the "cache4.db". The record 1 (device's user) and record 3 are the participants.

```
0000 fa 55 55 55 03 03 00 00 fd ca fc ff 00 00 00 00  .UUU............
0010 43 1c 71 71 6d bc b1 9d f4 10 43 7f da 1a a5 61  C.qqm.....C....a
0020 1d 54 68 69 73 20 77 69 6c 6c 20 63 6f 73 74 20  .This will cost
0030 79 6f 75 20 31 30 30 30 20 65 75 72 6f 73 00 00  you 1000 euros..
0040 20 63 ed 3d 15 c4 b5 1c 00 00 00 00 00 00 00 00   c.=............
0050
```

Objects in record 225 (Secret Outgoing Message)

555555FA	Integer flagging this TDS as a secret message between two users
71711C43	The hex value of the user's id sending the message in the conversation (1903238211)
7F4310F4	The hex value of the user's id receiving the message in the conversation (2135101684)
61A51ADA	The timestamp of the message (1638210266) in Unix Seconds

```
0000 fa 55 55 55 01 03 00 00 fc ca fc ff 00 00 00 00  .UUU............
0010 f4 10 43 7f 6d bc b1 9d 43 1c 71 71 e7 1a a5 61  ..C.m...C.qq...a
0020 19 57 65 20 61 67 72 65 65 64 20 6f 6e 20 66 69  .We agreed on fi
0030 76 65 20 68 75 6e 64 72 65 64 00 00 20 63 ed 3d  ve hundred.. c.=
0040 15 c4 b5 1c 00 00 00 00 00 00 00 00              ............
```

Objects in record 226 (Secret Incoming Message)

555555FA	Integer flagging this TDS as a secret message between two users
7F4310F4	The hex value of the user's id sending the message in the conversation (2135101684)
71711C43	The hex value of the user's id receiving the message in the conversation (1903238211)
61A51AE7	The timestamp of the message (1638210279) in Unix Seconds

Figure 5: The scheme of TDSs in both hex and ASCII encoding.

and can be lost if the changes occur on the operating device. It is true the team did test three different changes such as turning the device on and off that did not change the data. But the same test did show that actions inside the applications lead to such change. The question arises if the running of similar applications at the same time could lead to trigger such changes. In this vein of questioning is the decision to use just a limited amount of accounts while we understand that because of limited time and money for any such research smaller sample sizes must be used. During the research, the team has shown that a message from any source using the telegram application will modify the already stored data. That means that an average user's message would be much harder to retrieve especially if the user is an active user of the application.

While the usage of the same version of the application is ideal, this is a likely event for users which use the application regularly so we don't see much problem with this approach even if it is a little idealistic.

And lastly gaining root access in advance and installing custom recovery software before installing the telegram application is a problem in the methodology. As we can't be sure how gaining root access or installing the recovery menu, could change the data we are trying to access but this could be mitigated through a usage of multiple copies used in any forensic investigation even if it is extremely time consuming.

The other major category of the question raised is pertaining to the results themselves. They have had moderate success in retrieving different data from the telegram application, they freely admit that they have much greater success with text messages in comparison to the pictures. But the thing one must question is if the success from retrieving text data shouldn't be classified more as a partial or limited success. The problem which is already mentioned in the articles is the format of encoding used in telegram text data. While the team was on multiple occasions able to retrieve the raw data they couldn't read large or even in some cases whole portions of retrieved data as data was parsed inside the TDSs structure as seen in figure 5. One must then ask themselves if retrieved data that can't be decoded in reasonable timeframes can't even be considered retrieved or must it at most be just used as proof of communication between parties. The whole question of usability of the retrieved data does not only rests with the team conducting the research as the research has shown that most modern and available currency forensic tools aren't capable of decrypting the data. In the future, if better tools for decrypting the retrieved data are developed, such a method for retrieval may be used for more than just proof of communication.

7. CONCLUSION

We find that the article has a straight forward goal, clear methodology and a concise plan for proving its findings. In the article, the team thoroughly presents the Telegram applications, the methodology they used, their process and their results.

Conclusions regarding volatility of cache4.db database and the Write-Ahead log file along with the data collected from other articles on Telegram could be used in further investigations. In conclusion, we believe this article is an important first step at looking into retrieving the deleted records from the Telegram Messenger client. Further analysis would be needed in regards to discovering possible vulnerabilities of the application as too many tradeoffs and ideal conditions were used in the primary research for this article.

8. REFERENCES

[1] 700 million users and telegram premium. https://telegram.org/blog/700-million-and-premium. Accessed: 12-5-2023.

[2] Unsend messages, network usage, and more. https://telegram.org/blog/unsend-and-usageandroid-developers-never-sleep. Accessed: 13-5-2023.

[3] C. Anglano. Forensic analysis of whatsapp messenger on android smartphones. *Digital Investigation*, 11(3):201–213, 2014. Special Issue: Embedded Forensics.

[4] C. Anglano, M. Canonico, and M. Guazzone. Forensic analysis of telegram messenger on android smartphones. *Digital Investigation*, 23:31–49, 2017.

[5] J. Gregorio, A. Gardel, and B. Alarcos. Forensic analysis of telegram messenger for windows phone. *Digital Investigation*, 22:88–106, 2017.

[6] D. Pawlaszczyk and C. Hummert. Making the invisible visible – techniques for recovering deleted sqlite data records. *International Journal of Cyber Forensics and Advanced Threat Investigations*, 1(1-3):27–41, 2021.

[7] A. Vasilaras, D. Dosis, M. Kotsis, and P. Rizomiliotis. Retrieving deleted records from telegram. *Forensic Science International: Digital Investigation*, 43:301447, 2022.

[8] S. Wu, Y. Zhang, X. Wang, X. Xiong, and L. Du. Forensic analysis of wechat on android smartphones. *Digital Investigation*, 21:3–10, 2017.

Forensic investigation of Google Meet for memory and browser artifacts

Nino Brezac
Faculty of Computer and Information Science
Večna pot 113
Ljubljana, Slovenia
nb9762@student.uni-lj.si

Nela Petrželková
Faculty of Computer and Information Science
Večna pot 113
Ljubljana, Slovenia
np91887@student.uni-lj.si

Jakub Pokorný
Faculty of Computer and Information Science
Večna pot 113
Ljubljana, Slovenia
jp31870@student.uni-lj.si

ABSTRACT

Since the outbreak of the COVID19 pandemic, video conferencing applications experienced an abrupt boom in their usage due to the new work-from-home norm. Video conferencing was used not only for work or study purposes but it also allowed to connect people that were not able to meet physically. However, with the rise of these technologies, numerous malicious agents were attracted, searching for security vulnerabilities.

In this paper, we will focus on digital forensic analysis of a web-based video conferencing application - Google Meet. We will explore the approaches introduced in the reviewed paper [2]. These include experiments targeted on how various browsers and Random Access Memory (RAM) sizes of client devices affect the format and informational value of extracted memory artifacts. The aim is to explore, whether user data can be extracted from Google Meet artifacts and used as potentional digital evidence in criminal investigations.

Keywords

Digital Forensics, Memory Forensics, Video Conferencing, Google Meet

1. INTRODUCTION

Google Meet stands among top video conferencing applications, especially in the post COVID-19 era. Some of the others are namely *Zoom*, *Microsoft Teams* or *Cisco WebEx*, which are desktop client applications The major difference with *Google Meet* is that it is a web-based application, meaning that it can be run out-of-the-box: on different browsers and operating systems. The fact that *Google Meet* is a web application brought many security issues. Numerous ways of obstructing a video conference appeared:

- link stealing - accessing the meeting via an obtained link
- meeting bombing - joining a private meeting and/or obstructing it
- privilege exploits - elevating privileges, misusing privileges (muting and removing others from a meeting)
- malicious link sharing.

Another challenge that arose is the use of *Google Meet* data in a potential digital forensics investigation.

Web applications depend on Service Oriented Architecture (SOA), enabling dynamic web infrastructure and reflecting changing software requirements of users as stated in [1]. Moreover, they do not store client application directories on disk, which makes digital investigations harder, since it usually includes interesting forensic information. What it does store is a cache. Unfortunately, this browser cache that is stored on disk does not contain communication and meeting records. This information, however, can with patience and effort be carved from memory dumps. This is fundamental for the extraction of volatile data that are not present on a disk or may be encrypted.

The extraction and analysis of the memory artifacts was the main focus of the authors of the reviewed paper [2]. They performed several experiments with different settings of RAM sizes, types of browsers and operating systems. These were: *Google Chrome*, *Mozilla Firefox* and *Microsoft Edge* on both *Windows* and *Linux*, with different RAM sizes. Another obstacle in obtaining more data from video conferencing clients is the fact that they are not open-source, therefore there is no possibility of performing a source code analysis that would enable construction of high-level data structures. This forces digital investigators to perform reverse engineering on the structures without proprietary code.

2. PAPER OVERVIEW

In this section we will evaluate the Google Meet forensics paper [2]. *Google Meet* is used for quick meetings without downloading a desktop application. It offers three meeting scenarios:

- Create a meeting for later.
- Start an instant meeting.
- Schedule in Google Calendar.

In addition, users may invite others to meet via a link in the application. In the meet, we can chat, share our screen, caption the screen, and draw on a whiteboard, which uses another Google application called Jamboard.

What forensically valuable traces does Google Meet leave in our memory? Categorically:

- AC1 > Traces of Google Meet's usage (*.lnk* file).
- AC2 > Meeting records (meeting name, title, timestamp, user e-mail, device ID).
- AC3 > Communication Records (messages and timestamps).
- AC4 > Document/image downloads (file name, type, path etc.).
- AC5 > Correspondence (user e-mail).
- AC6 > Closed captioning transcripts.

An in-depth graphical representation is seen on figure 1.

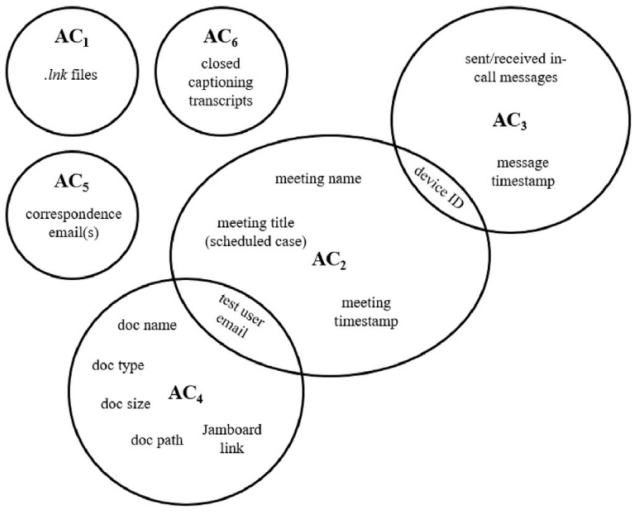


Figure 1: Google Meet forensic artifacts [2].

2.1 Memory forensics

Running processes can be extracted including execution timestamps. Every tab has its own process with a generic name and PID.

A *Google Meet* link file is present - *google meet.lnk*, where meeting names, email addresses, device ID, and the timestamp of the meeting can be extracted.

From a memory dump, it is possible to carve profile photos and favicons. We can also extract sent/received in-call messages including device ID and timestamp. Curiously, received messages are extractable for larger memory sizes.

Documents and images shared via whiteboard are also recoverable, including file name, size, extension, email address of the user, Jamboard link, and path of the stored file.

2.2 Browser forensics

Traces of usage are found on the disk in the Web Applications folders.

The *IndexedDB-levelDB* folder can reveal meeting IDs of all the previously held and joined meetings by the user without a timestamp.

Communication records can be found in the session's logs in a scattered and fragmented form. Parsing these logs is perhaps an option when capturing memory is not possible.

In case *Meet* is bookmarked, a Google Meet bookmark is saved in a browser with the timestamp, ID, and name of the bookmark.

The SQLite database is a useful tool for retrieving data related to *Google Meet*, including timestamps, visit counts, and duration of visits, as well as downloaded files' names, sizes, timestamps, and paths. Even if a suspect deletes their browser history, artifacts such as download and URL tables remain.

The cookies related to Google Meet can be extracted including names, host keys, values, creation times, expiration times, and last accessed times.

The cache folder stores profile pictures of the user and accounts the user interacted with. These profile pictures can be associated with corresponding *Google Meet* URLs. Links to Jamboard sessions and Google Calendar can also be found. Some other forensic traces include but are not limited to: meeting ID, name and description of the meeting, location, creator, attendees' email address, start and end timestamp of meeting, time zones, call UID, HTML link, meeting settings, and searches made using the browser. We have to bear in mind that, whilst this is a rich set of artifacts, all of these can be easily deleted by the user.

2.3 Case study

What does this mean in the context of a potential forensics investigation? A simple scenario shall be presented. Suppose there had been an insider leak of confidential company data. Moreover, *Bob* was found to be communicating with the insider. At this point we are trying to tie a particular suspect employee Eve, to Bob.

We begin with a memory and disk examination of Eve's PC. Whilst exploring browser traces, *Google Meet* is found as one of the services used. As shown, a record of Eve's meeting IDs can be recovered from the application's levelDB.

We follow the forensics trace and find the meeting IDs in memory dumps. After finding the meeting data that corresponds to our wanted ID, we carve out some artifacts:

- AC2 > Meeting records = meeting name, email address, device ID, timestamp
- AC3 > Communication records = sent/received in-call message, device ID, timestamp
- AC5 > Correspondence = email address

The key evidence extracted proves to be proof of correspondence i.e Bob's email. This indicates Eve had been in contact with Bob through Google Meet. However, as shown in the memory forensics section, received messages can not *always* be extracted.

This is why this information is cross-checked with an analysis of Bob's PC. Hypothetically, a match had been found and we have thus provided evidence of Eve communicating with Bob and have gotten closer to solving our case.

2.4 Takeaway

Web applications are ever more popular solutions for communication purposes. Simplicity and ease of use being key reasons for this. A caveat, however, is an increased attack surface and security challenges.

This study presented an exhaustive browser forensic analysis of *Google Meet* on different web browsers. extracting traces of usage, history, meeting data, downloads, cache, cookies, and others.

As communication and forensic traces are of significant importance in a court of law this work can be followed upon in other directions. Either with an analysis of *Google Meet* forensic artifacts on *macOS*, *Android*, *iOS*, or with a similar analysis on different communication platforms, whether of web apps such as *Discord*, or standalone desktop apps such as Zoom. A possibly interesting comparison would be, for example, comparing forensics traces of the browser version of Microsoft Teams and the desktop version (e.g. differences, how do they compare, which one reveals more).

3. RELATED WORK

But how do other apps differ? Do *Teams* or *Webex* reveal more information, is the approach similar at all or is it substantially different? This motivated us to dig deeper and go through other papers in this area of research.

3.1 Forensic Analysis of Microsoft Teams: Investigating Memory, Disk and Network

In this section we shall evaluate a similar paper [3] with a similar research cast - Z. Khalid and F. Iqbal, who evaluated Google Meet in a forensic setting, this time return with others to examine *Microsoft Teams* one of the top 3 videoconferencing applications. As such, the approach is somewhat similar - memory and disk forensic analyses lead, this time with network forensics added. *FTK* was used for memory dumps, *Wireshark* for network traffic and a set of forensic tools for disk examination - *Volatility*, *Photorec* and *Bulk Extractor* among others.

The authors opted for a test environment in the form of a Windows 10 virtual machine (VM), with a fixed amount of 4 GB RAM and 60 GB disk-space. A Microsoft Teams user account was created and signed-in. A clean environment makes analysis easier as there is no artifact mix-up with other applications or system files. Once again, typical use-cases were defined: setting up the user profile ID, searching for people in correspondence using keyword search, adding/deleting contacts, audio/videocalls and one-to-one/ group meetings etc.

3.1.1 Memory Forensics

Microsoft Teams is to start with a larger software package than *Google Meet*. It could be said that Meet's target quality is its ease-of-use for an end user, while *Microsoft Teams* strives to offer a sizeable, complete package for enterprise users.

The analysis kicks off with *Volatility*, an advanced memory forensics framework, and some of its plugins. A summary of tools and potential findings follows:

- *pslist* or *pstree* - plug-ins list all Teams processes.
- *yarascan* - another plug-in allows searching of artifacts particular to a process id (PID). It is necessary to state that these artifacts are limited in information, they require offsets to make sense of them.
- *Photorec* - allows carving of images from memory dumps. Logos and favicons, but more importantly profile photos, that are surprisingly stored in unencrypted form.
- *Bulk Extractor* - AES key extraction.
- Microsoft Teams Chat Files - store plenty of information, exchanged files, sizes, timestamps, sender usernames, e-mails, dates, IDs etc.
- plain string searches - reveal user artifacts such as nicknames, and again e-mails and IDs. Thankfully passwords are not available in plaintext.
- *skypexspaces-[UserID]db* - a database of all messages, even deleted ones.

As we can see lots can be extracted. In fact most interactions can be successfully extracted, from participant IDs and names to messages and media. What is particularly interesting is the non-encrypted profile photos and deleted messages that are also quite easily recoverable. A potential headache for end-users, but a valuable piece for forensics analysts.

3.1.2 Disk Forensics

Disks are the usual targets for forensic analyses. Unlike memory, disk-space stores information for a relatively longer time. Sadly, *Microsoft Teams*' client application folder did not reveal information of value, but the Windows Registry on the other hand, had a fair amount of forensic artifacts. We list following keys related to Teams and their explanations:

- HKCU\SOFTWARE\RegisteredApplications - List of registered applications in the client desktop (Microsoft Teams inclusive).
- HKCU\SOFTWARE\Microsoft\Office\TeamsUser - account information including email address, private meeting settings, installation source, web account ID and login information etc.
- HKCU\SOFTWARE\Microsoft\Office\Teams \Capabilities\URLAssociationsURL - associations of Microsoft Teams (e.g., sip, IM, callto etc.)
- HKCU\SOFTWARE\Microsoft\Office\Outlook\Addins \TeamsAddin.FastConnect - Microsoft Teams add-in for Outlook
- HKCU\SOFTWARE\Microsoft\UserData\ UninstallTimes - Microsoft Teams is listed if it is uninstalled.

The *TeamsUser* key is most probably the most information rich one, offering plenty of user information.

3.1.3 Network Forensics

Since memory is volatile it is not always available during an investigation. Disk-space, on the other hand, can be quite easily manipulated with. In such a case, network data is used as a reliable alternative for extracting artifacts because network traffic cannot be tampered with. Logs can of course be tampered with, but this again belongs to disk forensics.

To perform network forensic analysis of the *Microsoft Teams* application, a Wi-Fi hotspot was set-up to isolate the traffic. *Wireshark*, a known network protocol analyzer, was used to both capture and analyze the traffic. A network miner provided further insight in the *.pcap* traffic captured using *Wireshark*. The IP addresses of servers were investigated using `https://ipdata.co/?ref=iplocation`.

Netscan output offers valuable information in the form of timestamps, and other IP addresses that can be corroborated with the pslist output or packets captured using a network protocol analyzer such as *Wireshark*. All user actions (login, message, media exchange) were captured separately to be analyzed individually. It was discovered that all the network traffic of MicrosoftTeams was encrypted as no credentials, messages, or transferred image or text files were observed in the packet captures in plaintext. The encryption keys were exchanged using Elliptic Curve Diffie Hellman (ECDH) key agreement protocol, while the application data was transferred using either HTTP over TLSv1.2 or HTTP2.

The IP addresses and timestamps from the network traffic were used to reconstruct the history of communication.

3.2 Forensic Analysis of the Cisco WebEx Application

Here we evaluate another related paper - [4]. Z. Khalid and F. Iqbal return with other colleagues, and this time evaluate *Cisco Webex* in a forensic setting. The principle is the same as with the previous paper - an evaluation of memory, disk, and network forensics. A very similar experiment was conducted - with *FTK Imager* once again used for memory dumps, *Wireshark* for network traffic and a set of more-or-less same forensic tools.

3.2.1 Memory Forensics

Regarding memory artifacts, the authors stayed loyal to *Volatility* and its plugins. Once again, a summary of findings follows:

- *pslist* or *pstree* - plug-ins list all *WebEx* processes. Timestamps of executions are also present, possibly a forensic asset.
- *yarascan* - outputed with limited call information.
- *Photorec* - extracts the WebEx logo and favicons. Contraty to Teams, profile photos, are stored in encrypted form.
- AES keys were extracted, and a string search against the query tag revealed some keyword searches performed by the user.
- File names of text files exchanged were also extracted along with the size of the files.
- The communicated URLs and the deleted URLs were also recovered from the memory.
- plain string searches - reveal user artifacts such as usernames, e-mails, video addresses of users and a personal room number. Passwords are encrypted as expected.
- All communicated messages were found under the *spark MessageGroup* tag.
- Scheduled meetings were surprisingly less secure. A lot of information was found under the *WebExMeetingData* tag. Namely, passwords of the scheduled meetings in plaintext, along with the meeting key, meeting name, username, and site name of the user who scheduled the meeting.

A sizeable amount of data can be extracted from *WebEx* as well. In fact most interactions can be successfully extracted, participants, messages, URLs and timestamps. More or less, a story similar to Teams, with the main differences being: *WebEx* encrypting its profile photos and having less secure scheduled meeting data.

3.2.2 Disk Forensics

Traces of *Cisco Webex* usage is left in *Local*, *Locallow* and *Roaming folders*. The data in *Local* is largely uninteresting - temporary files, caches and configurations. The *Roaming* folder contains the bulk of interesting material. It is essentially a server copy of account data.

The *Local* folder has two subfolders: WebEx and CiscoSpark.

WebEx contains:

- .json startup scripts
- default webEx site information
- an encrypted cache
- application and plug-in binaries

The true data source however, is a database located in *\AppData\Local WebEx\\CiscoMeeting.db*. Here we can find both meeting and client data. Meeting data consists of only timestamps, whereas client data consists of a last meeting timestamp and account URL. Some other traces are present, but they are not of note (encrypted databases, log-in state of user).

3.2.3 Network Forensics

Similar to *Teams*, the authors hoped for additional value in the network traces of *WebEx* usage. A Wi-Fi hotspot was created to isolate the traffic, and *Wireshark* was used to capture the traffic in a .pcap file. Each user event, (such as log-in, chat, file sending) was analyzed independently.

The structure of a WebEx session proves to be quite complex. Logging into *WebEx*, creates a session with Amazon.com on port 443, naturally because *Cisco* uses *Amazon Web Services* and *Microsoft Azure* to provide cloud services. Another session with Cisco WebEx LLC servers (*GLOBALIDBROKEREU.WEBEX.COM*, *idbroker-eu.webex.com*, and *identityeu.webex.com*) is established on port 443 to authenticate the host to the cloud.

Some useful information, such as profile ID, contacs, user credential and files are found via a myriad of servers (*jabber-integration-k.wbx2.com*, *convk.wbx2.com*, *contacts-service-k.wbx2.com*, *avatark.wbx2.com*, *calendar-k.wbx2.com*), but sadly all of it is encrypted.

Results showed that *WebEx* encrypts more or less all information passed in the network. Each WebEx session has TLSv1.2 encryption, and all sent media is sent over encrypted UDP. The only element left "out in the open" are URLs exchanged in a session, as they are left in plaintext.

The only real forensic value from the network side of are IPs and timestamps fetched with *netstat*. This can serve as auxiliary evidence, that is, only if we can surely tie a person to an IP, can we combine it with this kind of data.

4. PRACTICAL TRIAL

The history file, `.\History`, contained some usage traces, for example, the exact time of the meeting, which we were able to extract using a simple SQL query. Some trace of usage is also present in the history file . If we had cleared the history before the meeting, we would have found nothing here and would have had to rely on the `.\Bookmark` file. In our case, we didn't bookmark the *Google Meet* client, so Bookmark contained no potential useful forensic trace related to *Google Meet*.

If we had not found anything useful in the history nor bookmarks, another option is to search for useful logs or even an .Idb file in `.\IndexedDB\https_meet.google.com_0.indexed\\db.leveldb`, for example 000005.Idb. All logs contained timestamps, as seen in figure 2.

url	last_visit_time	visit_time	Total time [s]
https://meet.google.com/	13328559883295100	13328559727122100	156,173
https://accounts.google.com/Service	13328559727122100	13328559727122100	0
https://meet.google.com/accounts/S	13328559727122100	13328559727122100	0
https://meet.google.com/?pli=1	13328559727615600	13328559727122100	0,4935
https://meet.google.com/?pli=1	13328559727615600	13328559727615600	0
https://meet.google.com/_meet/frr-	13328559733232300	13328559732540400	0,6919
https://meet.google.com/_meet/frr-	13328559733232300	13328559733232300	0
https://meet.google.com/frr-rddk-m	13328559733824300	13328559733824300	0
https://meet.google.com/	13328559883295100	13328559883295100	0

Figure 2: Extracted timestamps from the browser history.

We attempted to extract browser artifacts from our test calls on *Chrome*. The browser artifacts were found in `AppData\Local\Google\Chrome\UserData\Default`

In `.\Cache` and `.\Accounts\AvatarImages`, a profile picture can be found.

`.\LoginData` stores records of logins to Google Meet, with a timestamp and used email address.

In `.\DownloadMetadata` we found only recently downloaded files with URL links. Older downloads are not stored here. To retrieve older downloads, the most valuable method would be to search the browser history file, which seems to be a special scenario with a high chance of success.

The files in the `.\JumpListIconsRecentClosed` store only the last three closed tabs, which is a simple way to get a quick overview of recent activity. The `.\JumpListIconsMost\\Visited` folder stores five icons of the last browser session.

Files in `.\Sessions`, `.\Cookies`, `.\SafeBrowsingNetwork`, `.\SafeBrowsingCookies` folders are not as intuitive. There are mostly stored in binary format with lots of shortcuts and parameters.

5. CONCLUSIONS

Starting with disk forensics, Google Meet creates artifacts that can lead to evidence. Mostly basic traces of usage are created with the used browser in the browser history. This is stored in a format relatively easy to access - in the form of an SQL database.

Memory forensics, on the other hand, is another possibility for a more detailed view of suspects' activity. This presumes the special scenario of recently confiscated devices. While these memory snapshots can be quite rich with forensic artifacts, we have to be well aware that this is not something readily available for every case.

Lastly, conducting forensic analysis on the network itself is challenging. The network "does not lie", but is at the same time encrypted at all times, so our added value is little to none, unless we are in possession of the encryption keys.

The competitors of Google Meet paint a similar story. An overview of the research area has shown that the differences present are mostly of a functional nature, which protocol is used, and the way communication flows. In forensic terms the differences were minuscule. *Microsoft Teams* did not encrypt its profile photos, *Cisco WebEx* did, but did not secure its scheduled meeting information as much.

6. REFERENCES

[1] S. H. R. M. . B. R. Akremi, A. Applying digital forensics to service oriented architecture. 17(1):17–42, 2020.

[2] F. Iqbal, Z. Khalid, A. Marrington, B. Shah, and P. C. Hung. Forensic investigation of google meet for memory and browser artifacts. *Forensic Science International: Digital Investigation*, 43:301448, 2022.

[3] Z. Khalid, F. Iqbal, K. Al-Hussaeni, MacDermott, and M. Hussain. Forensic analysis of microsoft teams: Investigating memory, disk and network. 06 2022.

[4] Z. Khalid, F. Iqbal, F. Kamoun, M. Hussain, and L. A. Khan. Forensic analysis of the cisco webex application. In *2021 5th Cyber Security in Networking Conference (CSNet)*, pages 90–97, 2021.

4 - Mobilna forenzika / Mobile Forensics

Oddaljeno upravljanje in sledenje mobilnih naprav

Jan Adamič
Fakulteta za elektrotehniko
Tržaška cesta 25
Ljubljana, Slovenija
ja5023@student.uni-lj.si

Jon Selič
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
js4941@student.uni-lj.si

Luka Galjot
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
lg0775@student.uni-lj.si

POVZETEK

Izguba ali kraja mobilnih naprav lahko predstavlja veliko tveganje za osebno in/ali organizacijsko varnost, saj so lahko občutljive informacije shranjene v napravi, ogrožene. Z vidika mobilne forenzike imajo te metode pomembno vlogo ne le pri zaščiti občutljivih informacij, ampak tudi pri pridobivanju dokazov v primeru kazenske preiskave. Oddaljeno brisanje zagotavlja varnejšo metodo brisanja podatkov, vendar lahko forenzičnim preiskovalcem tudi ovira zmožnost obnovitve pomembnih podatkov. Lokalizacija na daljavo omogoča takojšnje sledenje lokaciji naprave, kar lahko preiskovalcem pomaga pri prepoznavanju morebitnih osumljencev ali določanju zadnje znane lokacije naprave. Ta članek se osredotoča na raziskovanje in razlago področja sledenja in brisanja mobilnih naprav na daljavo ter bo sledil ideji izvirnega postopka in drugih mobilnih naprav.

Ključne besede

Oddaljeno sledenje, Oddaljeno brisanje, UWB, Mobilne naprave

1. UVOD

V današnjem povezanem svetu so mobilne naprave postale ključni del našega življenja. Na njih se zanašamo za komuniciranje, produktivnost ter dostop do ogromnih količin osebnih in zaupnih informacij. Kljub temu pa ta priročnost prinese tudi določen del tveganja, v primeru da je naprava ukradena ali izgubljena. Tukaj postane oddaljeno upravljanje s podatki naprave ali pridobitev njene lokacije zelo pomembna storitev.

Oddaljeno upravljanje s podatki naprave in njena lokalizacija je pomemben korak v odvračanju tatvin mobilnih naprav ali pridobitev izgubljene naprave. Te zmožnosti omogočata dva največja ponudnika mobilnih storitev: Apple, z "Find My" - in Google s storitvijo "Find My Device". Storitev je pomembna tudi z vidika pravice zakona, saj shranjevanje zadnjih lokacij naprave omogoča tudi vpogled v gibanje njenega lastnika.

2. SORODNA DELA

Ob pregledu področja smo poleg osnovne predstavitve [7], ki jo je izvedel Mitch Kajzer na konferenci, našli še nekaj drugih pomembnih virov.

Glede zadev o zasebnosti tehnologij in lokacijskih podatkov na mobilnih napravah z iOS sistemom smo se obrnili na študijo [3].

O opisu UWB tehnologij obrnili na več različnih virov [10], [1]. Prvi opisuje uporabo s stališča proizvajalca Samsung mobilnih naprav, drugi je pa bolj splošen o tehnologiji in se bolj nanaša na naprave podjetja Apple.

3. PREGLED TEHNOLOGIJ

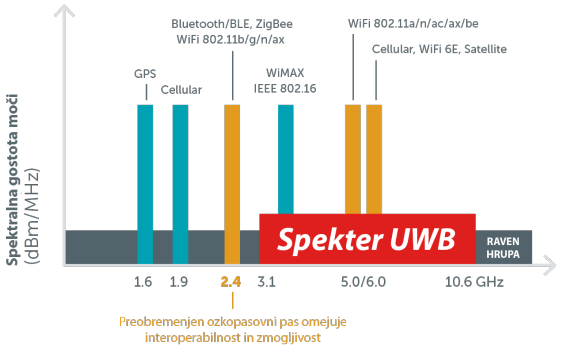
V tem razdelku obrazložimo način delovanja tehnologij in protokolov, ki dovolijo zaznavanje lokacije mobilne naprave, brisanje podatkov iz naprave ter deaktivacijo le-te. Razložimo čip U1 podjetja Apple, protokol Ultra-wideband (v nadaljevanju UWB) ter druge radijske protokole kot sta Wi-Fi in Bluetooth.

3.1 Protokol ultra wide-band

Impulzna radijska tehnologija (angl. Ultra wideband) je brezžična komunikacijska tehnologija, ki deluje z oddajanjem zelo kratkih (nekaj deset do sto pikosekund) moduliranih impulzov energije po zelo širokem frekvenčnem prostoru. Za delovanje mu je namenjen del spektra med 3,1 in 4,8 GHz ter med 6,0 in 10,6 GHz. Shema frekvečnega prostora je priakzana na sliki 1 Z uporabo pulzov v tako širokem delu spektra dosežejo, da signalov ne motijo razne ovire med napravami.

Apple, Google in Samsung so tehnologijo v svojih napravah uporabili za izboljšanje storitev AirDrop in Nearby Share. To je protokol za hipno deljene datotek med napravami. Z njo dosežejo, da se naprave med sabo prej najdejo. Poleg tega tehnologijo uporabljajo za sledenje lokacije drugih naprav. Zaradi natančnosti določanja oddaljenosti je tehnologija primerna za uporabo kot radijski ključ, ki omogoča odklepanje vrat in vstop v vozila.

Apple je svojo tehnologijo predstavil s telefonom iPhone 11, Samsung je z uporabo pričel v Galaxy S20 Note. Podporo za UWB je dodal tudi Google v svojih telefonih Pixel 6 Pro in Pixel 7 Pro. Njen namen v teh napravah je podoben Samsungovemu. Pri Applu tehnologijo uporabljajo vsi novejši telefoni, medtem ko je pri Samsungu in Googlu omejena



Slika 1: Shema uporabljenega frekvenčnega prostora povzeta po [11].

na višji cenovni razred. Samsung za delovanje tehnologije uporablja čipe podjetja NXP, Apple pa lastni čip U1.

3.2 U1 Čip

U1 je medijski čip, ki temelji na tehnologiji UWB in je namenjen za naprave podjetja Apple. Glavna funkcionalnost čipa je zmožnost natančnega zavedanja lokacije v prostoru, preko uporabe protokola UWB. S tem protokolom se meri čas potovanja signala med katerikoli napravo, ki vsebujejo čip U1 in so si dovolj blizu. To omogoča računanje točne lokacije naprave (z natančnostjo manj kot deset centimetrov). Čip vsebujejo novejše naprave podjetja Apple, kot so iPhone 11 in AirTag.

3.3 Primerjava z protokoloma Wi-fi in Bluetooth

Wi-Fi in Bluetooth sta brezžična komunikacijska protokola, ki se uporabljata za prenos podatkov. Protokol Wi-Fi se najpogosteje uporablja za prenos podatkov z visoko hitrostjo na dolge razdalje, medtem ko je Bluetooth optimiziran za manjšo porabo energije in namenjen za krajše razdalje in počasnejše hitrosti (kot so slušalke). UWB spada nekje vmes, saj se uporablja za hitre prenose med napravami na majhnih razdaljah. Je pa tudi optimiziran za nizko porabo energije. Primerjavo vseh tehnologij nas sliki 2 je pregledno pripravilo podjetje Qovro [9].

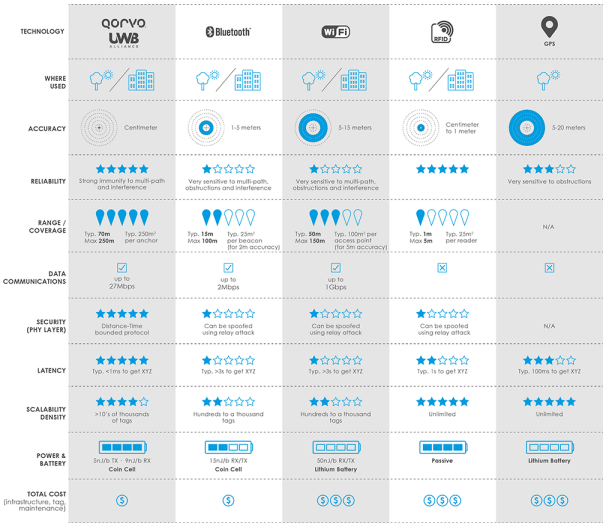
3.4 Način delovanja "Airplane Mode"

"Airplane mode" (angl. Letalski način) je značilnost vseh modernih mobilnih naprav, ki prepreči delovanje vseh brezžičnih komunikacijskih protokolov, kot so Wi-Fi, Bluetooth in mobilni podatki. Aktivacija letalskega načina povzroči, da naprava ni povezana z nobenim brezžičnim omrežjem in ne sprejema signalov.

3.5 Oddaljeno sledenje lokaciji

Moderne mobilne naprave omogočajo več načinov sledenja lokaciji. Vsaka med njimi ima svoje prednosti in slabosti, ki jih naredijo na določene načine boljše od drugih. Naprave uporabljajo GPS, mobilna omrežja, WiFi dostopne točke in druge radijske tehnologije.

Večina naprav omogoča sledenje preko GPS sistema, ki je natančen na nekaj centimetrov, vendar le pod pogojem, da



Slika 2: Pregled in primerjava trenutno uporabljenih brezžičnih tehnologij v današnji potrošniških napravah. Pridobljeno na [9].

naprava prejme njegov signal. Način zato deluje le zunaj na prostem.

Za bolj približen način lociranje naprav se upravljajo tudi mobilna omrežja in bližnje WiFi dostopne točke. Te lahko uspešno locirajo uporabnika na več sto metrov ali nekaj kilometrov. Natančnost je odvisna od gostote oddajnikov mobilnega omrežja in velikostjo ovir okoli naprave.

Kot bolj eksotično tehnologijo sledenja lokaciji naprav smo predstavili Ultra Wideband radijsko tehnologijo, ki omogoča lociranje naprav znotraj prostorov in jo nekateri imenujejo kot GPS za znotraj.

Organi kazenskega pregona imajo korist od oddaljenega sledenja naprav, ko gre za preiskovanje kaznivih dejanj in prijetje osumljencev. Ko je naprava prijavljena kot izgubljena ali ukradena, lahko podatke o sledenju delite z oblastmi, kar jim omogoči, da ustrezno ukrepajo. V primerih, ko je bila naprava ukradena, lahko organi pregona sodelujejo s ponudniki tehnologije za sledenje, da sledijo premikom naprave in potencialno identificirajo vpletene. Oddaljeno sledenje napravam se je izkazalo za ključno pri vračanju ukradenih naprav in privedbi kriminalcev pred sodišče.

Vendar je treba omeniti, da je uporaba oddaljenega sledenja naprav v kazenskem pregonu predmet zakonskih predpisov in pomislekov o zasebnosti. Najti ravnovesje med vračanjem ukradenega premoženja in spoštovanjem pravic posameznikov do zasebnosti je ključnega pomena. Zakoni in pravilniki v zvezi s sledenjem naprav na daljavo se med pristojnostmi razlikujejo in bistveno je, da organi kazenskega pregona upoštevajo uveljavljene protokole in pridobijo ustrezno pravno dovoljenje pri dostopu do podatkov o sledenju. Varovanje zasebnosti uporabnikov in zagotavljanje preglednosti pri uporabi tehnologije sledenja bi moralo vedno ostati prednostna naloga.

3.6 Oddaljeno brisanje

Oddaljeno brisanje je močno varnostno orodje, ki omogoča uporabniku zaščito osebnih in/ali zaupnih podatkov v primeru izgube, kraje ali nepooblaščenega dostopa. Ko izvedemo oddaljeno brisanje so vsi podatki na napravi izbrisani z izjemo pomnilniških kartic na Android telefonih.

Oddaljeno brisanje naprav preiskovalcem predstavlja problem, saj se v primeru, da naprava ni primerno zavarovana in shranjena, lahko izvede nepooblaščeno uničenje dokazov. Zato je pomembno, da ob zavarovanju naprave postopajo po protokolu in čim hitreje. Izvedejo lahko več različnih pristopov. Izvedejo lahko izklop radijskih sprejemnikov v napravi, če je naprava odklenjena, izoliranje naprave od dostopa do povezave z uporabo različnih škatel v obliki Faradayeve kletke, uporabo forenzičnih orodij, ki omogočajo preprečitev izbrisa podatkov.

Izbrisana naprava predstavlja preiskovalcem hud problem. Današnji moderni pametni telefoni privzeto izvajajo enkripcijo podatkov particij, kar v primeru izbrisa enkripcijskega ključa pomeni neuporabne podatke. Izjema izbrisanih podatkov so nekriptirane pomnilniške kartice, ki so izvzete iz ponastavljanja na privzete nastavitve. Vendar v večini primerov na njih ni podatkov aplikacij, kar vključuje tudi zgodovino komunikacij.

4. SLEDENJE IN ODDALJENO BRISANJE NA RAZLIČNIH PLATFORMAH

V naslednjem poglavju bomo predstavili kako poteka sledenje in oddaljeno brisanje na različnih mobilnih platformah.

4.1 iOS

V predstavitvi [7] je opisano, kako se določenim iOS napravam lahko sledi. Nekatere iOS naprave so sledljive tudi po tem, ko ugasnemo napravo. To funkcionalnost omogoča vgrajeni U1 čip. Kljub ugasnjeni napravi omenjeni čip stalno pošilja pulze, ki vsebujejo tudi javni ključ naprave. Ko pride v bližino naprava, ki prejme javni ključ, ta zabeleži trenutno lokacijo in jo zašifrira z javnim ključem naprave. Zatem na strežnike podjetja Apple pošlje zašifrirano lokacijo in zgoščeno vrednostjo ključa. Ob uporabi storitve Find My za iskanje lokacije naprav tako dobimo le zašifrirano lokacijo, ki pa jo lahko dešifriramo le z zasebnim ključem, ki je vezan na uporabnikovo enolično identifikacijsko številko AppleID.

Pri iOS napravah je tudi pomembno razlikovati kako ugasnemo in kako izklopimo Bluetooth, WiFi in dostop do mobilnega omrežja. Če to storimo preko nadzornega centra se storitve ugasnejo običajno le za 24 ur. Če želimo storitve izklopiti trajno, jih je potrebno preko nastavitev telefona.

Oddaljeno brisanje telefona iPhone lahko izvedemo preko iCloud. Tu se je potrebno najprej prijaviti. Ob pošiljanju ukaza za oddaljeno brisanje lahko dodamo sporočilo in telefonsko številko, ki bosta prikazana po koncu brisanja.

V tabeli 1 je prikazan povzetek testiranj iz predstavitve [7]. Testirana je bila dostopnost storitve sledenja na ugasnjenih napravah ter oddaljenega brisanja. Oddaljeno brisanje se izvede le v primeru, ko imamo prižgane vse komunikacijske protokole, ter telefon ponovno prižgemo. Za sledenje brez povezave pa je ključno, da je prižgan Bluetooth v nastavitvah, saj ta predstavlja tudi funkcionalnost UWB čipa U1.

Tabela 1: Sledenje in oddaljeno brisanje

	Sledenje	Odd. brisanje
Radijski sprejemniki aktivni	da	da
Letalski način	da	ne
BT izklj. v nadz. centru	da	ne
BT izklj. v nastavitvah	ne	ne
Radijski sprejemniki izklj.	ne	ne
Faradayeva torba	ne	ne

4.2 Android

Google ima za naprave Android storitev "Find My Device", ki omogoča oddaljeno sledenje lokaciji in oddaljeno brisanje. Ob vsaki lokalizaciji pa se tudi izpiše obvestilo na napravi. Poleg tega pa lahko vidimo druge podatke o napravi kot so enolična številka naprave IMEI, moč signala in v katero mobilno omrežje je naprava povezana, stanje napolnjenosti baterije, datum kdaj je bila naprava nazadnje videna. Poleg tega lahko na daljavo sprožimo predvajanje zvoka na napravi, ki se predvaja 5 minut na glas, tudi če je naprava nastavljena na tiho. Imamo pa tudi možnost oddaljenega zaklepanja naprave in odjave iz Googlovega računa. Na zaklenjenem zaslonu pa lahko ob tem izpišemo sporočilo ali telefonsko številko.

4.3 Ostali

Po podatkih Global Stats Counter [2] je delež operacijskih sistemov (podatki za april 2023), ki niso Android ali iOS le 0,77%.

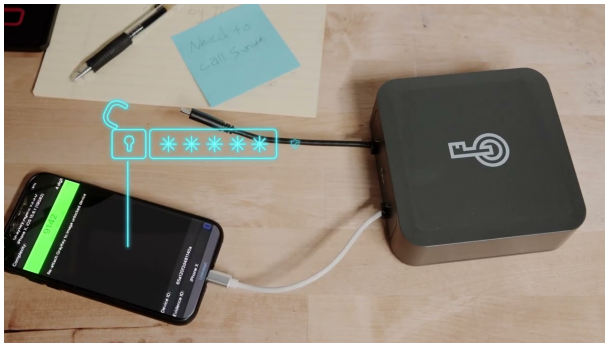
4.3.1 Librem

Librem 5 je telefon proizvajalca Purism, ki je del produktov Librem. Telefon je bil zasnovan z ciljem uporabniku zagotoviti čim večjo varnost in zasebnost. V večini uporablja prosto programsko opremo, vključno z operacijskim sistemom PureOS, ki temelji na operacijskemu sistemu Linux. Ena izmed značilnosti telefona je tudi uporaba mehanskih stikal za izklop strojne opreme (mikrofon in kamera, WiFi in Bluetooth, ter stikalo za izklop omrežja). Ker niso uspeli najti mobilnega modema z prosto strojno-programsko opremo, ga niso integrirali v vezje in s tem ločili od preostale strojne opreme in je na telefon povezan preko vodila USB.

5. ZLORABE/NAPADI

5.1 Lovilec IMSI

Lovilec IMSI (angl. International Mobile Subscriber Identity) [6] je naprava, ki se uporablja za sledenje mobilnih naprav. Gre za vrsto naprave, ki deluje tako, da se predstavlja kot mobilna bazna postaja med mobilno napravo in ponudnikovo bazno postajo - gre se za "man in the middle" napad. Na ta način je mogoče pridobiti identifikacijske podatke od mobilnih telefonov v bližini, približno lokacijo določenega telefona, prisluškovati komunikaciji. Od 3G standarda mobilne komunikacije naprej je sicer prisluškovanje oteženo zaradi obojestranske avtentikacije, vendar obstajajo napadi, ki prisilijo preklop na uporabo starejše storitve.



Slika 3: Naprava GrayKey za ekstrakcijo podatkov z naprav.

5.2 Sledenje z uporabo "object finders"

Apple AirTag, Samsung Galaxy SmartTag in Tile so eni izmed pripomočkov - značke, ki nam pomagajo najti izgubljene predmete. To so značke, ki jih namestimo na predmete, katerim bi radi sledili. Vendar so se te značke zlorabljale tudi za sledenje osebam. Podjetji Apple in Google sta zato napovedala boj proti uporabi značk v take namene [5] in bosta uporabnike mobilnih telefonov opozorila, če zaznata značko, ki sledi uporabniku. Apple je preventivne ukrepe za opozorilo o sledenju neznane značke že izdal. Google pa je napovedal, da se bo tudi sam temu pridružil konec poletja 2023, ko izda svojo storitev sledilnih značk.

6. PRINCIPI DOBRE PRAKSE

Ob izklopu vseh radijskih signalov je mobilna naprava nedosegljiva za namene sledenja njeni lokaciji preko signala UWB in brisanja na daleč. Če to ni možno, vklop letalskega načina prepreči samo njeno brisanje. Če je možno, bo faradajeva skrinja prestregla vse signale, kar lahko nadomesti izklop signalov. Seveda bo ob vklopu signala GPS naprava tudi vidna. Odsvetuje se tudi povezava z neznanimi omrežji Wi-Fi, saj s tem razkrijemo naslov MAC te naprave. S to informacijo lahko nekdo ugotovi splošno geografsko lokacijo.

6.1 Za forenzične preiskovalce

Iz vidika preiskovalcev, je potrebno z zaseženo napravo rokovati previdno. Če naprava ni ustrezno varovana, jo lahko osumljenec vseeno najde ali pa iz nje tudi pobriše podatke z uporabo prej omenjenih postopkov. Za največjo možno varnost in profesionalnost je zadostna izključitev vseh radijskih signalov. Uporablja se tudi orodje GrayKey (glej sliko 3), razvijalca Grayshift [4], ki poleg opravljanja nalog forenzike, tudi avtomatsko izklopi vse radijske signale [7].

6.2 Kako ravnati z Apple napravami

Podrobnosti naprav Apple zahtevajo, da pri izklopu radijskih vmesnikov pazimo na način izklopa Bluetooth-a, saj ta nastavitev vpliva na delovanje U1 čipa. V kolikor onemogočimo Bluetooth iz nadzornega centra, s tem ne bomo izklopili U1 čipa in bomo izklopili Bluetooth le začasno, tipično le za 24 ur. Da popolnoma izklopimo Bluetooth in UWB, moramo to storiti v sistemskih nastavitvah. Vklop letalskega načina onemogoči le povezavo v splet in prepreči brisanje telefona, ne pa tudi sledenje. Primera izklopa vidimo na sliki 4.

Oddaljeno brisanje naprave se lahko izvede le preko aktivne

Slika 4: Na levi sliki prikazan nadzorni center v iOS sistemu, na desni pa nastaviteve vmesnika Bluetooth.

internetne povezave preko mobilnega omrežja ali dostopne točke Wi-Fi. Kljub temu pa ne smemo pozabiti, da proizvajalec neprestano izboljšuje svoje storitve in se lahko zgodi, da bo nekoč dodal možnost oddaljenega brisanja prek ostalih radijskih sistemov.

Sledenje iPhonom se lahko izvaja tudi takrat, ko so te naprave izklopljene. Ob prvem izklopu kompatibilne naprave dobimo na izbiro možnost, da vklopimo oddajanje svoje lokacije ob izklopljenem telefonu.

Za korektno zavarovanje naprave je potrebno poskrbeti, da je naprave odklopljena od omrežja. V primeru, da imamo odklenjen telefon, ki ne zahteva gesla, lahko odidemo v nastavitve in izklopimo vse radije in vklopimo letalski način. Tako nastavljenem telefonu ni mogoče slediti in ne izbrisati podatkov na njemu.

V primeru, da nimamo kode za dostop, lahko poskusimo vklopiti letalski način. Na ta način bomo preprečili izbris telefona, ne pa tudi njegovo sledenje. V kolikor ni mogoče vklopiti letalskega načina, ga je potrebno čim prej prenesti v varno škatlo, ki deluje kot Faradayeva kletka.

Tistim, ki si lastijo primerno orodje, na primer GrayKey, lahko z njim zavarujejo zaklenjen telefon. Z orodjem GrayKey lahko poleg izvoza podatkov opravimo tudi izklop vseh radiev.

6.3 Kako ravnati z Android napravami

V nasprotju z iPhoni Android naprave nimajo zmožnosti biti locirane v primeru, da so izklopljene. Velik premik se bo zgodil v letu 2023, ko je Google napovedal [8], da bo poleg podpore za sledenje sledilnim obeskom dodal tudi možnost podpore lociranja izklopljenih naprav. V svojem blogu ni razkril podrobnosti o implementaciji in katere naprave bodo ob izidu podprte.

Samsung pri svoji storitvi sledenja napravam omogoča, da ta ob izklopu pošlje na strežnik svojo zadnjo lokacijo. Zato predlagamo, da pred izklopom naprave, če je to mogoče, izklopimo radije.

Za razliko od naprav Apple v razdelku obvestil in hitrih nastavitev na zaklenjenem telefonu ne moremo izklopiti radijev brez, da bi ga odklenili. Ta nastavitev je privzeto omogočena na telefonih, ampak jo je mogoče izklopiti v nastavitvah.

Zaradi pomanjkanja podpore sledenja napravam, ko so izklopljene je dovolj, da napravo izklopimo in jo shranimo v varno škatlo.

Mobilne naprave, ki uporabljajo Android in imajo podporo za UWB, lahko to tehnologijo napram Apple telefonom izklopijo ločeno od vmesnika Bluetooth (to zmožnost preverjeno deluje na Samsung Galaxy S23 Ultra). Google izklop omogoča na svojih telefonih od posodobitve januarja 2022.

7. ZAKLJUčEK

Oddaljeno upravljanje s podatki na napravi je uporabnikom omogočilo pomemben nadzor nad podatki v primeru izgube ali kraje mobilnih naprav. To predstavlja izziv preiskovalcem organov pregona, saj velikokrat izbrisana naprava pomeni izgubo podatkov. Zato je pomembno, da se naprava ob zasegu ali zavarovanju s kraja zločina pravilno shrani, da se onemogoči nepooblaščeno brisanje oziroma uničenje dokazov.

8. VIRI

[1] D. Coppens, A. Shahid, S. Lemey, B. V. Herbruggen, C. Marshall, and E. D. Poorter. An overview of UWB standards and organizations (IEEE 802.15.4, FiRa, apple): Interoperability aspects and future research directions. *IEEE Access*, 10:70219–70241, 2022.

[2] S. GlobalStats. Mobile operating system market share worldwide. https://gs.statcounter.com/os-market-share/mobile/worldwide.

[3] A. Gorra. *An analysis of the relationship between individuals' perceptions of privacy and mobile phone location data-a grounded theory study*. PhD thesis, Leeds Metropolitan University, 2007.

[4] Grayshift. Mobile device forensics tools. https://www.grayshift.com/. Dostopano: 14.5.2023.

[5] T. Guardian. Apple and google submit plan to fight airtag stalking. https://www.theguardian.com/technology/2023/may/02/apple-airtag-surveillance-tracking-google-samsung, Maj 2023. Dostopano: 12.5.2023.

[6] Imsi catcher (wikipedia). https://en.wikipedia.org/wiki/IMSI-catcher. Dostopano: 12.5.2023.

[7] M. Kajzer. Offline ios tracking and remote wiping. In *Proceedings of the Digital Forensic Research Conference*, USA, July 2022. DFRWS. Presentation.

[8] E. Kay. How android keeps you and your devices safe. https://www.blog.google/products/android/google-android-safety-features/, May 2023.

[9] WHY UWB IS THE PREMIER LOCATION TECHNOLOGY.

[10] M. Stone. What is ultra-wideband, and how does it work? https://insights.samsung.com/2021/08/25/what-is-ultra-wideband-and-how-does-it-work-3/, Aug 2021.

[11] L.-T. Wang, Y. Xiong, and M. He. *Review on UWB Bandpass Filters*, chapter 5. IntechOpen, 07 2019.

Forenzična analiza mobilnih aplikacij za vzdrževanje avtomobilov

Žan Korošak
Fakulteta za računalništvo in informatiko
Večna Pot 113
Ljubljana, Slovenija
zk4100@student.uni-lj.si

Nejc Velikonja
Fakulteta za računalništvo in informatiko
Večna Pot 113
Ljubljana, Slovenija
nv7912@student.uni-lj.si

POVZETEK
S pomočjo modernih tehnologij v avtomobilih se mobilne aplikacije za sledenje vožnji razvijajo vse bolj napredno. Te aplikacije zagotavljajo ključne podatke o vožnji, kot so hitrost vožnje, pozicija, zdravje avtomobila in zaviranje, ki so lahko uporabljeni v digitalnih preiskavah. Kljub temu se s pojavom teh podatkov poveča količina neuporabnih podatkov, kar lahko vpliva na procesirni čas. Članek poudarja pomembnost uporabe definiranega okvira za razlikovanje stopenj podatkov, vključno z opisom korakov, ki so potrebni za izluščenje uporabnih podatkov. Avtorji primerjajo različne mobilne aplikacije za sledenje vožnji avtomobila ter jih med seboj primerjajo po razširnosti.

Keywords
Forenzika vožnje avtomobila, mobilne aplikacije, forenzična triaža

1. UVOD
Članek obravnava pomembno temo digitalne forenzike - zaseganje in uporabo podatkov o vožnji avtomobila, ki se pridobivajo z mobilnimi aplikacijami med samo vožnjo. Avtomobili, ki so bili ustvarjeni v tem tisočletju, lahko preko protokola OBD-2 oddajajo podatke preko Bluetooth ali WiFi povezave, kar jih naredi zelo uporabne vire forenzičnih artefaktov. V preteklosti se je mobilna forenzika med vožnjo avtomobila osredotočala na informacijsko-zabavne sisteme, ki pa razkrivajo več o uporabniku, kot pa o samem avtomobilu [13].

Zaradi modernizacije novejših avtomobilov so takšni podatki postali vse pogostejši, kar pa odpre velik potencial za uporabo v digitalni forenziki. Kljub temu se s povečevanjem količine teh podatkov povečuje tudi čas, ki je potreben za njihovo zaseganje in preiskavo, kar lahko predstavlja težavo za preiskovalce. V tem smislu članek proučuje koncept "triaže"pri zaseganju podatkov na kraju zločina s strani preiskovalcev. Triaža je proces, ki ga uporabljajo preiskovalci za hitro in učinkovito zbiranje ter analizo podatkov na kraju zločina, s čimer se zbirajo samo ključni podatki za nadaljnjo preiskavo, kar prihrani čas in druge vire [12]. Ta je določena tako za ekspertne digitalne preiskovalce [8], kot za navadne preiskovalce [3]. Avtorji so v kombinaciji s tem uporabili tudi tri definirane metode za ekstrakcijo podatkov (ročna, logična, fizična), ki se razlikujejo po tem, kako ekstremen je poseg v podatke [1], [10].

V članku so nato predstavljene tri mobilne aplikacije - ZUS Smart Vehicle Health Monitor Mini, Veepeak OBDCheck in GoFar - ki so bile med seboj primerjane glede na enostavnost uporabe ter različne zasežene podatke, ki jih omogočajo. Te aplikacije so bile uporabljene v kombinaciji z metodami za ekstrakcijo podatkov na treh različnih avtomobilih različnih znamk in starosti.

2. PREGLED PODROČJA

2.1 OBD-2
OBD-2 (On-Board Diagnostics) protokol se je prvotno uvedel leta 1996 kot standardna metoda za preverjanje emisij vozil. Vendar pa je sčasoma postalo jasno, da ta protokol lahko zagotavlja veliko več kot le informacije o emisijah. Zaradi vedno bolj zapletene elektronike in senzorjev v modernih vozilih, OBD-2 omogoča dostop do različnih podatkov, ki se zbirajo med vožnjo. Ti podatki vključujejo informacije o motorju, menjalniku, zavorah, klimatski napravi, varnostnih sistemih in še več [15].

Mobilne aplikacije, ki so združljive z OBD-2 protokolom, so postale vse bolj priljubljene v zadnjih letih. Te aplikacije omogočajo lastnikom vozil, da spremljajo ključne informacije o svojih vozilih in poskrbijo za njihovo redno vzdrževanje. Poleg tega lahko te aplikacije pomagajo tudi pri odkrivanju težav z avtomobilom, kar lahko prihrani čas in denar pri morebitnih popravilih.

S povečanjem uporabe tehnologije v avtomobilih se število senzorjev in elektronike v vozilih nenehno povečuje, kar vodi do zbiranja vedno več podatkov med vožnjo. Zato je potrebna učinkovita obdelava in interpretacija teh podatkov. Razvoj umetne inteligence in strojnega učenja lahko pomaga pri boljšem razumevanju teh podatkov in njihovi uporabi v različne namene, kot so izboljšanje varnosti na cestah in razvoj bolj učinkovitih ter okolju prijaznih vozil. V kontekstu tega lahko navedemo Tesla vozila kot ekstremen primer uporabe umetne inteligence v avtomobilski industriji [7].

2.2 Ekstrakcija podatkov pri mobilni forenziki

Preiskava mobilnih naprav je postala ključna pri raziskovanju kriminalnih aktivnosti. Avtorji so v delu navedli tri različne metode ekstrakcije podatkov, ki se pi preiskovanju uporabljajo. Te so ročna, logična in fizična. Ročna metoda zahteva, da preiskovalec pregleda celoten vmesnik mobilne naprave ter ročno označi vidne podatke na zaslonu [1]. Logična metoda omogoča preiskovalcu, da kopira logični nosilec podatkov in dostopa do podatkovnega sistema naprave. Fizična metoda pa se osredotoča na ekstrakcijo in kopiranje slike datotečnega sistema [10], [5]. Te metode so bile uporabljene v številnih preiskavah, kot je na primer v primeru Apple proti FBI, kjer je bila uporabljena logična metoda za dostop do podatkov o terorističnem napadu v San Bernardinu leta 2015 [14]. Drug primer je bila uporaba fizične metode v preiskavi umora novinarja Khashoggija, kjer so preiskovalci pridobili slike datotečnega sistema na mobilnih napravah osumljencev. [11]

2.3 Triažni model

V zadnjih letih se je triažni model, ki se uporablja pri digitalni forenziki na kraju zločina, uveljavil kot ena izmed ključnih tem v znanstvenem svetu digitalne forenzike. Zaradi vedno večjih količin podatkov, ki so na voljo za digitalno preiskavo, so postopki preiskovanja postali zahtevnejši za preiskovalce. Vse pogostejše so tudi netemeljite digitalne preiskave, kar lahko privede do neustreznega zbiranja podatkov ali celo izgube dokazov [6]. Zato so avtorji v članku opisali metode za odvzem podatkov, ki jih uporabljajo preiskovalci na kraju zločina. S tem lahko zagotovijo, da odvzamejo samo ključne kose dokazov in zmanjšajo količino potrebnih preiskav.

Poleg človeškega pristopa k triaži obstaja tudi drugačen pristop z uporabo strojnega učenja, ki lahko zmanjša človeško napako na kraju zločina. V delu avtorja Horsmana [4] je predstavljeno ogrodje za izvajanje preiskavne triaže z minimalnim človeškim vnosom, kar dodatno omejuje človeško napako na kraju zločina. Predstavijo objektiven način za ocenjevanje naprav na kraju zločina z uporabo 'COLLECTORS' skale, z katero neki napravi dodelijo oceno. Avtorji v tem članku izvajajo triažo za odvzem živih podatkov iz avtomobilne aplikacije, kjer se podatki, ki so na voljo na napravah med delovanjem, obdelujejo čimhitreje. [6].

2.4 Preiskava mobilnih aplikacij za avtomobilizem

V preteklosti so v digitalni forenziki za preiskavo podatkov v vozilih najpogosteje uporabljali informacijsko-zabavne sisteme (ang. in-vehicle-infotainment - IVI). Informacijsko-zabavni sistemi so vgrajeni v sodobna vozila in omogočajo uporabnikom različne funkcije, kot so navigacija, glasba, dostop do aplikacij itd. Poleg tega so IVI sistemi tudi pomemben vir podatkov o vozniku, vozilu in poti, ki se lahko uporabijo v digitalni forenziki [2].

Aplikacije, ki omogočajo sledenje podatkov avtomobilov, so pogosto uporabljene v digitalni forenziki zaradi njihovega potenciala za pridobitev ključnih informacij o dogodkih v času kaznivega dejanja. Raziskovalci so preučevali različne informacijsko-zabavne sisteme, ki se uporabljajo v avtomobilih, za pridobitev in analizo podatkov. V eni študiji so avtorji Whelan et al. uporabili logično in fizično metodo za ekstrakcijo podatkov iz sistemov Toyota Extension Box in Uconnect v različnih avtomobilih. Ugotovili so, da sistem Toyota Extension Box zagotavlja veliko več podatkov o uporabnikovih aktivnostih v primerjavi z Uconnectom, ki ponuja le osnovne informacije o lokaciji avtomobila [13].

Avtorji v delu Shin et al. izvajajo primerjalno analizo različnih forenzičnih študij nad sistemoma Android Auto in Apple CarPlay. V članku predstavijo tudi lastno forenzično orodje, ki je namenjeno analizi teh informacijsko-zabavnih sistemov [9].

3. METODE

Avtorji so v članku za povezavo med avtomobilom in aplikacijo uporabili On-Board Diagnostics II (OBD-II) priključek, ki se drugače uporablja za servis in diagnostiko vozila. Za komunikacijo med aplikacijo in pametnim telefonom pa so uporabili Bluetooth tehnologijo. Podatki so bili pridobljeni na treh različnih znamkah avtov, in sicer Volkswagen Golf, Mini Cooper in Toyota Corolla. Avtorji so izbrali različne znamke, da ugotovijo ali se artefakti pridobljeni iz podatkov različnih vozil razlikujejo.

Pridobivanje podatkov so izvedli v treh korakih, prvo so uporabili ročno metodo, nato logično in nazadnje še fizično. Tako so se izognili morebitni izgubi podatkov zaradi poškodb naprave, saj si koraki sledijo od najmanj vsiljivega do najbolj. Na ta način lahko prve korake preiskave v postopku triaže prepustijo nestrokovnjakom.

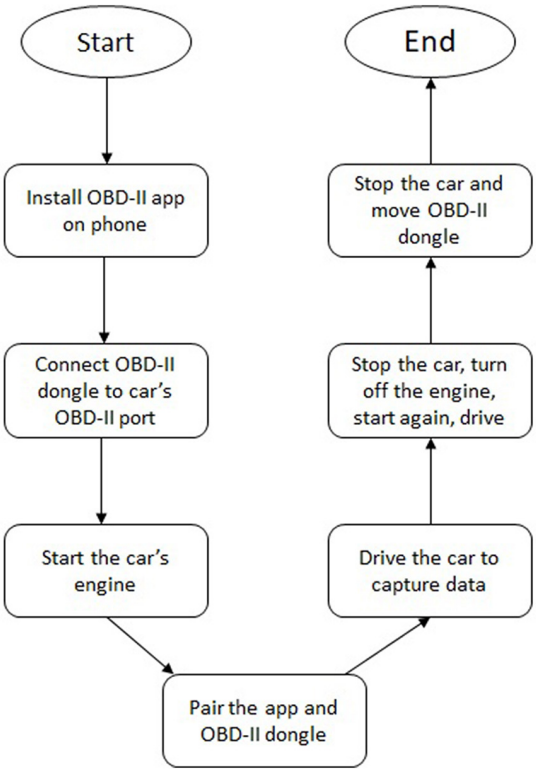
Pri forenzični raziskavo so v članku uporabljaji operacijska sistema Windows 10 Home 64-bit in Kali Linux. Glavno orodje za analizo pa je bil odprtokodni program Autopsy.

3.1 Zbiranje podatkov

V članku so avtorji za zbiranje podatkov uporabljali aplikacije GoFAr, Car Scanner, Veepeak OBDCheck Bluetooth OBD2 Scanner in ZUS Smart Vehicle Helth Monitor mini. Zbiranje podatkov je potekalo s pomočjo treh mobilnih naprav, in sicer Xaomi Redmi 9, Samsung Galaxy A20e in iPhone XR. Zadnja dva mobilna telefona sta bila uporabljena za validacijo natančnosti pridobljenih podatkov z aplikacijo GoFar.

Vsaka aplikacija se poveže z adapterjem za OBD2 priključek. Aplikacija na telefonu se nato poveže z adapterjem preko Bluetooth povezave. Pridobljeni podatki se shranjujejo na oblak. Avtorji so podatke zbirali med vožnjo avtomobilov Volkswagen Golf(2012), Mini Cooper(2018) in Toyota Corolla(2020). Vožnje so potekale v mestu Tallinn po polnoči, saj je takrat količina prometa manjša in omogoča boljšo fleksibilnost hitrosti. Potek je prikazan na sliki 1.

Postopek pridobivanja podatkov poteka celotno pot motorja, kar pomeni, da se adapter priklopi v OBD2 priključek in poveže s telefonom pred vžigom motorja. Avtorji so nato avto vozili približno 3.9 kilometre in nato motor ugasnili preden so končali pridobivati podatke.



Slika 1: Potek pridobivanja podatkov

4. REZULTATI

Rezultati pridobljenih artefaktov za različne aplikacije in modele avtomobilov so prikazani v tabeli 1. Artefakti so bili pridobljeni z ročnim in logičnim načinom pridobivanja podatkov. "Root"način logičnega pridobivanja in fizično pridobivanje podatkov nista prispevala dodatnih artefaktov. Analizo na omenjena načina so vseeno izvedli.

Izmed preverjenih aplikacij v članku, je aplikacija GoFar ponujala največ podatkov in artefaktov. Pri različnih avtomobilih pa se nabor artefaktov ni znatno spreminjal. Edina posebnost je bila pri uporabi Veepeak aplikacije na Toyota Corolli, kjer so avtorji pridobili dva dodatna artefakta tok baterije in napetost baterije.

Avtorji izpostavijo da ročna ekstrakcija podatkov v primeru ZUS in Veepeak omogoča, da strokovnjaki na področju pridobijo podatke o vrtljajih motorja in najvišji hitrosti avtomobila. To je lahko zelo uporabno v primeru prometne preiskave.

Ročno pridobivanje podatkov je potekalo s pomočjo nadzorne plošče aplikacije. Tu se je najbolje izkazala aplikacija GoFar, ki je ponujala največjo količino podatkov (prikazano v tabeli 1) ter preprost izvoz podatkov v datoteko csv.

Logično pridobivanje podatkov je zajemalo pregledovanje datotek uporabljenih aplikacij. Avtorji so ga izvedli na način z "root"dostopom in brez. Ugotovili so, da med njima ni razlike. Pri tem so avtorji pregledovali datoteke kamor aplikacije pišejo podatke (npr. log datoteke). Tukaj avtorji izpostavijo, da so uspeli s logičnim načinom pridobiti VIN številke avtomobilov. VIN številka je unikatna številka avtomobila in je uporabna pri preiskavah, ker lahko iz nje pridobimo nekaj osnovnih karakteristik avtomobila.

Validacijo podatkov so naredili na GPS podatkih vozila. Uporabili so dva dodatna telefona in med vožnjo snemali zaslon GPS aplikacij. Na koncu so podatke pridobljene iz avtomobilov primerjali z posnetki aplikacij. Podobno so preverili tudi zapisovanje hitrosti avtomobila. Posneli so števec hitrosti in z njim primerjali podatke o hitrosti iz GPS aplikacije in podatke pridobljene iz avtomobila. Ugotovili so, da so podatki pridobljeni z aplikacijo GoFar natančni, vendar niso v realnem času. To pripisujejo različnim procesnim sposobnostim OBD-II, kar privede do zanikov podatkov od 1 do 2 sekundi.

V nenehno digitaliziranem svetu se forenzika mobilnih aplikacij in forenzika podatkov o vožnji avtomobila uveljavljata kot pomembni področji raziskovanja. V članku avtorji predstavljajo kombinacijo teh dveh vej, pri čemer se poslužujejo metod za ekstrakcijo podatkov, ki se razlikujejo na pomembne načine. Forenzika mobilnih aplikacij se osredotoča na pridobivanje in analizo digitalnih sledi mobilnih aplikacij, medtem ko forenzika podatkov o vožnji avtomobila zajema digitalne sledi in podatke, ki jih generirajo avtomobilski sistemi, kot so senzorji, GPS podatki in podatki o delovanju motorja. S kombinacijo teh dveh vej lahko preiskovalci pridobijo obsežne količine podatkov o vožnji avtomobila in uporabi mobilnih aplikacijah. Ti podatki so lahko ključni pri reševanju kaznivih dejanj, povezanih z vozili. Z razvojem tehnologij in povečanjem količine podatkov, ki so na voljo preiskovalcem, postaja učinkovita uporaba metod za ekstrakcijo podatkov vedno bolj ključna. Triažni model, ki ga uporabljajo avtorji, lahko pomaga preiskovalcem pri hitrem odločanju o pomembnosti posameznih podatkov.

V članku avtorji primerjajo tri različne mobilne aplikacije, ki so nameščene na treh različnih avtomobilih, in med seboj primerjajo artefakte podatkov, ki jih omogočajo aplikacije. Ugotovijo, da je med količino podatkov velika razlika, glede na to, katero aplikacijo uporabljajo. Konkretno, aplikacija GoFar ponuja skoraj dvakrat toliko podatkov, kot jih ponujata ZUS in Veepeak. Avtorji poleg tega razkrivajo razlike v količini dostopnih podatkov med ročno metodo ekstrakcije in logično metodo. V primerjavi s ročno ekstrakcijo, kjer se preiskovalec osredotoči le na določene datoteke ali mape, logična ekstrakcija ponuja precej več podatkov, kot so recimo podatki o pospeševanju, zaviranju ter najvišji hitrosti. Te podatki so lahko zelo uporabni v preiskavi.

5. LITERATURA

[1] R. Ayers, S. Brothers, and W. Jansen. Guidelines on mobile device forensics, 2014-05-15 00:05:00 2014.

[2] P. Henry. Digital forensics - automotive infotainment and telematics systems.

[3] B. Hitchcock, N.-A. Le-Khac, and M. Scanlon. Tiered forensic methodology model for digital field triage by non-digital evidence specialists. *Digital Investigation*, 16:S75–S85, mar 2016.

[4] G. Horsman. The collectors ranking scale for 'at-scene' digital device triage. *Journal of Forensic Sciences*, 66,

Tabela 1: Povzetek pridobljenih artefaktov

Artefakti	ZUS	Veepeak			GoFar		
	Volkswagen Golf	Volkswagen Golf	Mini Cooper	Toyota Corolla	Volkswagen Golf	Mini Cooper	Toyota Corolla
VIN	X				X	X	X
Ime vozila/ID	X	X	X	X	X	X	X
ID uporabnika							
Email uporabnika					X	X	X
Ime in priimek uporabnika					X	X	X
Čas in datum (vožnja)	X	X	X	X	X	X	X
GPS koordinate vožnje	X				X	X	X
Prevožena razdalja		X	X	X	X	X	X
Najvišja hitrost	X	X	X	X			
Povprečna hitrost		X	X	X	X	X	X
Najvišji obrati/min	X	X	X	X			
Poraba goriva		X	X	X	X	X	X
Pospešek		X	X	X	X	X	X
Zaviranje					X	X	X
CO2 izpuhi					X	X	X
Model in proizvajalec vozila					X	X	X

10 2020.

[5] W. Jansen and R. Ayers. Guidelines on cell phone forensics, 2007-05-30 00:05:00 2007.

[6] D.-Y. Kao, N.-C. Wu, and F. Tsai. The governance of digital forensic investigation in law enforcement agencies. *2019 21st International Conference on Advanced Communication Technology (ICACT)*, pages 61–65, 2019.

[7] D. Kumari and S. Bhat. Application of artificial intelligence in tesla- a case study. *International Journal of Applied Engineering and Management Letters*, pages 205–218, 12 2021.

[8] M. Rogers, J. Goldman, R. Mislan, T. Wedge, and S. Debrota. Computer forensics field triage process model. *Conference on Digital Forensics, Security and Law*, 1:27–40, 06 2006.

[9] Y. Shin, S. Kim, W. Jo, and T. Shon. Digital forensic case studies for in-vehicle infotainment systems using android auto and apple carplay. *Sensors*, 22(19):7196, 2022.

[10] H. Srivastava and S. Tapaswi. Logical acquisition and analysis of data from android mobile devices. *Information and Computer Security*, 23:450–475, 11 2015.

[11] N. Stephens, C. Cadwalladr, and S. Kirchgaessner. Nso spyware used to target family of jamal khashoggi, leaked data shows. *The Guardian*, Jul 2021.

[12] A. Thakar, K. Kumar, and B. Patel. Next generation digital forensic investigation model (ngdfim) - enhanced, time reducing and comprehensive framework. *Journal of Physics: Conference Series*, 1767:012054, 02 2021.

[13] C. J. Whelan, J. Sammons, B. McManus, and T. W. Fenger. Retrieval of infotainment system artifacts from vehicles using ive. *Journal of Applied Digital Evidence*, 1(1), 2018.

[14] Wikipedia. Fbi-apple encryption dispute, 2015. [Online; accessed 10-May-2023].

[15] J. Zaldivar, C. Calafate, J.-C. Cano, and P. Manzoni. Providing accident detection in vehicular networks through obd-ii devices and android-based smartphones. pages 813–819, 10 2011.

5 - UI v forenziki / AI in forensics

Using deep learning to detect social media 'trolls': A review

Patrik Marčok
University of Ljubljana
Vecna Pot 113
1000, Ljubljana
pm29183@student.uni-lj.si

Simon Georg Antonischki
University of Ljubljana
Vecna Pot 113
1000, Ljubljana
sa87801@student.uni-lj.si

Marek Jusko
University of Ljubljana
Vecna Pot 113
1000, Ljubljana
mj26188@student.uni-lj.si

General Terms
Theory

Keywords
Deep learning, Trolls, Social media

ABSTRACT
In this paper, our main goal is to provide a comprehensive summary of a recent article that showcases the effectiveness of deep learning techniques in detecting trolls based on text content. Additionally, We explore similar articles in the existing literature, examining various approaches and their efficacy in identifying trolls and toxicity in both textual and visual content on social media platforms.

Moreover, We discuss the crucial correlation between social media content and criminal activities, underscoring the significance of digital forensics in the analysis of social media data. By delving into this topic, We aim to shed light on the broader implications and consequences of troll behavior and toxic content within online communities.

Furthermore, We propose several enhancements to the implementation of deep learning techniques. These include incorporating computer vision to analyze visual elements, examining the impact of custom emotes, utilizing topic sampling for a more comprehensive analysis, and emphasizing the importance of regularly updating the dataset.

Overall, this paper provides a detailed exploration of the current state of detecting trolls and toxic content on social media platforms using deep learning techniques. Through an extensive review of related literature and a primary focus on summarizing a recent article, We aim to contribute to the understanding of this field and highlight potential avenues for future research.

1. INTRODUCTION
In current days, social media has a big influence on our lives. It influences greatly how humans think and act. On the other side, social media is full of toxicity and criminal activity. Thus methods for predicting these undesirable behaviors are needed. In this Essay, We review already published work that shows deep learning to detect criminal activity and so-called 'trolls' in images.

A troll refers to a social media user, whose only goal is to spread conflict and deception. A meme is a picture or short video (also called GIF), often with text that expresses something.

Furthermore We present a wider summary of the field in terms of the current state of deep learning and show a few ways how to detect and prevent criminal activities on the web.

2. CURRENT STATE
In this section, We will show the current state of social media data and digital forensics, most importantly how to detect toxicity in social media and their correlation.

2.1 Detecting trolls in social media
In 2018, a first approach[1] towards detecting social media trolls was proposed by Ibrahim et al.[7] using different data augmentation approaches, however this only used comments on posts in social media to detect trolls as compared to the main article (see Chapter 3). Here, an ensemble of LSTM, CNN, and GRU performed best in terms of accuracy. In 2019, the approach was expanded by Anand and Eswari[3], which used further augmentation approaches. Here, Glove combined with CNN showed the best results.

In 2021, this approach was further expanded by Husnain et al.[6] by using Logistics Regression, Naive Bayes, and Decision tree classifier as well as a preprocessing schema, which shows a significant improvement.

In 2020, Salminen et al.[11] investigated comments under news videos to find the relationship between toxicity and news topics. Here they found that depending on the topic comments can be more or less toxic. In 2021, Pascual-Ferrá et al.[9] investigated the role of toxicity in the discussion

[1]To the best of our knowledge

about face masks during the COVID-19 pandemic on Twitter. Here they found that tweets with anti-mask hashtags were significantly more toxic compared to tweets with pro-mask hashtags.

2.2 Digital forensics and social media

In 2016, Aghababaei and Makrehchi[1] found a correlation between social media content and crimes, allowing us to improve at crime prediction. This was done by using a prediction model over content posted on Twitter, with data divided up in terms of geographical location. However, it is important to note that the amount of correlation depends on the type of crime, e.g. burglary has a high correlation with the data.

Also in 2019, Bhattacharya and Prithvi[4] showed the effects that memes can have on society, as they only theorised, without providing empirical data. Still, this can be used as a guideline on how to categorize the effect memes have on society. The authors proposed 5 categories:

- The astro-turfing effect, describing that memes can make the information seem reliable, even though it isn't.
- The maligned imaging effect, describing that poor quality of memes can lead to poor societal standing of the creator.
- The con-educating effect, describing that memes can lead to believing false claims[2]
- The discriminatory humor effect, describes that memes can lead to humor at the expense of others.
- The narcissist dreamer effect, describing that memes can lead to individuals being narcissists and only chasing dreams.

In 2019, Arshad et. al. provided a view on digital forensics of social media from a practical side, by describing the then-current state of evidence handling, as well as challenges that arise because of social media. This is also discussed in this book by Powell et al.[10] in more detail.

In 2021, Al-Room et al.[2] presented a novel approach to analyzing UAVs[3] for digital forensics. Hereby they extracted important data, e.g. the locations of the UAV, as well as captured images and videos, flight paths, etc. This data can greatly help when analyzing cases that involve a UAV.

3. SUMMARY OF THE MAIN ARTICLE

Social media platforms have become an inherent aspect of today's lives. These social platforms can bring many benefits, but on the other hand, they can be misused to spread negative behavior. One example of negative behavior is trolling, where users intentionally post offensive comments and content to provoke other users. An example of trolling text is: *Stupid peace of shit stop deleting my stuff asshole go die and fall in a hole go to hell!*. [4].

This article[12] describes how authors use deep learning to detect trolls. Traditional methods for detecting trolls are time-consuming and often inaccurate. As a result, automated deep learning algorithms are used. Deep learning is a sub-field of machine learning that involves training a neural network to recognize patterns in data. The networks analyze data, comments, and posts of users on social media networks, to detect patterns of negative behavior and create a classification model.

3.1 Data and model

The authors divided the data into three sets: training, validation, and test, using a ratio of 80:10:10. This resulted in 20,942 samples for training, 2,617 samples for validation, and 2,617 samples for testing. To achieve a more accurate outcome of the model, they only used two labels: toxic and non-toxic.

The authors used Recurrent Neural Networks (RNN), specifically Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) to train their model.

3.2 Training

Training of the model was performed in 30 epochs with patience set to 5. They used a validation score to stop training when there is no change. The training results are shown in Table 1.

	training loss	validation loss	training accuracy	validation accuracy
GRU	0.5984	0.6086	0.6825	0.6693
LSTM	0.5965	0.6083	0.6671	0.6532
Bi-GRU	0.5852	0.6014	0.6754	0.6547
Bi-LSTM	0.5789	0.6068	0.6903	0.6610
GRU + GloVe	0.1950	0.1950	0.1950	0.1950
LSTM + GloVe	0.1607	0.1903	0.9329	0.9239
Bi-GRU + GloVe	0.1587	0.1932	0.9372	0.9226
Bi-LSTM + GloVe	0.1686	0.1998	0.9353	0.9232

Table 1: Training results

3.3 Result

The training results show that the best-performing model is the Bidirectional LSTM model with GloVe embeddings. However, it is important to ensure that the model is correct, so We perform testing on unseen data. The test results reveal that the Bidirectional LSTM model with an embeddings layer demonstrated the highest performance, achieving 0.921 accuracy and 0.922 F1-measure. The positive effect of word embeddings on the model's performance is evident, as the results improved by 30% upon their implementation. All models using word embeddings achieved excellent results, with metric scores of over 90%. Furthermore, the number of epochs required to train the models decreased to 8. Testing results are shown in table Table 2.

4. IMPLEMENTATION IMPROVEMENTS

In this section, We will explore possible improvements to the implementation of deep learning techniques for the detection of trolls and toxic content on social media platforms. As

[2]Also called "fake news"

[3]Commonly called drones

[4]Taken from `https://www.kaggle.com/code/jannga rcia/tech-exchange-project/notebook`

Model	Accuracy	Precision	Recall	F1-score
GRU	0.662	0.746	0.493	0.593
LSTM	0.657	0.736	0.490	0.587
Bi-GRU	0.670	0.756	0.506	0.605
Bi-LSTM	0.673	0.767	0.497	0.602
GRU + GloVe	0.921	0.928	0.913	0.920
LSTM + GloVe	0.919	0.924	0.913	0.919
Bi-GRU + GloVe	0.917	0.931	0.900	0.915
Bi-LSTM + GloVe	0.921	0.918	0.925	**0.922**

Table 2: Testing results

discussed in the previous sections, the proposed methodology has shown promising results in identifying toxic images based on their embedded text content. However, there is still room for improvement to increase the accuracy and robustness of the model in detecting trolls and toxic content in different contexts. We will discuss several possible ways to enhance the implementation, including incorporating additional features besides text content, considering the context of the content, and monitoring the model's performance over time.

4.1 Computer Vision

One possible way to improve the performance of the deep learning model in detecting trolls and toxic content on social media platforms is to incorporate computer vision techniques. [5] While the study focuses on detecting toxic images based on their embedded text content, images can contain visual patterns associated with toxic content that are not captured by the text alone. Therefore, the incorporation of computer vision techniques, such as object recognition, facial expression analysis, and image clustering, can help improve the accuracy of the model in detecting toxicity.

For example, object recognition techniques can be used to identify objects in an image that are associated with toxic content, such as weapons or drugs. Facial expression analysis can detect negative emotions like anger, disgust, or contempt, which are often associated with trolling behavior.

However, incorporating computer vision techniques can also increase the complexity of the model and require additional computational resources. Therefore, it's essential to balance the benefits of using computer vision techniques with the cost of implementing them in the model.

4.2 Emotes

Incorporating custom emotes can significantly improve the accuracy of the deep learning model in detecting trolls and toxic content on social media platforms. However, the ambiguous meaning of emotes presents a significant challenge. Emotes can have multiple meanings and can be interpreted differently depending on the context in which they are used. Inaccurately interpreting the meaning of emotes can lead to false positives or false negatives in detecting toxicity. [8]

To address this challenge, it is essential to analyze and interpret the context and meaning behind custom emotes accurately. One approach is to analyze the surrounding text and language patterns to better understand the intended meaning of the emote. Another approach is to use computer vision techniques to analyze the visual characteristics of the emote, such as its color, shape, and facial expressions, to determine its meaning.

However, even with these techniques, accurately interpreting the meaning of emotes can be challenging, and there may be cases where their meaning is still ambiguous. In these situations, it may be necessary to rely on additional contextual information, such as the user's history of behavior and interactions on the platform, to determine the intent behind the emote.

Creating a comprehensive database of custom emotes and their meanings can be challenging due to the constantly evolving nature of online communities and trends. Therefore, it's crucial to regularly update the database to keep up with the latest custom emotes and their meanings.

4.3 Topic Sampling

Topic sampling is another technique that can improve the detection of trolls and toxic content on social media platforms. [11] With topic sampling, the model can be trained on specific topics or communities, such as politics or gaming, to better understand the unique language, expressions, and patterns of behavior within that community. By training the model on specific topics, it can better understand the context and sentiment of the content and improve its accuracy in detecting toxicity within that topic or community.

However, training the model on specific topics or communities also presents challenges. It may require a significant amount of data and resources to train the model effectively on each topic, and the model's performance may be limited to those specific topics or communities.

4.4 Updates

Regularly updating the dataset is crucial to ensure the deep learning model's accuracy in detecting trolls and toxic content on social media platforms. As new trends of trolling emerge, the dataset must be updated to include new types of toxic content and language patterns associated with these trends. For example, new slang terms or emojis may be introduced that convey negative connotations or are used in a toxic context.

5. CONCLUSIONS

In this paper, We reviewed the article on detecting trolls in social media. We mention what the authors have done, the way they did it, what methods they used, and the result of their work. We describe the current state of the art of the Field surrounding the mentioned article. At the end, We describe different methods of improvements, like using computer vision to achieve better results, and note the restrictions on these improvements.

6. REFERENCES

[1] S. Aghababaei and M. Makrehchi. Mining social media content for crime prediction. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 526–531, 2016.

[2] K. Al-Room, F. Iqbal, T. Baker, B. Shah, B. Yankson, A. MacDermott, and P. C. Hung. Drone forensics: A case study of digital forensic investigations conducted

on common drone models. *International Journal of Digital Crime and Forensics (IJDCF)*, 13(1):1–25, 2021.

[3] M. Anand and R. Eswari. Classification of abusive comments in social media using deep learning. In *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, pages 974–977, 2019.

[4] P. Bhattacharya. Social degeneration through social media: A study of the adverse impact of 'memes'. In *2019 Sixth HCT Information Technology Trends (ITT)*, pages 44–46, 2019.

[5] C. Courtois and T. Frissen. Computer vision and internet meme genealogy: An evaluation of image feature matching as a technique for pattern detection. *Communication Methods and Measures*, 17(1):17–39, 2023.

[6] M. Husnain, A. Khalid, and N. Shafi. A novel preprocessing technique for toxic comment classification. In *2021 International Conference on Artificial Intelligence (ICAI)*, pages 22–27, 2021.

[7] M. Ibrahim, M. Torki, and N. El-Makky. Imbalanced toxic comments classification using data augmentation and deep learning. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 875–878, 2018.

[8] J. Kim, D. Y. Wohn, and M. Cha. Understanding and identifying the use of emotes in toxic chat on twitch. *Online Social Networks and Media*, 27:100180, 2022.

[9] P. Pascual-Ferrá, N. Alperstein, D. J. Barnett, and R. N. Rimal. Toxicity and verbal aggression on social media: Polarized discourse on wearing face masks during the covid-19 pandemic. *Big Data & Society*, 8(1):20539517211023533, 2021.

[10] A. Powell and C. Haynes. *Social Media Data in Digital Forensics Investigations*, pages 281–303. Springer International Publishing, Cham, 2020.

[11] J. Salminen, S. Sengün, J. Corporan, S.-g. Jung, and B. J. Jansen. Topic-driven toxicity: Exploring the relationship between online toxicity and news topics. *PLOS ONE*, 15:1–24, 02 2020.

[12] Áine MacDermott, M. Motylinski, F. Iqbal, K. Stamp, M. Hussain, and A. Marrington. Using deep learning to detect social media 'trolls'. *Forensic Science International: Digital Investigation*, 43:301446, 2022.

Deepfake Noise Investigation and Detection

Miha Marinko
Faculty of Computer and Information Science
University of Ljubljana
Ljubljana, Slovenia
mm7523@student.uni-lj.si

Luka Šešet
Faculty of Computer and Information Science
University of Ljubljana
Ljubljana, Slovenia
ls90321@student.uni-lj.si

Peter Rotar
Faculty of Computer and Information Science
University of Ljubljana
Ljubljana, Slovenia
pr72753@student.uni-lj.si

ABSTRACT
In recent years, Deepfake technology has been rapidly improving, becoming more nuanced and harder to detect for the naked eye. This imposes challenge for Deepfake detection models: to stay up-to-date and refined in order to stay effective. Authors of this paper introduce new state-of-the-art Deepfake detection model that relies finding altered forensic noise traces.

Keywords
deepfake, neural network, forensics, noise traces, face, siamese model

1. INTRODUCTION
Deepfake is a facial synthesis technique that generates hyper-realistic fake videos using deep neural networks. Although Deepfake has positive effects in various industries, it has also become a serious threat to society, political systems, and businesses. Deepfake detection is challenging because of the high quantity and quality of the online-circulating Deepfake videos, and the existing Deepfake detection algorithms may not perform well against new and unknown Deepfake attacks.

When applying Deepfake modifications to a video, usually some detectable noise traces are left within the face area and the background area is usually unchanged. Authors of the article[10] present forensic approach to detect deepfake videos based on those traces. Implemented algorithm works in multiple stages, where each sample is processed through face-background strategy, Siamese noise trace and noise similarity analysis respectively.

2. DEEPFAKE
Deepfake refers to a facial synthesis technique that uses face-swapping to generate hyper-realistic fake videos using deep neural networks. They were first introduced by Reddit user "deepfakes" in 2017. Since appearing they have had potential for huge negative consequences, where the main problem is that anyone can become a victim of Deepfake in the future. In the following section, we will describe deepfake generation and some of its positive and negative applications.

2.1 Deepfake generation
In deepfake generation, the main architecture used is an autoencoder, which is composed of a shared encoder and two individual decoders. The shared encoder is responsible for learning common facial features, while the individual decoders are trained to generate faces with unique identities.

To create a deepfake, the well-trained autoencoder takes in a cutout video of face as input and passes the encoder-learned facial features to the decoder that corresponds to the source face. The decoder then generates a face with the resembling the source person while maintaining the facial expression and action of the input face.

Recently, the generative adversarial network (GAN) architecture has also been frequently adopted to improve the quality of Deepfakes. GAN consists of a generator and a discriminator that compete with each other to improve the output quality during the training process. The discriminator is periodically trained to distinguish between the real and fake faces generated by the generator, which helps to create more realistic Deepfakes.

After the synthesis process, the generated face is inserted back into the original image frame and further processed with smoothing and blur techniques to eliminate obvious traces of manipulation.

2.2 Positive applications of Deepfakes
There are several positive applications of deepfakes, such as in the fields of entertainment, education, and research. For example, deepfakes can be used in film and television to create realistic visual effects or to re-create historical figures for educational purposes.

2.3 Negative applications of Deepfakes
Deepfake technology, poses serious threats in various aspects of society. One of the potential risks is blackmail, where a deepfake video could be created to incriminate someone falsely. Another alarming use is the creation of non-consensual pornography, where celebrities' faces are placed onto pornographic videos. Fraud is another concern, as scammers use

deepfakes to trick individuals into thinking they are receiving instructions from a trustworthy source. Perhaps the most significant threat is the manipulation of political figures, as deepfake videos could be used to deceive voters and misrepresent politicians, leading to disastrous outcomes. Overall, deepfake technology has the potential to cause severe harm in several areas of society and requires careful consideration and regulation.

The authors of the paper recognized the potential risks associated with deepfake technology and thus proposed an approach to detect such videos.

3. NOISE DETECTION ALGORITHM

3.1 Convolutional neural networks (CNNs)

Convolutional neural network are mainly used in the area that specializes in pattern recognition. Roughly, it is an effective tool for image and video processing such as image classification, face recognition and object detection. It is designed to adapt based on a given input data to successfully perform complex tasks. The CNNs architecture consists of multiple layers, including convolutional layers, pooling layers and fully connected layers. In convolutional layers, network uses convolutional kernels or windows to detect local patterns in the data, filters specific data and pushes certain features in the foreground. The pooling layer is used to downsample the input data reducing its spacial size while retaining important features, making the network more efficient. Fully connected layers are used after convolutional layers and pooling layers, and performs the final classification task. The output is flattened into vector and fed into series of fully connected layers. The output is a prediction based on the learned features.

CNNs have also been used in speech recognition and natural language processing. There exists many CNN based architectures, each with their unique characteristics and performance. Some of these networks are Meso-4, VGG-19, ResNet, EfficientNet, etc. The choice of architecture depends on the specific task, size, complexity of the input data and the available resources. Few of these networks are briefly introduced in Comparative models section.

3.2 Gathering samples

Videos are commonly compressed to save space. Popular compression algorithms divide the video frames into multiple types, where one category called keyframes carries full image information and the others derive from it. In this process some information is lost and thus only keyframes should be used for Deepfake noise trace detection.

Since Deepfake techniques mainly modify only a specific portion of the frame, typically the facial area for face-swapping, the surrounding background area of the video is generally left unaltered. As a result, these regions may contain less noise than the modified areas. Authors capitalize on this idea by employing a face-background strategy that involves cropping the face square and the background square with the maximum Euclidean distance from the face square in a given keyframe. This ensures that the selected background region is less likely to have been manipulated. The algorithm repeats this process on multiple keyframes before proceeding to the next stage.

3.3 Siamese model

Siamese Networks are a type of neural network that have been frequently used for feature comparison. Authors adopt this design to extract noise traces from face and background squares. The squares are passed through modified DnCNN denoiser model to extract the underlying noise traces instead of eliminating them. We use the extracted forensic noise traces to compare patterns and make Deepfake detection decisions.

Improved denoiser uses three types of layers: Convolution, ReLU and Batch Normalization layer. Convolution produces new image where pixels are comprised of predefined computation of weights on a convolution window in the previous image. ReLU changes pixel value to 0 if its' value does not satisfy certain threshold. Bach Normalization speeds up computation by normalizing and adjusting pixel values such that their relation is not deformed. Denoiser goes through convolution layer and ReLU layer. The image is then passed through convolution, batch normalization and ReLU layers 18 times. Lastly, modified image runs through convolution layer, giving desired result.

3.4 Noise similarity analysis

The algorithm detects Deepfakes by comparing the noise traces in the unmodified clean area with the manipulated face, which typically has complex noise traces. The similarity between the two noise representations is computed as an inner product to identify the correspondence between every two vector entries.

Authors decided to compute similarity matrix S by using matrix multiplication since it effects more elements than summation, therefore increasing correspondence between face and background image. Each entry in S is defined as

$$S_{ij} = \frac{F_i \times B_j^T}{\|F_i\|_2^2 \cdot \|B_j\|_2^2}$$

where $\|A\|_2 = max_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2}$ is induced matrix 2-norm (or spectral norm) and F and B represent matrices of extracted noise for the face and background image, respectively.

To refine the similarity entries, the algorithm applies 2D convolutions with fully connected layers. This process computes the probability that the input face is a Deepfake. The model is tuned by

$$L_{CE} = -(t_1 \log \sigma(p)_1 + t_2 \log \sigma(p)_2)$$

where t_i is the ground truth value and $\sigma(p)_i$ is the Softmax prediction for class i upon the final output from the last fully connected layer.

Softmax prediction is a function that computes certain values into probabilities. For a given vector $v = (v_1, \ldots, v_n)$ it states probability of each possible outcome (element of v). We write Softmax function as:

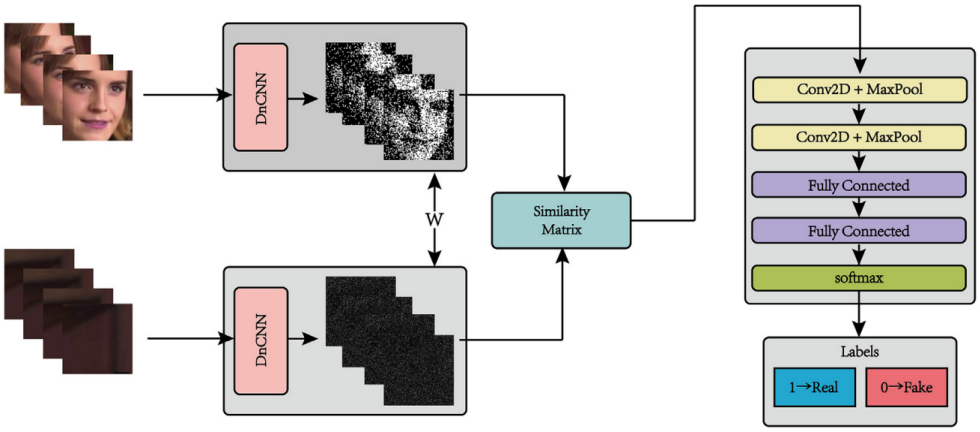


Figure 1: Proposed Deepfake detection model workflow: Keyframe face squares and background squares are processed by the Siamese Network to extract forensic noise traces. These traces are compared using a similarity matrix. The matrix is then refined through convolutional and fully connected layers to enhance features. Then softmax function is used for detection.

$$\sigma(v)_i = \frac{e^{v_i}}{\sum_{j=1}^{n} e^{v_j}}$$

4. EXPERIMENTS

4.1 Dataset

Deepfake technology is relatively new and is changing all the time, that is why is important to have a high-quality dataset for training detection models. Authors used the Celeb-DF dataset, which contains a mix of real and fake videos, including a set of highly difficult videos that challenge state-of-the-art Deepfake detection models. To test the model's effectiveness against unknown Deepfake threats in the future, the proposed noise-based Deepfake detection model was trained and tested on both balanced and unbalanced datasets from Celeb-DF, including the set of highly difficult videos.

4.2 Experiment settings

The proposed model is trained on a balanced dataset of real and fake keyframe pairs, with a total of 45,820 pairs used in the training process. The input face and background squares are resized to 64x64 for consistency.

The model uses a combination of convolutional layers, max pooling operations, and fully connected layers to refine the similarity matrix and decrease the feature dimension. The model was evaluated on both a normal testing dataset and an unknown attack dataset of 518 videos that are of high quality and difficulty. Performance was evaluated using accuracy and AUC score at the frame level. The AUC score is calculated by adjusting all threshold values throughout the whole range from 0 to 1 and plotting over the true positive rate and false positive rate values, providing a measure of the model's robustness.

4.3 Comparative models

For comparison, authors chose models that have source code published and can be reproduced for training and testing the same dataset. For the models that either have no source published or the source code is not reproducible, they adopt the experimental result for particular dataset. The following comparative test models are:

- MesoNet [1] analysis focuses on image properties on mesoscopic level, that is, focusing on a tiny portions of an image. They use CNN based architecture Meso-4 and its optimized variant MesoInception-4. The latter can be reproduced and is used for model comparison.
- Capsule [8] adopts VGG-19 (Visual Geometry Group) network and capsule structure, that is, a capsule divided into three parts: 2D-convolutional part, a statistical pooling layer and 1D convolutional part. After going through all the steps the output is sent to output capsules. From there the final result is calculated. Image keyframe is processed in each capsule, meaning, more capsules model has, the better performance it may have. On the other hand each capsule costs computational power, therefore fewer capsules model has, the faster it computes. The model can be reproduced and is used for model comparison.
- DSP-FWA [11] is an improved method based on spatial pyramid pooling module [3], that is, a modified that can generate a fixed-length representation regardless of image size or scale. That means it can work with DeepFake videos with different resolution qualities. Method takes modified CNN based ResNet as the backbone. Since the model is not reproducible, the authors took reported result of Celeb-DF test dataset.
- Ensemble [2] takes CNN based EfficientNetB4 as the backbone network architecture and proposes attention mechanism for performance improvements. The idea behind is to allow the algorithm to utilize the most relevant parts of the input system. The authors utilize two different approaches for network training: One-to-End and Siamese training. The latter uses triple margin loss which takes anchor sample, positive sample and negative sample. The model can be reproduced and is used for model comparison.

- DFT-MF [4] method focuses on mouth features by cropping mouth from a face in a frame and uses CNN based model to classify videos into real or fake depending on a threshold number of fake frames that are identified by calculating word per sentence, speech rate and frame rate. Since the model can not be reproduced, the authors took reported results of Celeb-DF test dataset.

- FFD [9] takes the CNN based Xception architecture as the backbone. It also utilizes attention mechanism to highlight the manipulated image regions, therefore guide the network to detect these regions. The model can be reproduced and is used for model comparison.

- Face X-ray [6] simulates medical x-ray examination for finding forged images by using key observation: when an image is formed by blending two images, there exists intrinsic image discrepancies across the blending boundary. Therefore the blending boundary shows in forged images while no blending is shown in real images. Method focuses on key factors that contribute to image formation (hardware and software) and are likely to be disturbed in an altered image. Since the model can not be reproduced, the authors took reported results from the paper of Celeb-DF test dataset.

- Multi-Attention [12] uses CNN based EfficientNetB4 network as the backbone. Authors in the paper propose multi-attentional deepfake detection network. Model consists of three parts: multiple spatial attention heads to make the network attend to different local parts, textural feature enhancement block to zoom in the subtle artifacts in shallow features and aggregate the low-level textural feature and high-level semantic features guided by the attention mechanism. Since the model can not be reproduced, the authors took reported results from the paper of Celeb-DF test dataset.

- The Two-Stream [7] uses CNN based model detector that exploits high frequency features and RGB, respectively. In the paper authors use three modules to study and give an adequate representation of utilized features. The model can be reproduced and is used for model comparison.

- TAR [5] (Transfer learning-based Autoencoder with Residuals) uses a residual-based auto-encoder with a latent space Facilitator module that forces and divides the latent space between real and fake embeddings. The model can be reproduced and is used for model comparison.

Authors trained and tested all reproducible models while maintaining their original optimal experimental settings when possible. They were trained on the same dataset, but in some cases where the model's source code was not publicly available, authors used the published checkpoints of already trained models.

4.4 Evaluation results

Firstly, the proposed Deepfake detection model was trained on their balanced training dataset, the second step was to preform detection on difficult unknown attack dataset which consists of 518 videos. Former achieved high frame-level accuracy of 99.15% and AUC score of 99.92%, highest among models with published source code 1. The latter achieved 88.95% AUC score, highest among every model mentioned.

Model names	Accuracy	AUC (%)
MesoNet	92.36	97.82
Capsule	99.02	99.83
FFD	98.69	99.92
Ensemble	70.77	78.15
Two-Stream	92.27	98.19
TAR	50.00	50.00
Authors approach	**99.15**	**99.92**

Table 1: Comparitive test on the normal testing dataset with reproducible models.

For 2 authors extracted reported performance on high challenging Deepfake dataset of 518 videos, where most detection algorithms fail. In this test the authors' model performs above the other models as well. Compared to other detection algorithms this method is more generalized and works well on various types of videos.

Model names	AUC score (%)
MesoNet	70.57
Capsule	77.55
FFD	81.05
Ensemble	80.83
Two-Stream	84.19
TAR	50.00
DSP-FWA	64.60
DFT-MF	71.25
Face X-ray	80.58
Multi-Attention	67.44
Authors approach	**88.95**

Table 2: Comparative test on the unknown attack, challenging Deepfake dataset.

4.5 Forensic noise trace visualization

Authors state that although some state-of-the-art Deepfake detection models have displayed good performance in accuracy, none of them have the ability to visualise forensic traces as evidence. Recent works have used heat-maps, but they do not display distinguishable difference between real and altered faces. Authors imply, that in addition to constructing a model which outperforms state-of-the-art Deepfake detection models, they are able to extract and visualise Deepfake forensic noise traces. They took values of the noise trace extractor in Siamese structure part and displayed extracted forensic noise traces. As 2 shows, there is distinguishable difference between real and altered faces. The more complicated face extracted noise trace is, higher the likelihood of image being altered. This straight forward visual evidence further supports models detection performance.

5. CONCLUSION

Deepfake technology has been improving overtime as researchers and developers continue to refine algorithms and features. As Deepfake technology increases, so does the demand for Deepfake detection algorithms.

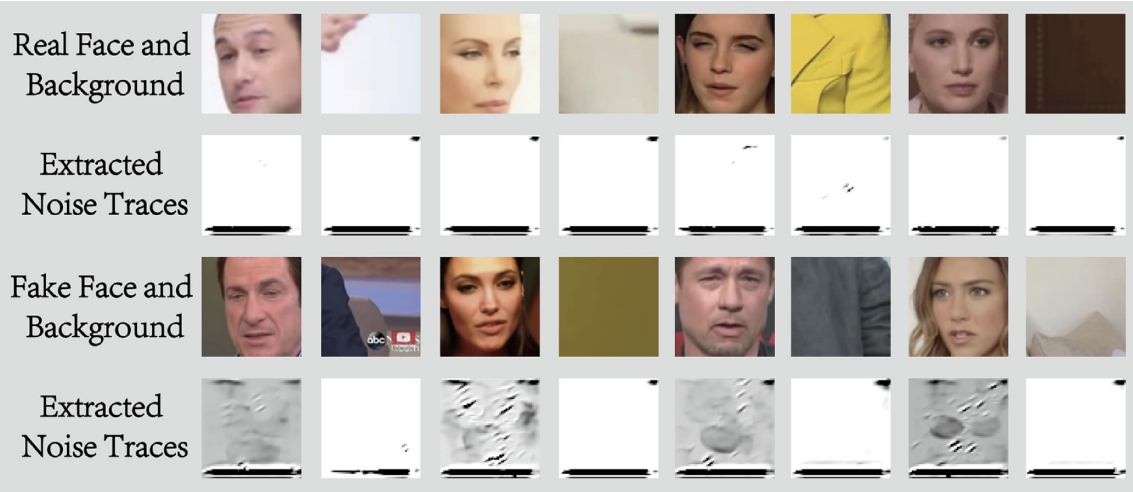


Figure 2: Visualisation of extracted images and their noises. First and third row are presented by face and background of the image, respectively. First two rows present real face and background and extracted noise traces. Last two rows present fake face and background and extracted noise traces. Reader can quickly distinguish the difference in noise between real and altered faces. The image is taken from the paper.

Authors of this paper introduced a new Deepfake detection model that relies more on forensic approach than other state-of-the-art Deepfake detection models. With complex convolutional neural network architecture, authors are able to extract noise traces from a given dataset. The more noise certain data had, higher the likelihood it was altered.

Authors compared their work with other detection models and showed it out-performs every state-of-the-art Deepfake detection model both on normal and unknown, challenging dataset. Furthermore, the visualised noise traces further supports models ability to detect altered faces.

6. REFERENCES

[1] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen. Mesonet: a compact facial video forgery detection network. *CoRR*, abs/1809.00888, 2018.

[2] N. Bonettini, E. D. Cannas, S. Mandelli, L. Bondi, P. Bestagini, and S. Tubaro. Video face manipulation detection through ensemble of cnns. *CoRR*, abs/2004.07676, 2020.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *CoRR*, abs/1406.4729, 2014.

[4] M. Jafar, M. Ababneh, M. Al-Zoube, and A. Elhassan. Forensics and analysis of deepfake videos. pages 053–058, 04 2020.

[5] S. Lee, S. Tariq, J. Kim, and S. S. Woo. TAR: generalized forensic framework to detect deepfakes using weakly supervised learning. *CoRR*, abs/2105.06117, 2021.

[6] L. Li, J. Bao, T. Zhang, H. Yang, D. Chen, F. Wen, and B. Guo. Face x-ray for more general face forgery detection. *CoRR*, abs/1912.13458, 2019.

[7] Y. Luo, Y. Zhang, J. Yan, and W. Liu. Generalizing face forgery detection with high-frequency features. *CoRR*, abs/2103.12376, 2021.

[8] H. H. Nguyen, J. Yamagishi, and I. Echizen. Use of a capsule network to detect fake images and videos. *CoRR*, abs/1910.12467, 2019.

[9] J. Stehouwer, H. Dang, F. Liu, X. Liu, and A. K. Jain. On the detection of digital face manipulation. *CoRR*, abs/1910.01717, 2019.

[10] T. Wang, M. Liu, W. Cao, and K. P. Chow. Deepfake noise investigation and detection. *Forensic Science International: Digital Investigation*, 42:301395, 2022.

[11] X. Yang, Y. Li, and S. Lyu. Exposing deep fakes using inconsistent head poses. *CoRR*, abs/1811.00661, 2018.

[12] H. Zhao, W. Zhou, D. Chen, T. Wei, W. Zhang, and N. Yu. Multi-attentional deepfake detection. *CoRR*, abs/2103.02406, 2021.

Razložljiva Umetna Inteligenca za Digitno forenziko: Proti ublažitvi nezaupanja v Digitalno forenziko, ki temelji na Umetni inteligenci z intepretabilnimi metodami

Ivo Pajer
Fakulteta za Računalništvo in Informatiko
Večna pot 113
Ljubljana, Slovenija
ip3586@student.uni-lj.si

Nejc Ahtik
Fakulteta za Računalništvo in Informatiko
Večna pot 113
Ljubljana, Slovenija
na4379@student.uni-lj.si

Veronika Sovdat
Fakulteta za Računalništvo in Informatiko
Večna pot 113
Ljubljana, Slovenija
vs4086@student.uni-lj.si

1. POVZETEK

Ta članek raziskuje pomembnost razložljivosti pri analizi forenzike na podlagi umetne inteligence ter preučuje potencial razložljivih modelov pri reševanju skrbi glede zaupanja in transparentnosti. Ker digitalna forenzika postaja vse bolj odvisna od tehnik umetne inteligence, nejasnost teh algoritmov povzroča dvome o njihovi zanesljivosti. Z vključitvijo razložljivih modelov preiskovalci pridobijo vpogled v odločitvene procese sistemov, kar povečuje zaupanje in sprejemanje dokazov, ki jih generira umetna inteligenca. Članek poudarja koristi razložljivih modelov pri zmanjševanju pristranosti, zagotavljanju pravičnosti ter mostu med naprednimi tehnikami umetne inteligence in človeško razumljivostjo, kar končno povečuje zaupanje v analizo digitalne forenzike.

Ključne besede

umetna inteligenca, razložljiva digitalna forenzika AI, zmanjševanje nezaupanja, analiza na podlagi umetne inteligence, razložljivi modeli.

2. UVOD

V zadnjih dveh desetletjih strojno generirani dokazi prevladujejo pri iskanju dokazov s povečano natančnostjo [22]. Pojavljajo pa se skrbi v legalnosti le teh, saj se dokazi med različnimi modeli lahko razlikujejo, kot pri človeških strokovnjakih. Pri strojnih rešitvah se pojavi problem zaprte škatle [10], ki preiskovalce lahko pripeljejo do napačnih zaključkov [21, 4]. Najverjetnejši razlogi za to so slabi algoritmi, neprimerni ali manipulirani nabori podatkov, ter okvarjene sistemske komponente. S tem je pomembno razločiti med strojem in človekom, saj je človek samo razlagalec, in ne edini vir izjav, zato je strojno usmerjena forenzična raziskava produkt "porazdeljenega spoznavanja" med strojem in človekom [8]. Problem UI in njene nepreglednosti je še vedno prisoten, saj se teh modelov ne da interpretirati oz. razložiti [5]. Zaključki raziskav z UI so s tem lahko znanstveno vprašljivi ter premalo transparentni. Verjetno pa bo potrebno s povečanjem prefinjenosti tehnologij uporabljati bolj robustne in enako inteligentne sisteme, kot na primer UI za identificiranje morebitnih dokazov.

Prvi cilj dela je preučiti ideje razložjivosti in interpretabilnosti UI, s poudarkom na digitalni forenziki. Drugi cilj je diskusija modela zaprte škatle in potencialne rešitve tega problema. Glavni prispevek tega članka pa so priporočila za omilitev nezaupanja v digitalno forenzične raziskave s pomočjo UI.

3. KONCEPTI

Obljuba UI je bila doseči boljše odločanje, kot videno v nekaterih oblikah medicinskega diagnosticiranja [6] ali spremljanja morebitnih finančnih goljufij [2]. Je pa bilo o tem podano nekaj dvomov v kontekstu pravosodja in policije [2]. Veliko povpraševanja je na temo razlaganja odločitev UI, kar omogoča veja raziskovanja imenovana Razložljiva UI (RUI), ki se fokusira na izdelovanju UI sistemov, kjer so podatki in delovanje sistema - to imenujemo t.i. steklena škatla - prikazani transparentno [11]. Da bolje razumemo te koncepte, so v naslednjih podpoglavjih razloženi novi pojmi.

3.1 Razložljivost

Govori o ideji, da se poveže strojni proces odločanja s človeškimi razlagami, ki so natančne in razumljive [12]. UI modeli in njihovi izhodi naj bi bili razloženi tako, da jih človek lahko razume in sprejme. Še posebej se ta problem kaže pri globokih nevronskih mrežah.

3.2 Razumljivost

Se nanaša na funkcije modela, ki mu omogoča samoumevnost pri funkcionalnosti - ne potrebujemo razlage o notranji strukturi.

3.3 Dojemljivost

Sposobnost modela, da razloži svoj proces učenja na dojemljiv način [12]. Pomaga, da v UI dodamo deduktivne simbole, ki pomagajo pri obratnem inženirstu umetne inteligence.

6 - Razno / Miscellaneous

Analysis of Alt-Right Social Media Platforms

Yannik Dreß
Technische Universität Berlin
Straße des 17. Juni 135
10623 Berin
y.dress@campus.tu-berlin.de

Dominik Göller
Technische Universität Berlin
Straße des 17. Juni 135
10623 Berlin
goeller@campus.tu-berlin.de

ABSTRACT

This paper examines the increasing popularity of alt-right platforms from two angles: a forensic analysis of the digital artifacts they produce and an exploration of their sociopolitical impacts.

Previous investigations uncovered that significant private and secure information can be extracted from these platforms. Vulnerabilities concerning user privacy, exposing opportunities for unauthorized data access were identified. In addition, we review the ASNAAT tool designed to aggregate this data for digital forensic investigations.

The study further explores the broader societal consequences of these alt-right platforms. We delve into their influence on political discourse. The challenges of regulating these platforms, balancing free speech with the prevention of harm, are also discussed.

The paper concludes with the understanding that these platforms, while promoting diverse discourse, also pose potential risks to societal cohesion and democratic processes. While also lacking secure communication and data transfer protocols.

Keywords

forensic analysis, social media, disinformation

1. INTRODUCTION

Over the last few years there has been a increasing popularity in the use of alternative tech social platforms like Parler, Gettr and MeWe. This is due to the exclusion of far-right people on well-established social networks and the efforts of those networks to prevent the spread of disinformation. A good example for this is the ban of Donald Trump from Twitter in the aftermath of the Capitol attack in the United States on the 6th of January in 2021.

However, the rise of these alternative tech social platforms has also raised concerns about their role in amplifying hate speech, misinformation, and extremist content. As these platforms become more mainstream, it is important to consider the implications of their popularity for online discourse and societal values.

This paper discusses the efforts towards the forensic analysis of those alternative social media platforms and the development of tools in order to help digital investigators in their work with their applications. There is the need to put this information into further context. Thus, we continue with the exploration of the sociopolitical consequences these alt-right social media platforms impose. We discuss key points like disinformation campaigns, the creation of echo chambers as well as the attempts to regulate such platforms.

2. FORENSIC ANALYSIS OF ALTERNATIVE SOCIAL MEDIA APPLICATIONS

Thus far there has not been a lot of forensic analysis of alternative social media applications. The main contribution was made by [Joh+22] who analyzed the forensic artifacts of nine different applications which we will also focus on in this paragraph. The authors provided a mobile and network forensic analysis of the applications for Parler, GETTR, MeWe, Wimkin, 2nd1st, Clouthub, Minds, and SafeChat and discussed the forensic artifacts found for each application.

Their approach consists of four steps, namely

1. Setup and scenario creation,
2. data acquisition,
3. data analysis, and
4. providing enhancements to the forensic tool

they developed in order to help investigators finding forensic artifacts from those applications.

The Alternative Social Networking Applications Analysis Tool (ASNAAT) has the purpose to aggregate forensic relevant information from alt-tech social platforms. [1] It extracts and presents critical pieces of information to the inves-

[1] https://github.com/BiTLab-BaggiliTruthLab/ASNAAT

Algorithm 1 High-level Automation Algorithm

Requirements: Python3
Input: TAR image of a device.
Output: HTML Report

Select files to compare:
if *"Help Option"* **then**
 | show_manual();
else if *"Apple Tar"* **then**
 | A();
else if *"Android Tar"* **then**
 | B();

Select from installed apps (For terminal output):
if *"All"* **then**
 | AnalyzeApps = Installed;
else if *"Specified"* **then**
 | AnalyzeApps = Specified;

All apps analysis:
Initial_hash();
Search_archive();
extract_found();
analyze_files();
check_hashing();
generate_report();

Figure 1: Algorithm

tigator and assists him in finding evidence. Each found artifact file is automatically hashed with SHA256 to uniquely identify it. Figure 1 shows the high level algorithm of ASNAAT.

The tool only takes in tar forensic images of Android or Apple smartphones. Furthermore, the device is analysed using manually established wordlists containing filenames and file paths. It targets artifact types such as SQLite databases, XML files and files found within caches that are relevant. The investigator can execute this tool via the command line.

Figure 2 shows an example Apple forensics report output. The box at the top of the image shows the report information such as case number, timestamp and its hashes. Below you can find the several alt-right applications and their artifacts presented as a column interface for understandable readability.

2.1 Forensic Artifacts

From every application forensic artifacts can be found in the mobile device examined. Those artifacts range from user information to post and comment information to even private chat message information. Those artifacts will be shortly presented here.

User Information. Six out of the nine applications tested stored user account information. This information included the username, user ID, full name, email or phone number

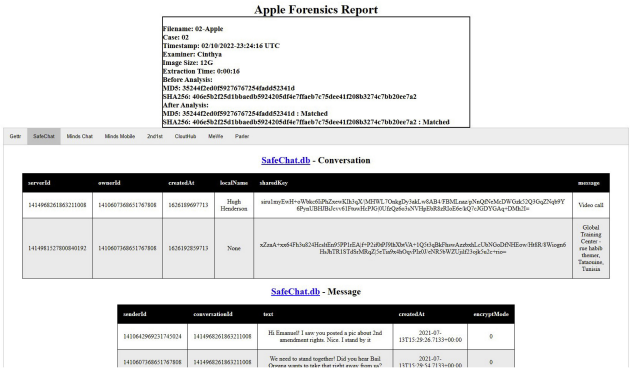


Figure 2: ASNAAT Output

used during login, and timestamps related to events. In addition, thirteen databases also stored user information related to the contact the account owner was interacting with. In addition, four XML files were found to contain user information, with one of them containing the *unsalted* MD5 hashed password to log into the user's account. Finally, one of the files stored notification information, including private chat information, file attachments, and the user ID, full name, and username of the contact the account owner was interacting with.

Posting and Comment Information. Out of the nine applications examined, eight stored some information about postings and comments. Seven databases contained post and/or comment information, with five storing user IDs, full names, usernames, timestamps, post and comment IDs, and contents. One XML file stored notifications about posts or comments on the account owner's page. The Gettr application's libCachedImageData.db database stored metadata related to GIFs posted by the account owner, including a URL to access the GIF and a timestamp. Two XML files stored the ID of recent GIFs posted by the account owner and plain text keywords of GIFs searched by the account owner. Conversely, the SafeChat.db database contained the post ID and timestamp of other posts clicked on and recently viewed by the user. Finally, eighteen out of the thirty-four directories found stored posted videos and images

Private Chat Information. Out of the nine applications studied, five stored some form of private messages locally. The study found nine artifacts that stored chat information, including user IDs, names, timestamps, and plain text chat messages, with some messages containing links and attachment information. Six artifacts contained usernames of chat participants. The study also found that media and other types of files sent and received through private messaging were automatically downloaded to the device. Of the thirty-four folders discovered containing media and other types of files, seventeen contained files sent or received through private messaging. One database from the SafeChat application was discovered to store information on files downloaded from private messages, including the ID, URL, filename, and timestamp of the download. However, not all of the dis-

cussed applications have the functionality for private chats

Cached Data. Also cached data retrieved from mobile applications has been analyzed, which is stored on the device to aid in retrieving information faster and provide an efficient user experience. This data is supposed to be temporary and will eventually clear itself or allow manual clearing. However, cached data can contain structured and unstructured data that can be useful in investigations. Cached data could contain critical artifacts that could be categorized as evidence in an investigation. Out of the nine applications examined, eight contain some type of cached data. Cached data can include images/videos posted by the account owner, sent/received through private messages, profile pictures of public users, and emojis. In addition to media files being cached and stored by the applications, three of the nine applications tested contained cache databases populated with other cache data, such as posts, comments, and user information.

Unauthorized Access to Private Data. Furthermore, vulnerabilities were found in three out of the nine applications tested, which allowed access to personal data without proper authorization. Unencrypted links to media posted and sent through direct messages were stored in databases and could be accessed without proper authorization. Certain URL fuzzing attacks could be used to target servers hosting these data, causing attackers to download personal files. One of the databases stored private information from channels that had been created, including phone numbers, addresses, and emails that had been entered during the account creation. The authors chose not to provide a more detailed description of the vulnerabilities to prevent malicious actors from exploiting them.

2.2 Alternative Social Networking Applications Analysis Tool

To further support digital investigators in their work the authors developed the Alternative Social Networking Applications Analysis Tool (ASNAAT, GitHub Link). It provides its user with aggregated forensic data of the examined alternative social media applications which can be useful to collect evidence on someones phone. The application-specific wordlist files are utilized to gather relevant artifacts from the mobile phone images. However because of this approach the tool is not very adaptable to changes for example in the file management or even usable for other apps. For example after an update of the Parler application the amount of data found on the device was significantly less. Thus, a constant checking of changes needs to be done to keep the tool up to date to all nine applications. To our knowledge this does not happen as the last structural update of the Code on GitHub was about two years ago.

3. IMPACT ANALYSIS: THE SOCIOPOLITICAL CONSEQUENCES OF ALT-RIGHT SOCIAL MEDIA PLATFORMS

As digital platforms continue to reshape our societal interactions, it is becoming increasingly crucial to understand the broader implications of these digital spaces. This is particularly true for alt-right social media platforms, which have demonstrated a significant influence on political discourse, social polarization, and even real-world events. This section aims to delve into the multifaceted impacts of these platforms, exploring their role in shaping politics, societal norms, and attitudes.

Political Influence. Alt-right social media platforms significantly shape political discourse and election outcomes by disseminating alternative political viewpoints [AG17]. The spread of misinformation during the 2016 U.S. presidential election, for example, impacted voters' perceptions. Additionally, these platforms function as echo chambers, reinforcing ideologies and intensifying partisanship, thereby affecting democratic processes. The following section will explore one of these mechanisms: the creation of echo chambers.

Echo Chambers and Polarization. These platforms often exacerbate political polarization by acting as echo chambers [QSS16]. Users frequently encounter and engage with content that aligns with their existing beliefs, thereby isolating them from diverse views. This dynamic hardens ideological stances and heightens political polarization, posing a challenge to democratic dialogue and consensus-building. The echo chamber effect underscores the need for strategies fostering a more balanced and inclusive digital discourse.

Disinformation Campaigns. Alt-right platforms also fuel disinformation campaigns, posing a serious threat to societal trust and institutional integrity [VRA18]. Such campaigns can erode public trust, amplify societal divisions, and disrupt democratic processes. The capacity of these platforms to spread false narratives underscores the need for improved digital literacy, robust fact-checking mechanisms, and effective platform moderation policies.

Societal Impact. Beyond politics, these platforms have profound societal impacts, contributing to increased hate crimes and societal divisions [MS21]. By enabling the spread of hate speech and extremist ideologies, these platforms can serve as catalysts for real-world violence. The unchecked propagation of divisive content can fuel societal tensions and contribute to a climate of intolerance and hostility, necessitating a comprehensive approach to content moderation and digital education.

Regulatory Response and Challenges. The task of regulating alt-right social media platforms is fraught with challenges, primarily due to the delicate balance that must be maintained between upholding freedom of speech and preventing harm [Kei+16]. These platforms, often championed as bastions of free discourse, can become conduits for harmful content, including hate speech, disinformation, and extremist ideologies.

From a regulatory standpoint, the challenge lies in defining

the boundaries of acceptable content. While it is critical to prevent the spread of harmful and misleading content that can contribute to societal discord, it is equally important to safeguard the fundamental principle of free speech. This means ensuring that users have the right to express differing political ideologies, even if they diverge from mainstream perspectives.

These considerations complicate efforts to moderate content on these platforms. Overzealous regulation risks censoring legitimate political discourse and infringing on users' rights to freedom of expression. On the other hand, lax regulation allows for the unchecked spread of harmful content, fueling division and potentially inciting violence.

Moreover, the global nature of these platforms presents additional regulatory challenges. Different jurisdictions have varied legal frameworks and cultural norms regarding freedom of speech and acceptable content, complicating the implementation of universally applicable moderation policies.

In conclusion, while regulating alt-right social media platforms is both necessary and challenging, it is a task that requires nuanced and flexible strategies. Policymakers and platform administrators must collaborate to devise effective moderation policies that curb harmful content while respecting users' rights to free speech. This necessitates an ongoing dialogue that is responsive to the evolving digital landscape and the complex interplay of factors that shape online discourse.

4. CONCLUSIONS

The rise of alternative social media platforms like Parler and MeWe has amplified the spread of radical propaganda and false information. Previous research identified valuable data from these platforms for digital forensics investigations and revealed user privacy vulnerabilities. Most importantly, user information such as usernames, emails, full names, phone numbers, profile pictures, and more could be found, along with posts and comments made, and private messages.

These platforms also impact political discourse, fuel polarization and disinformation campaigns, and contribute to societal divisions and violence. The echo chambers they create and the unchecked spread of disinformation disrupt democratic processes and erode public trust.

Our work has contributed to forensic analyses and impact analysis of alt-right social media platforms. The challenge ahead lies in regulating these platforms, fostering balanced digital discourse, and mitigating their potential harm to democratic systems and societal cohesion.

References

[Kei+16] Teo Keipi et al. *Online hate and harmful content: Cross-national perspectives*. Taylor & Francis, 2016.

[QSS16] Walter Quattrociocchi, Antonio Scala, and Cass R Sunstein. "Echo chambers on Facebook". In: *Available at SSRN 2795110* (2016).

[AG17] Hunt Allcott and Matthew Gentzkow. "Social media and fake news in the 2016 election". In: *Journal of economic perspectives* 31.2 (2017), pp. 211–236.

[VRA18] Soroush Vosoughi, Deb Roy, and Sinan Aral. "The spread of true and false news online". In: *science* 359.6380 (2018), pp. 1146–1151.

[MS21] Karsten Müller and Carlo Schwarz. "Fanning the flames of hate: Social media and hate crime". In: *Journal of the European Economic Association* 19.4 (2021), pp. 2131–2167.

[Joh+22] Hailey Johnson et al. "Alt-tech social forensics: Forensic analysis of alternative social networking applications". In: *Forensic Science International: Digital Investigation* 42 (2022), p. 301406.