

UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

UNIVERSITY OF LJUBLJANA
FACULTY OF COMPUTER AND INFORMATION SCIENCE

ZBORNIK

DIGITALNA FORENZIKA

**Seminarske naloge,
2024/2025**

Ljubljana, 2025

Zbornik

Digitalna forenzika, Seminarske naloge 2024/2025

Editors: Andrej Brodnik, Miha Frangež, Dimitrije Stefanović, študenti

Template authors: David Klemenc, Miha Frangež

Ljubljana : Univerza v Ljubljani, Fakulteta za računalništvo in informatiko 2025.

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

Kazalo / Contents

1	Uvod / Introduction	2
2	Povzetki / Summaries	3
2.1	WARNE: Orodje za zbiranje dokazov o zalezovalski programski opremi	3
2.2	How to detect cryptocurrency miners? By traffic forensics!	3
2.3	Was the clock correct? Timestamps and time anchors across operating systems	3
2.4	Enhancing speaker identification in criminal investigations through clusterization and rank-based scoring	3
2.5	Oblachna forenzična analiza Amazon iRobot Roomba sesalca	4
2.6	Hit and run: Forensic vehicle event reconstruction through driver-based cloud data from Progressive's snapshot application	4
2.7	Grand Theft API: A Forensic Analysis Of Vehicle Cloud Data	4
2.8	Applying digital stratigraphy to the problem of recycled storage media	4
2.9	Forenzični vidiki zloženih datotečnih sistemov	5
2.10	Forenzična analiza naprave Steam Deck: Predstavitev in razširitev članka Well Played, Suspect!	5
2.11	Beyond the 3DS: A Cross-Platform Study Inspired by Nintendo 3DS Residual Data Analysis	5
2.12	Analyzing Deterministic and Heuristic Approaches to JPEG Fragmentation Detection	5
2.13	Natančnost geolokacijskih metapodatkov slik pametnih telefonov	6
2.14	Mobile Device Forensics: A Targeted Overview	6
3	Razno / Miscellaneous	7
3.1	WARNE: Orodje za zbiranje dokazov o zalezovalski programski opremi	8
3.2	How to detect cryptocurrency miners? By traffic forensics!	14
3.3	Was the clock correct? Timestamps and time anchors across operating systems	22
3.4	Enhancing speaker identification in criminal investigations through clusterization and rank-based scoring	27
4	IoT in avtomobili / IoT and cars	32
4.1	Oblachna forenzična analiza Amazon iRobot Roomba sesalca	33
4.2	Hit and run: Forensic vehicle event reconstruction through driver-based cloud data from Progressive's snapshot application	38
4.3	Grand Theft API: A Forensic Analysis Of Vehicle Cloud Data	43
5	Datoteke in sistemi / Files and systems	50
5.1	Applying digital stratigraphy to the problem of recycled storage media	51
5.2	Forenzični vidiki zloženih datotečnih sistemov	54
5.3	Forenzična analiza naprave Steam Deck: Predstavitev in razširitev članka Well Played, Suspect!	59
5.4	Beyond the 3DS: A Cross-Platform Study Inspired by Nintendo 3DS Residual Data Analysis	66
6	Mobilna forenzika / Mobile forensics	75
6.1	Analyzing Deterministic and Heuristic Approaches to JPEG Fragmentation Detection	76
6.2	Natančnost geolokacijskih metapodatkov slik pametnih telefonov	81
6.3	Mobile Device Forensics: A Targeted Overview	86

1 Uvod / Introduction

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

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

This conference gathers the seminar papers of master's students at the Faculty of Computer Science and Information, University of Ljubljana 2024/2025. It explores a variety of topics including mobile forensics, logs, password forensics and cyphering, browser forensics and forensic investigation. Each paper contributes to the ongoing development and application of digital forensic techniques, enhancing our ability to navigate the complexities of today's digital landscape.

Multiple papers focus on Internet of Things devices and automotive systems, reflecting the increasing integration of cloud-connected technologies in everyday objects. One paper presents cloud forensic analysis of iRobot Roomba vacuum cleaners, demonstrating how household IoT devices can contain valuable digital evidence. Two papers examine vehicle forensics through cloud data analysis: one investigates forensic vehicle event reconstruction using mobile car insurance application data for hit-and-run cases, while another provides a comprehensive forensic analysis of vehicle cloud data systems. These works highlight the forensic potential of connected vehicles and IoT devices in modern investigations.

In files and systems forensics, fundamental aspects of file systems and storage media are addressed, covering both traditional and gaming platforms. One paper applies digital stratigraphy concepts to recycled storage media problems, offering new approaches to data recovery from reused storage devices. Another examines forensic aspects of complex file systems, addressing challenges in analyzing modern storage architectures. Two papers focus on gaming platform forensics: one extends forensic analysis of Steam Deck devices, while another conducts cross-platform studies inspired by Nintendo 3DS residual data analysis, demonstrating the forensic value of gaming device artifacts.

In mobile forensics, various aspects of mobile device examination are explored, from image analysis to comprehensive device investigation. One paper analyzes deterministic and heuristic approaches to JPEG fragmentation detection, important for recovering corrupted or partially deleted images. Another investigates the accuracy of geolocation metadata in smartphone images, crucial for establishing location evidence. The third provides a targeted overview of mobile device forensics, offering comprehensive insights into current methodologies and challenges in examining smartphones and tablets for digital evidence.

Finally, the miscellaneous category covers diverse digital forensic topics span malware detection, cryptocurrency forensics, temporal analysis, and audio forensics. One paper presents WARNE, a specialized tool for collecting evidence of stalkerware, addressing the growing concern of privacy violations through malicious surveillance software. Another explores cryptocurrency mining detection through traffic forensics, providing methods to identify unauthorized mining activities on networks. A third paper examines timestamp accuracy and time anchors across different operating systems, crucial for establishing digital evidence timelines. The final work enhances speaker identification in criminal investigations through advanced clusterization and rank-based scoring techniques, improving the reliability of audio evidence analysis.

2 Povzetki / Summaries

2.1 WARNE: Orodje za zbiranje dokazov o zalezovalski programski opremi

Intimnapartnersko nasilje pogosto vključuje uporabo zalezovalskih aplikacij, katere storilci namestijo na žrtvino mobilno napravo in s tem posegajo v njeno zasebnost. V članku opišemo delovanje programske opreme WARNE. To je forenzično orodje za prepoznavo takšnih aplikacij in zbiranje dokazov o identiteti zalezovalca. Orodje je uspešno odkrilo pomanjkljivosti pri 20 od 30 testiranih zalezovalskih aplikacijah in s tem pokazalo velik potencial za pomoč žrtvam in preiskovalcem.

Ključne besede: zalezovalske aplikacije, digitalna forenzika, varnost mobilnih naprav, intimnapartnersko nasilje, IPN

2.2 How to detect cryptocurrency miners? By traffic forensics!

The paper by Veselý and Žádník presents a hybrid approach for detecting cryptocurrency mining traffic in large-scale network environments. Using flow-level features extracted from data collected in a national academic network, a labeled dataset was constructed through a semi-automated annotation process involving both passive analysis and active probing, enhanced by leveraging the sMaSheD catalog of known mining pool addresses. Two classifiers were developed and evaluated: a manually tuned rule-based classifier and a supervised machine learning model. The manual classifier serves as an interpretable baseline, relying on heuristic feature thresholds, while the machine learning model demonstrates superior accuracy and generalization. Evaluation results show a significant reduction in false positives and false negatives with the ML-based detector. A two-step detection schema, combining statistical classification with behavioral confirmation via active probes, further enhances precision and reliability. The findings confirm that mining traffic exhibits distinct, detectable characteristics, and that combining machine learning with domain knowledge, along with up-to-date mining pool information, provides an effective solution for real-world deployment.

Keywords: Traffic forensics, Cryptocurrency, Cryptojacking, Mining pool

2.3 Was the clock correct? Timestamps and time anchors across operating systems

Accurate event reconstruction in digital forensics depends on the integrity of system-generated timestamps, yet operating-system clocks are often misaligned with real time because of drift, battery failure or deliberate tampering. The authors formalise this problem by introducing time anchors: artefacts that embed both a local timestamp supplied by the device and an external timestamp issued by an independent source. By comparing the two we can quantify clock skew at the moment an artefact is created and label events as anchoring (verifiable) or non-anchoring. The authors also define time anomalies — inconsistent skews or impossible temporal sequences — that signal manipulation or hardware faults. Two controlled Windows 10 experiments validate the approach. First, Google-search artefacts (browser history, cache entries and server logs) were collected with the system clock set correctly and then shifted three hours behind real time. The skew measured in each anchoring artefact precisely matched the imposed offset and exposed the unreliability of non-anchoring events. A second experiment repeated the procedure for local file creation, demonstrating how non-anchoring events can be corrected once neighbouring anchors establish the true skew. Our addition is a review of methods of timestamp manipulations on Linux system, in which we review how to spot a faked timestamp using the touch command, extended file attributes, and journal history.

Keywords: timestamp, time shift, time anchor, OS forensics

2.4 Enhancing speaker identification in criminal investigations through clusterization and rank-based scoring

Speaker identification is a crucial task in criminal investigations, but analyzing large volumes of noisy audio and comparing them with extensive databases often leads to a high rate of false positives when using standard methods. This paper presents an approach to improve the identification of forensic speakers by combining state-of-the-art speaker embeddings with clustering and a rank-based scoring mechanism. The method extracts embeddings using an ECAPA-TDNN model, groups audio segments using HDBSCAN, and applies a rank-adjusted scoring system. Experiments demonstrate that this approach significantly reduces the number of potential candidates requiring manual review compared to baseline methods, making the investigation process more efficient and manageable in real-world forensic scenarios.

Keywords: Digital forensic, Speaker identification, Clusterization, Audio analysis, Scoring

2.5 Oblačna forenzična analiza Amazon iRobot Roomba sesalca

Članek predstavlja možnost uporabe avtonomnih naprav, kot je Amazonov-ova iRobot Roomba, za preiskovanje zločinov. Kot mnogo drugo IoT naprav tudi Roomba zbira in hrani kup podatkov o uporabi, lokaciji, sobah in pohoštvu v stanovanju, časovne žige, metapodatke in drugo. Ti podatki bi lahko bili zelo koristni pri raziskavi zločinov, saj nam bi lahko podali dragocene informacije o poteku dogodkov. Avtorji so najprej analizirali vse API dostopne točke, preko katerih Roomba komunicira s strežnikom, nato pa jih uporabili za razvoj nove odprtokodne aplikacije PyRoomba, ki izboljša uradno Roombino aplikacijo, tako da doda več podrobnosti in izriše bolj natančne 2D zemljevide prostorov, v katerih je sesalec aktiven. Aplikacija je bila nato testirana v več različnih prostorih z različno velikostjo in številom objektov, nato pa je bila še simulirana scena umora, kjer so na tla položili truplo in več nožev, ter testirali, ali jih bo Roomba zaznala. Preizkusi so se izkazali za uspešne, saj je sesalec identificiral dodatne ovire, ki se povezuje z umorom.

Ključne besede: Digitalna forenzika, Forenzika interneta stvari, Oblačna forenzika, Amazon iRobot Roomba

2.6 Hit and run: Forensic vehicle event reconstruction through driver-based cloud data from Progressive’s snapshot application

This paper provides a summary and critical analysis of a pioneering study by Onik et al. (2024) on the forensic analysis of the Snapshot application from Progressive Insurance. Driving Insurance Applications (DIAs) gather extensive data on driver behavior, yet their cloud-based data repositories remain a largely untapped resource for forensic investigations. The original study developed PyShot—an open-source Python tool—to extract granular data from the Progressive cloud, much of which is inaccessible via the standard user interface. It investigated the accuracy of Snapshot’s location and speed data, its resilience against GPS spoofing, and its ability to provide detailed event information for reconstructing a simulated hit-and-run scenario. This seminar paper evaluates the methodology and findings of that work, contextualizes it within the broader field of vehicle and cloud forensics, and discusses its implications. The findings confirm that telematics cloud data offers a robust and reliable new avenue for investigating traffic incidents and crimes, even when faced with user tampering.

Keywords: Driving insurance application (DIA), Vehicle forensics, Digital forensics, Snapshot, Driving patterns, Cloud forensics

2.7 Grand Theft API: A Forensic Analysis Of Vehicle Cloud Data

Modern cars collect and send meaningful amount of data to manufacturer cloud systems. This situation creates new possibilities for modern digital forensic investigations. In this paper, we look at how investigators can get vehicle cloud data directly through manufacturer APIs, using login details taken from suspects’ mobile devices. We explain the basics of vehicle forensics, look at current methods for getting vehicle cloud data, and show how API-based methods work in practice. After analyzing 23 different vehicle apps and manually testing with Mercedes-Benz and BMW cars, we found that a significant amount of vehicle data can be accessed through these APIs. This includes current car status, location, and personal info. Our research shows that using APIs to get vehicle cloud data gives valuable information for forensic investigations. This approach could change how evidence is collected from modern vehicles. It represents a step forward in digital vehicle forensics and allows access to both real-time and historical vehicle data while following proper forensic principles.

Keywords: Vehicle Forensics, API-based Acquisition, Digital Forensics, Telemetry Data, Cloud Forensics

2.8 Applying digital stratigraphy to the problem of recycled storage media

This report provides an overview of a study by Schneider et al. (2024) that applied digital stratigraphy to recycled storage media in a forensic context. The main goal of that research was to determine whether this technique can distinguish older residual data from new data on a reused device, especially when such fragments are found in unallocated space without metadata. The methodology relies on the concept of digital stratigraphy – adapted from geology – to analyze how file fragments are layered on disk over time. Schneider et al. implemented an automated file system simulation framework that systematically creates, modifies, and deletes files using real file system drivers, generating realistic layers of data for analysis. Experiments showed that examining these data layers and observing file system behavior can reveal stratigraphic markers indicating a fragment’s likely origin. These markers help investigators infer whether a recovered file fragment originated from the device’s previous usage or the current user’s activities. While the digital stratigraphy approach cannot definitively attribute every fragment, it significantly improves data provenance assessment on recycled storage media when conventional file metadata is absent.

Keywords: Digital stratigraphy, File System Upper Bound (FSUB), Data layering, Recycled Storage Media, Forensic analysis

2.9 Forenzični vidiki zloženih datotečnih sistemov

Članek obravnava izzive forenzične analize zloženih datotečnih sistemov, ki so postali bolj kompleksni zaradi večslojnih arhitektur in naprednih funkcij, kot je npr. šifriranje. Tradicionalne forenzične tehnike niso primerne za te sisteme, saj zahtevajo analizo tako zgornjih kot spodnjih slojev datotek. Predlagan je nov model, ki vključuje fazo za prepoznavanje indikatorjev in analizo časovnih žigov. Za preverjanje učinkovitosti modela so avtorji izvedli študijo primerov na sistemih MooseFS, GlusterFS in eCryptfs, pri čemer so prepoznali ključne forenzične izzive teh sistemov, kot sta fragmentacija in prisotnost skritih podatkov. Model omogoča izboljšanje forenzične preiskave z obnovo izbranih datotek in celovitejšo analizo datotečnih sistemov.

Ključne besede: Zloženi datotečni sistemi, Forenzična analiza, Digitalna forenzika

2.10 Forenzična analiza naprave Steam Deck: Predstavitev in razširitev članka Well Played, Suspect!

V članku analiziramo področje digitalne forenzike na igralnih konzolah in igrah. Osredotočamo se predvsem na članek Well Played, Suspect!, ki je bil osnova za našo raziskavo. V prvem delu predstavimo postopek diferencialne forenzike in opišemo napravo Steam Deck ter operacijski sistem SteamOS, ki sta glavno področje naše raziskave. V drugem delu naredimo pregled področja, kjer opišemo sorodne članke ter njihove ugotovitve, poleg tega še podrobneje predstavimo članek, ki je osnova za naše delo. Nato opišemo praktični del našega članka, kjer pridobimo slike fizične in virtualne naprave in na njih preverimo delovanje dodatka oziroma razširitve za orodje Autopsy, ki so jo napisali avtorji predhodnega članka. Razširitev tudi popravimo. Na koncu predstavimo rezultate in primerjamo delovanje razširitve na obeh napravah.

Ključne besede: Digitalna forenzika, Diferencialna forenzika, SteamOS, Steam Deck, Nintendo 3DS, PlayStation 4, Autopsy

2.11 Beyond the 3DS: A Cross-Platform Study Inspired by Nintendo 3DS Residual Data Analysis

This paper presents an extended study inspired by a 2024 forensic case-study of secondhand Nintendo 3DS devices. The original work analyzed residual personal and technical data retrievable from these consoles and highlighted their potential forensic value. Building upon that research, we provide a broader contextual analysis by reviewing related studies on other gaming platforms such as the Xbox and PlayStation, offering a comparative perspective on cross-platform forensic readiness and privacy implications. We further investigate the limitations of existing forensic tools and propose enhancements to extract additional artifact types, such as deleted files and detailed game logs. Finally, we simulate a hypothetical forensic investigation involving a game console, analyzing how the extracted evidence might be handled in court with regard to legal admissibility, chain of custody, and ethical boundaries. Our findings highlight the role of game consoles in modern digital forensics and the need for standardized approaches to data recovery and privacy management across different hardware platforms.

Keywords: Digital Forensics, Game Consoles, Nintendo 3DS, Xbox, PlayStation, Cross-Platform Analysis, Data Recovery, Residual Data, NAND Forensics, Ethical Forensics, Tool Development

2.12 Analyzing Deterministic and Heuristic Approaches to JPEG Fragmentation Detection

JPEG files are commonly used because of their efficient compression. However, they are prone to fragmentation, which complicates data recovery in forensic. This paper reviews a newly proposed deterministic algorithm designed to detect fragmentation points in JPEG files by analyzing inconsistencies in the bitstream. Unlike traditional methods, this one uses Huffman code lookup errors and quantization array overflows to identify fragmentation. The algorithm was tested on a custom dataset of real-world images. The research contributes an open-source tool (algorithm) and a real-world dataset to support digital forensics.

Keywords: JPEG fragmentation, Digital forensics, Data recovery, Fragmentation point detection, Huffman code analysis, Quantization table overflow, Bitstream inconsistency, File carving

2.13 Natančnost geolokacijskih metapodatkov slik pametnih telefonov

Geolokacijski podatki, še posebej tisti v EXIF metapodatkih fotografij pametnih telefonov, predstavljajo pomemben dokaz v digitalni forenziki. Ključnega pomena za njihovo uporabo v pravnih postopkih je zanesljivost in natančnost teh podatkov. Ta članek obravnava natančnost geotagiranja fotografij, posnetih s pametnimi telefoni, z izvedbo empirične študije, ki temelji na metodologiji Ryserja in sod. (2024). Preučujemo vpliv različnih dejavnikov na natančnost lokacijskih podatkov, vključno z uporabljenimi viri lokacije (GNSS, Wi-Fi, mobilno omrežje), okoljem (urbano proti podeželskemu) in generacijo mobilnega omrežja (2G, 3G, 4G). Cilj raziskave je oceniti, v kolikšni meri ti dejavniki vplivajo na napake pri določitvi lokacije ter osvetliti zanesljivost lokacijskih metapodatkov fotografij kot forenzičnega dokaza.

Ključne besede: geolokacija, forenzika mobilnih naprav, zanesljivost metapodatkov, natančnost lociranja naprave, analiza napak pri pozicioniranju, statistična analiza

2.14 Mobile Device Forensics: A Targeted Overview

We summarize a findings from several papers, with the main focus being article Nyon unchained: Forensic analysis of bosch's ebike board computere, where a Bosch Nyon device is used to obtain user information. Besides the hardware components, the methodology used and the analysis is described. Additionally we look into papers linked to the main one. One of them explores the possibilities of using cars and their infotainment systems as sources of digital evidence, where a general approach is also outlined. Furthermore, we give attention to forensics of Internet of Things (IoT) devices, mainly the Amazon Echo Dot version 2. Lastly, we look into acquiring data from backups of smartphones.

Keywords: Digital forensics, e-bike, car, smartphone, Internet of Things

3 Razno / Miscellaneous

WARNE: Orodje za zbiranje dokazov o zalezovalski programski opremi

Dominik Hrovat
Fakulteta za računalništvo in
informatiko
Ljubljana, Slovenija

Milka Gruber
Fakulteta za računalništvo in
informatiko
Ljubljana, Slovenija

Timen Bobnar
Fakulteta za računalništvo in
informatiko
Ljubljana, Slovenija

Povzetek

Intimnopartnersko nasilje pogosto vključuje uporabo zalezovalskih aplikacij, katere storilci namestijo na žrtvino mobilno napravo in s tem posegajo v njeno zasebnost. V članku opišemo delovanje programske opreme WARNE. To je forenzično orodje za prepoznavo takšnih aplikacij in zbiranje dokazov o identiteti zalezovalca. Orodje je uspešno odkrilo pomanjkljivosti pri 20 od 30 testiranih zalezovalskih aplikacijah in s tem pokazalo velik potencial za pomoč žrtvam in preiskovalcem.

Ključne besede

zalezovalske aplikacije, digitalna forenzika, varnost mobilnih naprav, intimnopartnersko nasilje, IPN

1 Uvod

Intimnopartnersko nasilje (IPN) je bilo zaznano s strani Svetovne zdravstvene organizacije kot globalen problem, ki prizadane osebe vseh starosti, spolov ter družbenih ozadij. V ZDA je žrtev IPN kar vsaka tretja ženska in vsak šesti moški. Žrtve lahko čutijo daljnoročne posledice na področju zdravja, osebnih financ in širšega družbenega življenja. Intimnopartnersko nasilje lahko zajema vse od fizičnega do psihološkega nasilja znotraj partnerskega odnosa.

Vse več žrtev IPN poroča, da so bile s strani partnerja nadzorovane s pomočjo t.i. zalezovalskih aplikacij (angl. stalkerware). Mobilne naprave so bogat vir informacij, kar storilci izkoriščajo za nadzorovanje žrtvinega brskanja po spletu, lokacije, izmenjave sporočil, slik in drugih virov informacij. V grobem so zalezovalske aplikacije mobilne aplikacije, ki omogočajo prikrit oddaljen nadzor nad mobilno napravo.

2 Pregled področja

V članku [3] iz leta 2018 so avtorji izvedli študijo, v kateri so analizirali kako storilci v IPN izkoriščajo tehnologijo za ustrahovanje, izsiljevanje, nadzorovanje in impersonacijo svojih žrtev. Študija, ki je temeljila na sodelovanju 89 žrtev, je razkrila, da storilci pogosto uporabljajo preproste metode, kot so dostop do telefona prek lastništva naprav, prisila nad žrtvijo, da deli svoja gesla in fizični nadzor naprav. Storilci zlorabljajo tudi legitimne aplikacije za starševski nadzor. Avtorji so opazili, da je večina teh metod nesofisticiranih z vidika informacijske varnosti, ter da za zlorabo storilci ne potrebujejo posebnega tehničnega znanja.

Delo [1] iz leta 2018 obravnava dostopnost zalezovalskih aplikacij na spletu. Glavna hipoteza je, da večina storilcev najde zalezovalske aplikacije na Google Play Store ali Apple's App Store. Avtorji so opravili ročno in delno avtomatizirano analizo Googlovih iskanj in sorodnih predlogov. Našli so obsežno število spletnih strani, katerih namen je, da pomagajo storilcem namesiti zalezovalsko aplikacijo

in jim svetovati o uporabi le-te. Prav tako so raziskovalci v komentarjih uporabnikov v Google Play, s pomočjo iskanja prek ključnih besed povezanih z IPN, odkrili 103 komentarjev, ki izrecno omenjajo uporabo legitimne aplikacije za namene zalezovanja partnerja. Raziskovalci so analizirali tudi oglaševalske prakse zalezovalskih aplikacij na uradnih straneh ter raznih blogih in forumih. Uradne spletne strani aplikacij sicer trdijo, da je njihova uporaba namenjena starševskemu nadzoru. Določeni ponudniki pa aktivno oglašujejo in spodbujajo zlorabo s skritimi oglasi ali preko podpore strankam.

Raziskovalci so v delu [4] obravnavali monetizacijo zalezovalne programske opreme na mobilnih platformah. Z analizo 6432 aplikacij so ugotovili, da 99 odstotkov zalezovalskih aplikacij, ki uporabljajo oglase, izkorišča Google AdMob, medtem ko se plačilni mehanizmi razlikujejo od tradicionalnih vgrajenih nakupov do kriptovalut. Sprememba Googlovih pogojev uporabe leta 2020, ki je prepovedala zalezovalske aplikacije in je zahtevala, da morajo biti aplikacije, ki omogočajo nadzor mobilne naprave, bolj transparentne, je zgolj povzročila premik k drugim plačilnim načinom, aplikacije pa še zmeraj kršijo pogoje uporabe.

Študije so se v preteklosti usmerile tudi v zaznavanje obstoječe programske opreme v povezavi z nasiljem v partnerskih odnosih [2]. Raziskovalci so zaznali prisotnost številnih aplikacij z dvojno namembnostjo (angl. dual-use), ki na videz izgledajo nedolžne (npr. kot kalkulator), v resnici pa zbirajo podatke o oškodovani osebi in pa legitimne aplikacije za nadzor otrok, ki se zlorabijo za zalezovanje. Te aplikacije pogosto vsebujejo številne varnostne ranljivosti, kot so:

- nezaščitene podatkovne baze in
- slabo implementirano šifriranje.

Analizirali so tudi obstoječa komercialno dostopna orodja za zaznavanje take programske opreme (angl. anti-spyware tools), vendar so ugotovili, da večina teh ne zazna aplikacij z dvojno namembnostjo. V članku [5] so avtorji raziskali mehanizme zalezovalskih aplikacij na Androidu in s tem podali podroben opis njihovega delovanja. Sodobne tehnike za zaznavo zalezovalskih aplikacij temeljijo na analizi dejavnosti in preverjanju omrežnega prometa.

V nasprotju s temi pristopi pa so avtorji osrednjega obravnavanega članka [6] preko analize zalezovalskih aplikacij in njihovih spletnih storitev, razvili orodje WARNE, s katerim lahko neposredno identificirajo storilce in podjetja, ki te aplikacije razvijajo. Njihov cilj je bil razviti forenzično orodje za zbiranje dokazov, ki bo koristno v pravnih postopkih.

3 Pregled aplikacij in izbor

V tem poglavju si ogledamo, kako na splošno delujejo takšne aplikacije, ter predstavimo izbor aplikacij, na katerih so raziskovalci v članku [6] testirali razvito orodje WARNE.

3.1 Pregled delovanja

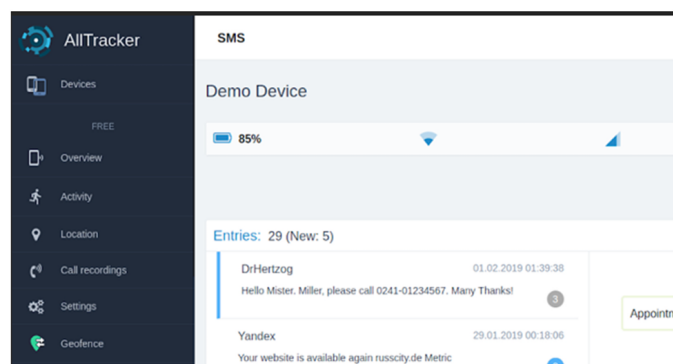
Za uspešen nadzor oziroma pridobivanje podatkov z mobilne naprave partnerja (žrtve) se mora storilec najprej registrirati na spletni strani, ki omogoča to dejavnost. Nekatere spletne strani so plačljive, vendar jih je veliko tudi brezplačnih.

Ko storilec opravi registracijo, mora pridobiti fizični dostop do telefonske naprave žrtve. Na telefon žrtve naloži aplikacijo, ki na videz izgleda nedolžna (npr. kalkulator), med nastavitvijo pa aplikaciji dodeli posebna dovoljenja za dostop do zasebnih informacij. Ko storilec aplikacijo namesti in se vanjo prijavi, napravo vrne žrtvi. Žrtev napravo uporablja normalno, medtem ko se storilec lahko kadarkoli poveže na napravo preko spletne strani in pridobi veliko zasebnih informacij.

3.2 Aplikacije za analizo

Avtorji so v obravnavanem delu poiskali številne aplikacije, ki omogočajo to dejavnost. Poudarjajo, da je težko najti celovit seznam takih aplikacij. Dodaten problem pri zbiranju aplikacij je dejstvo, da veliko aplikacij, omenjenih v različnih člankih, ni več aktivnih. Številne takšne aplikacije so spremenile domeno, kar otežuje iskanje.

Avtorji so uspeli najti 96 aplikacij, ki že v opisu vsebujejo izraze kot so "spouse", "husband" ali "wife" v kombinaciji z "spy" ali "cheating". Odkrili so tudi 80 spletnih strani, preko katerih je mogoče pridobiti APK datoteko (to je format datotek, ki jih uporablja sistem Android za mobilne aplikacije). Izkazalo se je, da le 50 od teh strani vsebuje različne APK datoteke, medtem ko ostale uporabljajo isto aplikacijo. Veliko spletnih strani svojo dejavnost prikriva s tem, da se promovirajo kot aplikacije za nadzor otrok. Te aplikacije so avtorji analizirali na podlagi informacij, ki so tekom zlorabe posredovane storilcem. Raziskali so tudi, kako iz informacij na spletni strani pridobiti podatke o storilcu. Na podlagi teh informacij so v nadaljevanju testirali in analizirali, kako aplikacije delujejo in kakšne napake vsebujejo, da bi jih lahko izkoristili za pridobivanje informacij o storilcih ali razvijalcih takšnih aplikacij.



Slika 1: Zajem zaslona nadzorne plošče povezane z aplikacijo AllTracker. Storilec ima dostop do žrtvine lokacije, aktivnosti in posnetkov klicev ter druge komunikacije.

Med analizo članka smo preverili tudi, ali je katera od aplikacij še vedno dostopna. Izkazalo se je, da ni.

4 Identifikacija ranljivosti v stalkerware aplikacijah za zbiranje dokazov

V tem poglavju opišemo, kako so avtorji izvajali analizo strani in aplikacij za odkrivanje možnih varnostnih pomanjkljivosti. Primer spletne strani, ki omogoča brskanje po žrtvinih podatkih lahko vidimo na sliki 1.

Na podlagi analize so avtorji odkrili 5 pomanjkljivosti, ki so prisotne v večini zalezovalske opreme. Od teh 5 pomanjkljivosti so avtorji izkoristili 2 za pridobivanje dokazov.

4.1 Priprave za analizo

Za analizo so avtorji uporabili telefon Google Pixel 3 z operacijskim sistemom Android 12 ter virtualni stroj z Androidom 10. Obe napravi sta bili "rootani", kar pomeni, da so imeli večji vpogled v delovanje naprave in dodatne zmogljivosti, kot je Android Debug Bridge (ADB), ki pomaga pri analizi in nadzoru naprave ter omogoča namestitvev tudi potencialno škodljivih aplikacij.

Avtorji so APK datoteke aplikacij nalagali neposredno na telefon, kadar je bilo to mogoče. V nekaterih primerih je spletna stran vodila do namestitvenega programa. V tem primeru so avtorji uporabili ADB za dostop do lokalnih datotek in tako pridobili APK datoteko.

Za analizo so avtorji uporabili dve orodji:

- **Frida:** Za prestrezanje Java funkcij, sledenje procesov in zbiranje informacij in
- **Burp Suite:** Proxy orodje za prestrezanje in spreminjanje HTTPS prometa med ogroženo napravo in strežnikom stalkerware aplikacije.

4.2 Medmrežno skriptiranje (XSS)

Avtorji so preverili, če so spletne nadzorne plošče ranljive za XSS napad (angl. Cross-site scripting attack). To je napad, pri katerem napadalec v spletno stran vstavi zlonamerno kodo, ki se potem izvede na napravi osebe, ki brska po tej strani.

V našem primeru to pomeni, da se koda skriva v datoteko (npr. ime kontakta ali SMS sporočilo) na žrtvinem telefonu. Ta datoteka se nato prikaže na spletni strani za nadzor. Ko storilec odpre datoteko, se koda izvede in razkrije informacije o njegovi napravi ali računu.

Ključna prednost te metode je, da ko se koda enkrat izvede, ostane shranjena na strežniku in ostane aktivna tudi, če jo izbrisemo iz telefona.

4.3 Neomejeno nalaganje datotek (Unrestricted File Upload)

Stalkerware aplikacije pogosto samodejno pošiljajo datoteke iz žrtvinega telefona na svoj strežnik. Avtorji so ugotovili, da 22 preverjenih aplikacij lahko dostopa do notranjega pomnilnika naprave. Ugotovili so tudi, da aplikacija ne preverja vsebine datotek, preden jih naloži na strežnik.

Avtorji to izkoristijo tako, da na žrtveni telefon namestijo posebej pripravljene datoteke, ki se nato aktivirajo, ko jih storilec prenese na svojo napravo. Ta postopek je zelo podoben postopku medmrežnega skriptiranja.

4.4 Okvarjena avtentikacija in nadzor dostopa (Broken Authentication and Access Control)

Večina stalkerware aplikacij uporablja centralizirane platforme za shranjevanje podatkov od žrtve. Avtorji so preverili, ali te platforme vsebujejo varnostne pomanjkljivosti v avtentikaciji in nadzoru dostopa.

Raziskovalci so najprej ustvarili dva računa, enega za storilca in enega za preiskovalca. Potem so na testno (žrtvino) mobilno napravo namestili stalkerware aplikacijo.

Pognali so orodje Auth Analyzer, ki prepozna napake v avtorizaciji, tako da avtomatsko ponavlja zahteve. Medtem so raziskovalci brskali po nadzorni plošči. Ranljivost so prepoznali na sledeč način - če strežnik sprejme zahteve preiskovalca kot legitimne (enak odziv kot za storilca), potem obstaja ranljivost v nadzoru dostopa.

Podobno so uporabili orodje Auth Analyzer za ponavljanje zahtevkov s praznimi sejami ali neveljavnimi žetoni. Če je strežnik uspešno odgovoril na tak zahtevek, je avtentikacija okvarjena.

Opazili so, da so za avtentikacijo in upravljanje uporabnikove seje pogosto uporabljeni žetoni JSON Web Tokens (JWT). Preverili so, če strežnik pravilno preverja podpise JWT žetonov. Uporabili so odprtokodno orodje za testiranje napak v podpisih (npr. sprejem nepodpisanih žetonov).

Zato, da se podatki prenesejo iz žrtvine mobilne naprave na strežnik, ki ga uporablja zlonamerna aplikacija, mora strežnik napravo identificirati, overiti in shraniti prenesene informacije. Avtorji so s prestrežanjem prometa (Burp Suite) analizirali, kako se naprava avtentificira pri nalaganju podatkov. Če strežnik ne zahteva pravilne avtentikacije, lahko preiskovalci pošiljajo podatke v imenu žrtve brez dostopa do naprave.

Te pomanjkljivosti omogočajo prevzem računov, dostop do tujih podatkov in ponarejanje podatkov.

4.5 Nezaščitena spletna shramba

Stalkerware aplikacije pogosto ločeno shranjujejo besedilne podatke (npr. SMS) in multimedijske podatke (npr. slike, video posnetke, posnetke zaslonov in zvočne posnetke). Zajeti besedilni podatki so običajno neposredno prikazani na spletni strani, medtem ko za dostop do multimedijskih datotek potrebujemo poseben URL.

V tem delu si ogledamo, kako dostopati preko URL naslovov. Avtorji so to poskušali na več načinov:

- Pošiljanje curl zahtevkov brez avtentikacijskih žetonov, s čimer so preverili, ali je shramba sploh zaščitena.
- Z ugibanjem URL naslovov - na podlagi znanih URL naslovov so konstruirali nove in tako ugotavljali strukturo URL naslovov.
- Preverjanje, ali so podatki še vedno dostopni preko URL-ja po izbrisu računa.

Avtorji so ugotovili, da številne spletne shrambe imajo vse tri probleme: dostop brez avtentikacije, ranljiv zapis URL-ja in trajnost podatkov. Na ta način lahko preiskovalci lažje dostopajo do podatkov.

4.6 Nezaščitena lokalna shramba podatkov

V tem delu si ogledamo podatke, ki se shranjujejo lokalno (kontakti, SMS sporočila, zgodovina tipkanja, dejavnost aplikacij itd.).

Avtorji so ugotovili, da se te informacije običajno shranjujejo v preprosti SQLite podatkovni bazi na žrtvini napravi. Z root dostopom lahko enostavno dostopamo do teh informacij. Na ta način lahko iz naprave žrtve pridobimo povezavo do storilčevega računa (e-pošta in geslo), API ključ in žetone za dostop do strežnika ter podatke o zbranih informacijah o žrtvi.

4.7 Napadi CSRF (Cross-Site Request Forgery)

V stalkerware aplikacijah je mogoče "prisiliti" storilca, da izvede dejanje, ki ga ni načrtoval, in sicer s pošiljanjem ukazov na napravo žrtve. Takemu napadu pravimo ponarejanje spletnih zahtev oz. napad z enim klikom (CSRF). To izkoristimo tako, da prepričamo storilca, da klikne na podtaknjeno povezavo.

Avtorji zaznavajo CSRF ranljivosti na vseh HTTP/s zahtevah, ki so sprožene med uporabo spletne strani za nadzor. Najprej preverijo, ali se v zahtevi nahajajo anti-CSRF žetoni. Nato preverijo, ali so ti žetoni povezani s storilčevim računom. Na ta način lahko potrdijo ali ovržejo prisotnost ranljivosti za CSRF.

5 Orodje WARNE

V članku avtorji predstavijo svoje orodje WARNE, ki omogoča zbiranje digitalnih forenzičnih dokazov v primeru zlorabe zalezovalskih aplikacij. WARNE izkorišča ranljivosti znotraj stalkerware aplikacij ter nezaščiteno lokalno shrambo podatkov na napravah. Orodje za prepoznavo zalezovalskih aplikacij uporablja znane indikatorje ogroženosti (angl. indicators of compromise oz. IoCs). Delovanje orodja je prikazano na sliki 2. Orodje na koncu napiše poročilo, ki vsebuje podatke, s katerimi je mogoče sklepati o identiteti napadalca (npr. IP naslove, e-poštne naslove, itd.). Delovanje orodja je razdeljeno na osem korakov.

- (1) **Začetna nastavitve:** Ob zagonu orodje WARNE omogoči uporabniku konfiguracijo analize prek treh ločenih nastavitvev. Uporabnik lahko poda: (a) poljuben XSS payload, ki kaže na že vnaprej nastavljen zunanji strežnik; (b) sumljivo ime paketa, ki morda ni vključeno na seznam znanih stalkerware aplikacij; ter (c) geslo oziroma skrivni ključ za varnostno kopijo naprave, če je analizirana naprava rootana ali šifrirana (tj. kadar so vse varnostne kopije zaščitene z geslom).
- (2) **Zaznavanje paketov:** WARNE pregleda seznam nameščenih aplikacij in označi sumljive: bodisi tiste, ki so na seznamu sumljivih aplikacij (AssoEchap's stalkerware Indicators of Compromise list - IOC), bodisi tiste, katerih ime ustreza ročno vnesenemu paketu.
- (3) **Zajem informacij o aplikacijah:** Za vsako zaznano sumljivo aplikacijo WARNE zajame podatke, kot so datum namestitve, zahtevana dovoljenja, poti do lokalne shrambe, itd. Prav tako se preveri, ali je potreben root dostop za pridobivanje podatkov.
- (4) **Ekstrakcija lokalnih datotek:** WARNE poskuša prenesti vse datoteke, ki so povezane s sumljivo aplikacijo. Postopek je odvisen od tega, ali je naprava rootana. Prenosa ni mogoče opraviti na ne-rootani napravi, če aplikacija ne dovoli kreiranja varnostne kopije.

- (5) **XSS injekcija in poslušalec:** WARNE omogoča povezavo z orodjem *XSShunter-express*. Uporabniški payload se avtomatsko injicira, orodje pa posluša sprožilce. V primeru aktivacije se zabeležijo podatki o sprožitvi in pripnejo poročilu.
- (6) **Zbiranje vsebine poročila:** Vsi zbrani podatki se shranijo v berljivi obliki, vključno s podatki o aplikacijah, imeni datotek in njihovo vsebino. Poleg tega se zabeležijo tudi neberljivi, prazni podatki, ali podatki, ki jih ne moremo odpreti.
- (7) **Razčlenjevanje poročila:** WARNE obdela zbrane informacije in uporabniku predstavi povzetek ugotovitev, ki jih uporabnik lahko podrobneje analizira.
- (8) **Odstranitev aplikacije:** Na koncu orodje uporabniku predlaga, da odstrani aplikacijo. Če je aplikacija nameščena kot skrbnik naprave, WARNE poda dodatna navodila. Pri odstranitvi moramo biti previdni, saj lahko odstranitev sproži odziv zlorabitelja.

6 WARNE - Implementacija

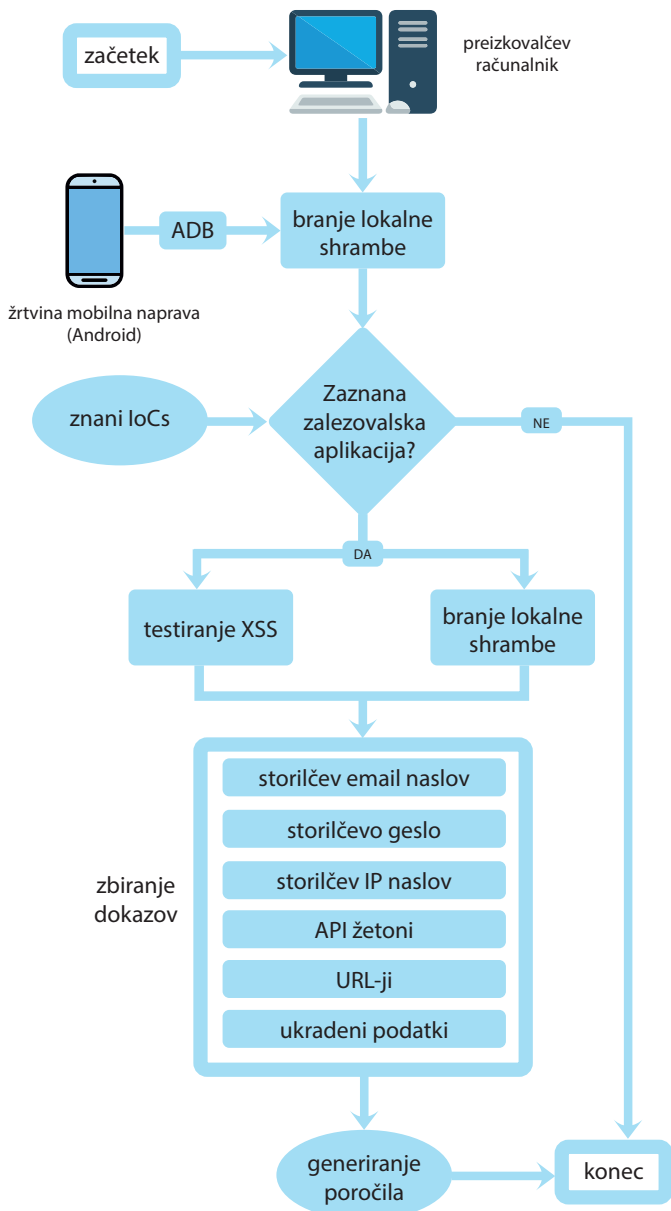
V tem poglavju bomo najprej opisali predpogoje, ki so potrebni za pravilno delovanje orodja WARNE.

6.1 Predpogoji in dodatna orodja

WARNE je narejen za operacijski sistem Linux in analizo telefonov z operacijskim sistemom Android. Koda za analizo je napisana v programskem jeziku Python in vključuje grafični uporabniški vmesnik, ki omogoča preglednost in enostavno uporabo. Ta vmesnik deluje z uporabo knjižnice Dash. Komunikacija s telefonom se izvaja preko orodja Android Debug Bridge (ADB), ki mora biti nameščen na gostiteljski napravi in omogočen na ciljni napravi. Omogočimo ga z aktiviranjem razvijalskih možnosti in dovoljevanjem USB debugging. ADB omogoča izvajanje ukazov na ciljni napravi in je ključnega pomena za večino funkcij, ki jih ponuja WARNE, vključno z zaznavanjem paketov, ekstrakcijo podatkov, odstranitvijo aplikacij in injiciranjem XSS napadov.

WARNE uporablja orodje Android Backup Extractor za ekstrakcijo podatkov iz šifriranih varnostnih kopij. Prav tako uporablja Android Asset Packaging Tool (AAPT) za izvajanje različnih poizvedb o paketih, nameščenih na ciljni napravi. AAPT omogoča izpis podatkov o določeni aplikaciji, kar je uporabno za pridobivanje informacij iz njenega manifesta, kot je običajno ime aplikacije, ki je prikazano na uporabniškem vmesniku naprave. Pri zagonu WARNE samodejno preveri prisotnost AAPT in ga prenese, če je to potrebno.

Obvladovanje XSS payloadov in sprožilcev poteka preko XSShunter-express, to je različica orodja XSShunter s prilagodljivimi nastavitvami in možnostjo samostojnega gostovanja. Da bi omogočili združljivost z WARNE, so avtorji orodje prilagodili tako, da ustvarja besedilna poročila o sproženih payloadih in jih povezuje z napravo, na kateri teče WARNE, preko SSH File System (SSHFS). Orodje nato redno preverja nastanek sprožilnih datotek in posodobi ustrezno besedilno poročilo aplikacije. Za identifikacijo izvora sprožilca se preveri URL, s katerega je sprožilec prišel, ter ga primerjamo z našim seznamom groženj IOC (Indicators of Compromise), da bi našli ustrezno ime paketa. V primeru, da ni najdena ustrezna aplikacija, se podatki o payloadu shranijo v ločeno datoteko skupaj z vsemi drugimi zabeleženimi sprožilci. Ker ta funkcionalnost zahteva precej kompleksno nastavljanje, njena uporaba ni nujna za



Slika 2: Prikaz delovnega toka orodja WARNE.

delovanje WARNE. Dodatno, ker je ta instanca XSShunter dostopna javno, morajo biti upoštevana standardna priporočila za zaščito strežnikov.

6.2 Implementacija funkcionalnosti WARNE

6.2.1 Zaznavanje stalkerware aplikacij. Ena izmed osnovnih funkcij orodja WARNE je identifikacija znanih stalkerware aplikacij. Orodje

za ta namen uporablja javno dostopen GitHub repozitorij, ki ga vzdržuje projekt AssoEchap. Ta repozitorij vsebuje seznam znanih paketnih imen sumljivih aplikacij.

WARNE pri zagonu pridobi seznam vseh aplikacij, nameščenih na ciljni napravi, in preveri morebitna ujemanja z znanimi imeni paketov. V primeru, da zazna potencialno stalkerware aplikacijo, zabeleži dodatne informacije, kot so datum namestitve ter vsa dovoljenja, ki jih je aplikacija zahtevala.

6.2.2 Dostop do lokalne shrambe. Dostop do vsebine notranje shrambe aplikacij na Android napravah običajno zahteva root dostop. WARNE pa zaobide to omejitev z uporabo sistema za varnostne kopije aplikacij, ki omogoča dostop do podatkov tudi na nerootanih napravah, če aplikacija ne preprečuje varnostnega kopiranja prek nastavitve `allowBackup=false` v manifestu.

Če aplikacija dovoljuje izdelavo varnostne kopije, WARNE sproži postopek ustvarjanja kopije za vsako sumljivo aplikacijo. Nato prenese kopijo na analizni sistem, kjer z uporabo posebnih orodij razčleni strukturo varnostne kopije. Vsaka datoteka v notranji shrambi aplikacije se prebere, SQLite podatkovne baze pa se razbijejo v pregledne tekstovne zapise z vsemi razpoložljivimi tabelami.

6.2.3 Generacija poročil. Orodje WARNE je zasnovano tako, da ustvari podrobno in lahko berljivo poročilo o izvedeni analizi. Končno poročilo nastane z združitvijo vseh zbranih podatkov v eno samo tekstovno datoteko. Po sestavi se poročilo dodatno obdela z uporabo regularnih izrazov, s katerimi se pridobijo ključni podatki. Med te informacije sodijo e-poštni naslovi, ki ne pripadajo primarnemu uporabniku naprave, imena gostiteljev in URL naslovi, pojavitve izrazov kot sta "uporabniško ime" in "geslo" ter prepoznavni Google API ključi ali žetoni.

7 Rezultati

Avtorji članka so orodje WARNE preizkusili na 30 zalezovalskih aplikacijah. Ugotovili so, da 26 aplikacij zbira SMS sporočila in dnevnik klicev, 27 aplikacij omogoča GPS sledenje in geoograjevanje (opozarjanje, ko žrtev zapusti določeno območje). To sta najpogostejši funkciji analiziranih aplikacij. Druge pomembne zmogljivosti vključujejo skrito snemanje s kamero (17) ali mikrofonom (13), keylogger, ki zbira vnose tipk in lahko storilcu posreduje žrtvina gesla (14), dostop do datotek (fotografij, videoposnetkov, dokumentov) (16) in klepetov v družbenih omrežjih (Facebook Messenger, Instagram, WhatsApp, Viber).

Zbrani podatki se pošiljajo storilcu na dva načina: V 29 od 30 primerov se podatki naložijo v spletno bazo podatkov, kjer si jih lahko storilec ogleda preko nadzorne plošče. V drugem primeru se podatki pošiljajo neposredno na storilčev e-poštni naslov.

7.1 Varnostne ranljivosti

7.1.1 Cross-site scripting (XSS). Raziskovalci so odkrili 20 aplikacij, katerih spletne nadzorne plošče niso izvajale validacije vnosov pred prikazom zbranih podatkov. Odkrili so, da se XSS lahko izvede preko SMS sporočil v 18 aplikacijah, kontaktnega imenika (16) ali imen datotek (11). XSS payload lahko tudi potrdi prisotnost zalezovalske aplikacije na telefonu, če na primer pošljemo SMS z vstavljenim payloadom. Opozoriti je potrebno, da lahko ta napad

opozori storilca, saj lahko ta na nadzorni plošči zazna sumljivo formatirana sporočila.

7.1.2 Pokvarjena avtentikacija. Raziskovalci so prepoznali 8 aplikacij, ki so ranljive za prevzem računa ali neavtorizirano pošiljanje ukazov zaradi pokvarjene avtentikacije. Dve aplikaciji uporabljata Google Identity Toolkit za preverjanje uporabnika in vodenje njegovega računa. Ta uporablja žeton, da identificira storilca in je shranjen na žrtvini napravi. Druga aplikacija uporablja Google Cloud Messaging. Ključ za GCM se nezaščiten nahaja na žrtvini napravi.

7.1.3 Nevarne metode shranjevanja. Raziskovalci so ugotovili, da 6 aplikacij shranjuje občutljive podatke (npr. e-poštni naslov, geslo) v nezavarovanih lokacijah. Od teh 3 aplikacije omogočajo neomejen dostop do slik preko statičnih URL-jev. To pomeni, da lahko v omejenem časovnem obdobju kdorkoli dostopa do teh slik. Ta pomanjkljivost predstavlja dodatno tveganje za žrtev, saj lahko do podatkov dostopajo tudi druge osebe poleg storilca.

Štiri od analiziranih aplikacij hranijo občutljive podatke o storilcu kar na žrtvini mobilni napravi. Podatki, kot so storilčev e-mail naslov, storilčeva registracijska številka, PIN številka aplikacije in celo storilčevo geslo se hranijo v notranjih datotekah na napravi.

Raziskovalci so pri treh aplikacijah našli podatkovno bazo, ki hrani podatke o storilcu, kar je lahko koristno v primeru, ko storilec izbriše svoj račun in odstrani aplikacijo.

7.1.4 Nepreverjeno nalaganje datotek. Nobena aplikacija ne preverja datotek, naloženih preko nadzorne plošče. To pomeni, da lahko pošljemo zlonamerne datoteke storilcu. Kdorkoli pozna žrtvino telefonsko številko, bi lahko poslal zlonamerne datoteke prek SMS-sporočila, ki bi se prenesle do storilca, brez da bi bile vsebovane na žrtvini mobilni napravi.

7.1.5 Cross-site request forgery (CSRF). Raziskovalci so ugotovili, da so 4 aplikacije ranljive za CSRF napade, kar omogoča spreminjanje storilčevih gesel ali preusmerjanje obvestil s storilčevega na druge e-poštne naslove.

8 Zaključek

Raziskovalci so razvili orodje za zbiranje informacij in dokazov o nameščenih zalezovalskih aplikacijah na žrtvini mobilni napravi. Analiza 30 aplikacij je razkrila številne varnostne pomanjkljivosti, ki jih je mogoče izkoristiti za identifikacijo storilcev. Orodje je bilo uspešno kar v 20 primerih in se je izkazalo kot potencialno koristno pri podpori žrtvam nasilja v partnerskih odnosih.

Menimo, da so raziskovalci s tem ponudili velik prispevek k temu dosedaj slabo razvitemu področju. Večina nasvetov, ki smo jih našli na spletu, žrtvam svetuje, kako odkriti zalezovalsko aplikacijo in jo odstraniti, s čimer se izgubijo potencialni dokazi proti storilcu. A vendar orodje žal (še) ni javno dostopno. To nam je onemogočilo eksperimentalni del, kjer smo želeli vsaj preizkusiti namestitev programske opreme WARNE in pregledati svoje osebne mobilne naprave. Analiza zalezovalskih aplikacij bi bila težka, saj bi zato potrebovali dodatne mobilne naprave, zato smo se v našem članku osredotili zgolj na teoretičen del.

Literatura

- [1] Rahul Chatterjee, Periwinkle Doerfler, Hadas Orgad, Sam Havron, Jackeline Palmer, Diana Freed, Karen Levy, Nicola Dell, Damon McCoy, and Thomas Ristenpart. 2018. The Spyware Used in Intimate Partner Violence. In *2018 IEEE Symposium on Security and Privacy (SP)*. 441–458. doi:10.1109/SP.2018.00061
- [2] Jonathan Dalman and Vincent Hantke. 2015. Commercial Spyware: Detecting the Undetectable. Black Hat USA 2015. <https://www.blackhat.com/docs/us-15/materials/us-15-Dalman-Commercial-Spyware-Detecting-The-Undetectable.pdf> Accessed: [4. 5. 2025].
- [3] Diana Freed, Jackeline Palmer, Diana Minchala, Karen Levy, Thomas Ristenpart, and Nicola Dell. 2018. “A Stalker’s Paradise”: How Intimate Partner Abusers Exploit Technology. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) (*CHI ’18*). Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3173574.3174241
- [4] Cassidy Gibson, Vanessa Frost, Katie Platt, Washington Garcia, Luis Vargas, Sara Rampazzi, Vincent Bindschaedler, Patrick Traynor, and Kevin Butler. 2022. Analyzing the Monetization Ecosystem of Stalkerware. *Proceedings on Privacy Enhancing Technologies* 2022 (10 2022), 105–119. doi:10.56553/popets-2022-0101
- [5] Enze Liu, Sumanth Rao, Sam Havron, Grant Ho, Stefan Savage, Geoffrey Voelker, and Damon McCoy. 2023. No Privacy Among Spies: Assessing the Functionality and Insecurity of Consumer Android Spyware Apps. *Proceedings on Privacy Enhancing Technologies* 2023 (01 2023), 207–224. doi:10.56553/popets-2023-0013
- [6] Philippe Mangeard, Bhaskar Tejaswi, Mohammad Mannan, and Amr Youssef. 2024. WARNE: A stalkerware evidence collection tool. *Forensic Science International: Digital Investigation* 48 (2024), 301677. doi:10.1016/j.fsidi.2023.301677 DFRWS EU 2024 - Selected Papers from the 11th Annual Digital Forensics Research Conference Europe.

How to detect cryptocurrency miners? By traffic forensics!

Vladimír Veselý, Martin Žádník

Igor Sitek

Univerza v Ljubljani, Fakulteta za računalništvo in
informatiko
Ljubljana, Slovenia
is85415@student.uni-lj.si

Mariana Dinis

Univerza v Ljubljani, Fakulteta za računalništvo in
informatiko
Ljubljana, Slovenia
mf21598@student.uni-lj.si

Telmo Sendino

Univerza v Ljubljani, Fakulteta za računalništvo in
informatiko
Ljubljana, Slovenia
ts72421@student.uni-lj.si

Gonçalo Cardoso

Univerza v Ljubljani, Fakulteta za računalništvo in
informatiko
Ljubljana, Slovenia
gr88022@student.uni-lj.si

ABSTRACT

The paper by Veselý and Žádník[13] presents a hybrid approach for detecting cryptocurrency mining traffic in large-scale network environments. Using flow-level features extracted from data collected in a national academic network, a labeled dataset was constructed through a semi-automated annotation process involving both passive analysis and active probing, enhanced by leveraging the sMaSheD catalog of known mining pool addresses. Two classifiers were developed and evaluated: a manually tuned rule-based classifier and a supervised machine learning model. The manual classifier serves as an interpretable baseline, relying on heuristic feature thresholds, while the machine learning model demonstrates superior accuracy and generalization. Evaluation results show a significant reduction in false positives and false negatives with the ML-based detector. A two-step detection schema, combining statistical classification with behavioral confirmation via active probes, further enhances precision and reliability. The findings confirm that mining traffic exhibits distinct, detectable characteristics, and that combining machine learning with domain knowledge, along with up-to-date mining pool information, provides an effective solution for real-world deployment.

KEYWORDS

Traffic forensics, Cryptocurrency, Cryptojacking, Mining pool

1 INTRODUCTION

In recent years, cryptocurrency in general, and Bitcoin in particular, has gained significant interest, causing Bitcoin's price to reach record highs. Cryptocurrency provides a decentralized currency whose trustworthiness is not backed up by a Central Bank (General Reserve) like the traditional bank system; the trust on cryptocurrencies is built on cryptographic algorithms used to verify transactions and fair emission of new currency units into circulation.

There are some controversies surrounding Bitcoin as several studies [11, 2] have targeted Bitcoin as the main currency of any digital black marketplace. As cryptocurrencies provide some grade of anonymity, they usually allow criminals to circumvent law enforcement agencies.

There is a vast variety of cryptocurrencies, the ones that refer to themselves as alternatives to Bitcoin are called "altcoins", for example, Litecoin or Ethereum. Tokens are also part of the cryptocurrency ecosystem, they represent a digital utility that relies on another coin blockchain for being accounted.

Bitcoin is a peer-to-peer (P2P) network with a distributed infrastructure of users and miners. Miners lend their computational power solving cryptographic puzzles to validate transactions and secure the network in exchange of a *economic revenue* (cryptocurrency).

However, this mining process has attracted unauthorized actors who deploy mining malware within corporate environments or any device connected to the Internet. This technique is known as *cryptojacking*, the victim is charged with the operational costs of mining (electricity bills, depleted computational resources, etc.), and the attacker only has profits obtained in the form of cryptocurrency.

In terms of cybersecurity, detecting illicit mining traffic is becoming a necessity, although this poses some challenges: protocol communications are often lightweight, encrypted, and carried over common ports (e.g., 80, 443), mimicking benign services.

This article examines two complementary detect strategies, traffic-predecessor forensics and a curated mini-catalog, demonstrating how their combined deployment enhances visibility into unauthorized mining activity.

2 PURPOSE OF CRYPTOCURRENCY MINING

2.1 Foundation of Decentralized Cryptocurrencies

Cryptocurrency mining forms the base of decentralized digital currencies such as Bitcoin, serving as the primary mechanism to introduce new units into circulation and to verify transactions [9, 13]. This process is fundamental to the operation of these systems, as it distributes control over the currency across a network of participants rather than relying on a central authority [9]. By engaging in mining, individuals and entities contribute their computational resources to validate transactions, thereby maintaining the integrity and security of the blockchain network, the distributed ledger that records all cryptocurrency transactions [9, 13]. This decentralized

nature ensures that no single entity can manipulate the transaction history or control the supply of the currency, this translates into trust among users who may not know or trust each other [9], they do not need to have this judgment. The very essence of a decentralized cryptocurrency falls on this distributed validation process, making mining a critical activity that leaves noticeable footprints on network traffic as participants communicate to validate and record transactions.

The increasing market value of Bitcoin and other cryptocurrencies has further incentivized mining as a potential source of revenue [13]. This economic incentive drives individuals and organizations to invest in specialized hardware and expend significant energy to participate in the mining process [9]. The competitive nature of mining, where participants race to solve complex computational problems, results in constant communication across the network as miners attempt to validate blocks of transactions and claim rewards. This continuous activity generates a substantial amount of network traffic, which can be analyzed to understand and potentially detect mining operations [10, 13].

2.2 Transaction Verification and Blockchain Security

A core function of cryptocurrency mining is to verify and add new transaction records to the blockchain. Miners utilize their computational power to solve intricate cryptographic puzzles, a process known as Proof-of-Work (PoW), to validate these transaction blocks. The first miner to successfully solve the puzzle earns the right to add the newly validated block of transactions to the blockchain. This addition makes the transactions permanent and impossible to tamper with, as each new block is cryptographically linked to the previous one, forming an immutable chain of records [9].

The difficulty of these cryptographic puzzles is not static, it adjusts dynamically based on the total computational power of the network to maintain a consistent rate of block creation. For example, in the Bitcoin network, the difficulty is adjusted to aim for a new block approximately every 10 minutes. This automatic adjustment ensures that the pace of transaction processing and the creation of new currency units remains relatively stable despite fluctuations in the number of miners participating in the network. Furthermore, the immense computational cost required to solve these puzzles acts as a robust security mechanism. To alter a past transaction block, a malicious actor would need to redo the work for that block and all subsequent blocks, a feat that would require a prohibitive amount of computing power, likely exceeding that of the entire legitimate network. This inherent security feature of the blockchain, backed by the mining process, is crucial for the trustworthiness and reliability of cryptocurrencies [9].

2.3 Cryptocurrency Generation and Rewards

As an incentive for contributing their computational resources to the network, miners receive rewards in the form of newly minted cryptocurrency, known as the block reward, along with any transaction fees included in the validated block [8]. This reward system is designed to encourage participation in the mining process, which is essential for the network's security and functionality. The initial block reward for Bitcoin, for instance, was 50 bitcoins, but this

reward is halved approximately every four years through a process called "halving". This diminishing block reward mechanism is designed to control the total supply of the cryptocurrency over time [9].

The transaction fees, mentioned above, are voluntarily included by users when sending cryptocurrency. These fees serve as an additional incentive for miners to prioritize certain transactions, particularly when the network is experiencing high traffic. As the block reward continues to decrease with each halving event, transaction fees are expected to become an increasingly significant source of income for miners, ensuring the long-term sustainability of the mining process. The economic motivation behind these rewards drives miners to actively engage with the network, seeking efficient connections to mining pools and striving to maximize their computational output. This pursuit of rewards often manifests in specific network communication patterns that can be identified and analyzed [13, 10].

2.4 Legitimate Mining vs. Cryptojacking

Cryptocurrency mining has evolved into a legitimate business operation for many, serving as a means to generate revenue through the validation of transactions and the acquisition of new cryptocurrency units. However, the same underlying process is also exploited in a malicious activity known as cryptojacking [7]. Cryptojacking involves the unauthorized use of compromised computer resources, such as CPUs, GPUs, or even entire networks, to mine cryptocurrency for profit without the owner's knowledge or consent. This can be achieved through various methods, including installing malware on the victim's machine or embedding mining scripts in websites that run in the user's browser [7].

The detection of cryptocurrency mining activity within a network is therefore crucial not only for understanding legitimate network resource utilization but also for identifying potential security threats such as malware infections or unauthorized exploitation of company resources [7, 1]. While the purpose differs significantly between legitimate mining and cryptojacking, the underlying technical processes and the resulting network traffic patterns can share similarities, making it a challenging task to distinguish between them. However, a thorough understanding of mining protocols and typical communication patterns with mining pools is essential for developing effective detection mechanisms [5, 12, 10].

3 CRYPTOCURRENCY MINING POOLS

3.1 The Necessity of Pooling Resources

As the overall computational power of cryptocurrency networks has grown exponentially, the difficulty of mining new blocks has increased significantly. This increase in difficulty has made it increasingly challenging and often unprofitable for individual miners with limited computational resources to mine blocks on their own. Solo mining has become highly reliant on chance, even with substantial investment in mining hardware [6, 8].

To overcome this challenge, most cryptocurrency miners join mining pools. These pools are collaborative groups of miners who combine their individual computational resources, or hash rate, to increase their collective probability of successfully mining a block. By pooling their hashing power, miners can work together

to solve the complex cryptographic puzzles required by the Proof-of-Work consensus mechanism more efficiently than they could individually. This collaborative approach allows individual miners to earn more consistent, although smaller, rewards compared to the highly variable and often infrequent rewards of solo mining. The prevalence of mining pools signifies that network traffic analysis aimed at detecting cryptocurrency mining should primarily focus on identifying communication patterns between individual miners within a network and these centralized pool servers [5, 12, 10].

3.2 Architectural Overview of Mining Pools

At the heart of a mining pool is a central server that acts as a coordinator for all participating miners. This pool server plays a crucial role in distributing the overall mining task into smaller, manageable work units that are then assigned to individual miners within the pool. As miners work on these individual tasks, they submit the results of their computations, often referred to as "shares," back to the pool server. The pool server aggregates these submitted shares from all participating miners.

Miners typically connect to the pool server using specialized mining software installed on their computers or dedicated mining hardware. This connection is usually established over the internet using specific mining protocols, which facilitate the exchange of information between the miner and the pool server. The architecture of a mining pool thus follows a client-server model, where individual miners act as clients, sending computational results and receiving new work assignments from the central pool server [6]. This client-server interaction generates network traffic that can be intercepted.

3.2.1 Work and Reward Distribution Strategies. Once a miner finds a valid share of the work previously distributed by the pool, they submit it back to the pool server. The pool server keeps track of the number of shares submitted by each miner, using this information to measure their contribution to the overall effort of the pool. When one of the miners in the pool eventually finds a full solution that is accepted by the cryptocurrency network, the pool receives the block reward and any associated transaction fees. This reward is then distributed among the participating miners. The distribution is typically based on the amount of computational work each miner contributed to the pool's effort, as measured by the number of valid shares they submitted. The process of distributing work and aggregating results, particularly the frequent submission of shares from miners to the pool server, needs the exchange of network packets that can be monitored.

Mining pools employ various reward distribution schemes to ensure fairness and incentivize continued participation. Some common schemes include Pay-Per-Share (PPS), where miners are paid a fixed amount for each valid share they submit, regardless of whether the pool actually finds a block. Another common method is the Proportional scheme, where miners are paid out proportionally to the number of shares they contributed during the round when the pool found a block [6]. Other schemes like Score-based systems also exist.

3.3 Types of Mining Pools

Mining pools are often categorized based on the specific cryptocurrency they facilitate mining for. For instance, there are dedicated Bitcoin mining pools, Ethereum mining pools, Monero mining pools, and so on. The choice of cryptocurrency often dictates the specific mining algorithm used (e.g., SHA-256 for Bitcoin, Ethash for Ethereum, CryptoNight for Monero) and can influence the technical details of the communication protocols used within the pool [8].

The underlying consensus mechanism of the cryptocurrency can also play a role in the architecture of mining pools. While the focus here is primarily on Proof-of-Work (PoW) based cryptocurrencies, other consensus mechanisms like Proof-of-Stake (PoS) typically do not involve traditional mining in the same computational sense, though staking pools do exist. Furthermore, with the emergence of novel consensus mechanisms like Proof-of-Neural-Architecture (PoNA), new types of mining pool architectures are being explored to facilitate collaborative work on tasks like deep learning model training [6]. Understanding the different types of mining pools and the cryptocurrencies they support is important because different networks might exhibit variations in their network communication patterns, which could be exploited for detection strategies.

3.4 Overview of Mining Protocols

3.4.1 Necessity of Communication Protocols. For cryptocurrency miners and mining pool servers to interact effectively, standardized communication protocols are essential. These protocols establish a common language and a set of rules that govern the exchange of information necessary for the mining process, including tasks such as the assignment of mining jobs, the submission of computational results, and the dissemination of updates regarding mining difficulty or new blocks. Without these standardized protocols, the communication between diverse mining software and various pool server implementations would be chaotic and unreliable, hindering the efficient operation of the cryptocurrency network.

The use of such specific protocols implies that the network traffic generated by cryptocurrency mining activities will likely exhibit predictable communication patterns. These patterns, characterized by the types of messages exchanged, their frequency, and their structure, can be identified and analyzed through network traffic forensics to detect the presence of mining operations [13].

3.4.2 Stratum Protocol. The Stratum protocol has emerged as the most widely adopted communication protocol between cryptocurrency miners and mining pool servers. It was introduced in 2012 to address some of the limitations of earlier protocols, which we will later discuss, and has since become the *de facto* standard for pooled mining of many popular cryptocurrencies, including Bitcoin, Ethereum, and others.

Stratum operates over the Transmission Control Protocol (TCP), which provides a reliable and connection-oriented communication channel. TCP ensures that data packets are delivered in order and without loss, making it suitable for the continuous and often long-lived connections between miners and pool servers. The use of TCP means that network traffic associated with Stratum will typically involve the standard TCP handshake process (SYN, SYN-ACK, ACK) to establish a connection, followed by the exchange of data, and

finally a connection termination sequence (FIN) when the communication ends.

The Stratum protocol uses JavaScript Object Notation - Remote Procedure Call (JSON-RPC) as its message encoding format. JSON is a lightweight, text-based data format that is easy for both humans and machines to read and parse. JSON-RPC defines a simple protocol that allows for making remote procedure calls. In the context of Stratum, this means that the miner can send requests to the pool server to invoke specific methods, and the server can send responses back to the miner, as well as notifications for events like new mining jobs or changes in difficulty. The use of JSON-RPC implies that the content of the network packets exchanged between the miner and the pool server will follow a specific structured format, typically containing fields like "method" (indicating the action being requested or the event being notified), "params" (containing the parameters for the method), and "id" (used to match requests with their corresponding responses. Even if the underlying TCP connection is encrypted (e.g., using TLS/SSL), the characteristic structure and content of these JSON-RPC messages can sometimes be inferred or analyzed to identify Stratum protocol communication [13].

In Figure 1 we can see an example of a miner (in red) communicating with a pool (in blue). The communication can be described as the following steps: A connection to the mining pool begins with the initial message (#1), which includes the miner's authentication details. The server acknowledges this and confirms the connection with message (#2). It then sends a work package to the miner (#3) that needs to be processed. Once the mining software is properly initialized, the miner requests a new work package (#4), which the server provides in response (#5). After successfully computing the required hash, the miner submits the solution back to the server (#6). The server evaluates the submitted result, and if it is correct, as is the case in (#7), it sends another work package (#8) for the miner to work on. While processing the task, the miner also periodically reports its local computational speed to the server (#9) so the server can dynamically adjust the difficulty or size of future work packages.

[illegible]

Figure 1: Stratum Protocol Message Exchange

3.5 Earlier Protocols

Before the widespread adoption of Stratum, an earlier protocol known as 'GetWork' was commonly used for communication between cryptocurrency miners and mining pools, particularly with

the original Bitcoin client. However, 'GetWork' had several limitations that eventually led to the development and preference for Stratum [13]. One significant limitation was its reliance on the Hypertext Transfer Protocol (HTTP) for communication, which can be less efficient than TCP for the continuous and long-lasting connections typical of mining operation.

Another issue with 'GetWork' was its approach to work distribution. It typically required the miner to download and manage more of the blockchain data themselves, including constructing the block header. This could be resource-intensive for individual miners, especially as the blockchain grew in size and complexity. Furthermore, 'GetWork' often used a form of short polling, where the miner would repeatedly ask the server for new work. This could lead to inefficiencies and potential idling of miners if new work was not immediately available. Stratum addressed this with the use of long polling (though not explicitly mentioned in the provided snippets, it's a known advantage), where the connection between the miner and the server remains open, and the server pushes new work to the miner as soon as it's available, reducing latency and improving efficiency. The shift from 'GetWork' to Stratum highlights the importance of efficiency and scalability in mining protocols as cryptocurrency mining evolved. While 'GetWork' might still be in use in some older or niche mining setups, Stratum is the dominant protocol in modern pooled mining.

4 TRAFFIC MONITORING

4.1 Two-Step Detection Schema

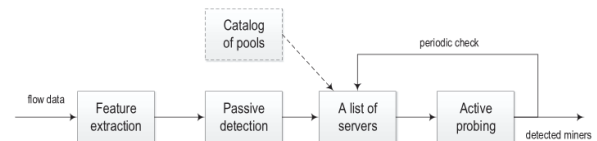


Figure 2: Two-step detection schema.

The implementation of the detection approach utilized a two-step schema, as seen in Figure 2. In the first step, the manual classifier was applied to passively observed traffic, analyzing the flow characteristics. Potential mining clients identified in this step were subjected to active probing in the second step. The goal of this probe was to confirm the existence of mining-related communication by eliciting specific responses from suspected mining servers. Probing was updated periodically due to the forever-changing mining-poll server list of known servers on the Internet.

4.2 Manual Rule-Based Classifier

To establish a baseline, a manually designed classifier was created using domain knowledge. This classifier uses a scoring scheme based on a series of simple heuristic rules derived from the distribution of features observed in the labeled dataset, as seen in Table 1. Each feature that met a mining-specific condition contributed an increment of $1/n$ to a total score, where n is the number of heuristic rules. The final classification decision was based on whether this score exceeded a chosen threshold T .

Feature	Condition
Avg. bytes per packet	35–80 or 105–110
Avg. packets per flow	Not in 5–40
Packets per minute	2–8 or 40–5300
Duration	> 300 seconds
ACK + PUSH ratio	> 90%
RST flag ratio	< 1%
SYN flag ratio	< 5%
FIN flag ratio	< 5%
Src port > Dst port	> 90%

Table 1: Selected features for manual classifier.

The threshold T was tuned manually to achieve approximately 90% true positive rate on the training data. However, due to the rigidity and limited adaptability of manually crafted rules, the classifier exhibited about 10% false positives and 10% false negatives during evaluation.

4.3 Machine Learning Classifier

To improve detection performance, a supervised machine learning classifier was trained on the labeled dataset. A decision tree model was selected for its balance between interpretability and performance. Prior to training, the ClassBalancer filter was applied to address the significant class imbalance.

Training was conducted using 5-fold cross-validation with 100 as the minimum number of instances per leaf node. This constraint results in a shallower tree, which improves generalization and reduces overfitting.

Features used by the model included all those utilized in the manual classifier, along with additional flow statistics such as:

- Flow duration and inter-arrival time statistics
- Packet size standard deviation
- Fraction of flows with repeated TCP flags
- Byte and packet volume ratios

4.4 Dataset Labeling and Preprocessing

The dataset used for training and evaluation was collected from the Czech National Research and Educational Network (CESNET), which interconnects more than 30 institutions, serving over 400,000 users. However, to ensure manageability and facilitate manual inspection, the data was narrowed to include only traffic from three subnets representing three major organizations, each with over 50,000 users.

The captured traffic was aggregated into flow triplets, each uniquely defined by the source IP address, destination IP address, and destination port. For each triplet, a set of flow-level features was computed and stored in memory. The system retained these features using a 1-hour inactivity timeout policy.

Triplets were annotated using a semi-automated labeling schema that combines passive analysis with active probing. In the passive phase, traffic characteristics were examined and compared against known mining patterns. In the active phase, probes were issued to suspected mining servers to validate whether they responded in a

manner consistent with mining pool behavior (same as in detection schema).

The final labeled dataset consisted of 273 positive samples and 356,574 negative samples (non-mining clients). The severe imbalance necessitated the use of balancing techniques during model training to prevent classifier bias toward the majority class.

4.5 Evaluation

During evaluation, the machine learning classifier outperformed the manual method across all key metrics. It achieved a false positive rate of 2.6% and a false negative rate of 4.7%. The confusion matrix indicated that mining communications were correctly identified in most cases, and misclassifications were substantially reduced.

5 CATALOG OF MINING POOLS

This section details the second approach presented in the article for detecting cryptocurrency miners in networks, which consists of creating a catalog of mining pools, exemplified by the sMaSheD project.

5.1 Motivation and Goal

The catalog approach was developed as a more straightforward and convenient route, which is suitable for even small corporate networks that lack the capacity or resources to install dedicated probes or implement machine learning-based traffic monitoring.

The main goal is to provide conclusive detection results with a minimum set of input information, in this case merely a catalog. If the IP address of a mining pool server is known, it is possible to reliably distinguish between mining and non-mining connections. Knowing the port number used by the pool operator allows for the identification of which cryptocurrency is being mined through that connection.

5.2 sMaSheD tool

Network administrators and law enforcement officials maintain logs that include source and destination IP addresses, port numbers, and protocol identifiers. By identifying the IP address of a mining pool server, we can distinguish mining traffic from other types of network activity: this is the logic behind the catalog-based approach. The hostname and port number of the mining server are generally publicly available on the pool’s website, as they are required to configure the mining software. The port number in particular is relevant to determine which cryptocurrency is being mined.

The sMaSheD project began by manually collecting mining software configurations advertised by the largest mining pools for several important cryptocurrencies. The data collected are stored in a database accessible via a Web application called sMaSheD (Mining Server Detector of Cryptocurrency Pools).

Information collected for each pool includes the following:

- The pool name and its home URL;
- A list of pool servers identified by FQDN (Fully Qualified Domain Name), including ports associated with mined cryptocurrencies;
- Each mining server FQDN is resolved into a list of IPv4/IPv6 addresses.

The choice of supported coins was based on public news, dedicated reports, and consultations with partners from law enforcement agencies. Popular cryptocurrencies such as Bitcoin (due to its importance), Monero (increasingly used by malware due to anti-forensic features), and Ethereum (popular due to smart contracts and tokens) were considered. sMaSheD was designed to be a generic catalog, easy to maintain and operate, and to include other cryptocurrencies in the future. A limitation is that the catalog does not contain a complete list of pools, as some are private and do not maintain publicly available web pages for web scraping. However, private pools make up only a small fraction of the total pools.

5.3 Operation of sMaSheD

The first step is to investigate information about the distribution of mining among pools as most of them add their signature to mined blocks and publicly announce their hashrate. This information can be obtained from dedicated web pages like Blockchain.info (for Bitcoin) or Etherscan.io (for Ethereum).

Pool operators provide FQDNs, which are resolved by miners into various IP addresses based on geolocation. A single FQDN can resolve to many IP addresses to ensure high availability. Given this, sMaSheD tries to keep the list of IPs associated with each mining server as up-to-date as possible, which is crucial for pools using cloud deployments due to IP address rotation.

5.3.1 Probing technique. To provide more reliable results on whether a given IP address belongs to a mining server, a probing mechanism was developed. The process entails probing that connects to a mining server pretending to be a regular miner asking for work. If the server responds with an expected response (for instance, a message containing a work package), it confirms that the device is a mining server for the pool. This technique is not supported by all mining protocols, which can be a limitation.

There are three main probing Return Codes:

- **DOWN:** Probing failed because the connection was not even established (port closed or blocked)
- **LISTEN:** The connection was accepted on the specified port, but the server returned an empty response (server using a different protocol or port connected to another application)
- **UP:** Probing succeeded because the mining server responded with a mining protocol message containing valid content (validity based on tests of JSON fields like error, result, etc.)

sMaSheD records each probing attempt, creating a history of service availability for a given mining server. This temporal data can prove that an IP address was used by a mining server at a certain time.

5.3.2 Probing evaluation. To validate probing results, sMaSheD’s behavior was compared with official mining software, specifically cgminer 3.7.2[4], due to its strong reputation and compatibility with all existing mining protocols.

Several parameters were tested, including connection success rate and the number of messages exchanged between the miner and the mining server. Overall, no significant differences were observed between sMaSheD and cgminer regarding connection attempts.

5.4 Implementation and Content

sMaSheD is coded as a web application using the PHP Laravel 5.8 framework with a Bootstrap 3 frontend. It runs in a Docker container on CentOS 7.

As of August 2018, the sMaSheD database contained:

- 15 cryptocurrencies mined on 96 different port numbers;
- 58 pools operating 212 servers with 987 addresses;
- 3954 probing associations (called "mining properties").

The sMaSheD catalog offers access to all data in JSON format, periodically probes pool servers to verify if they provide mining services, and records the results. It also allows privileged users to manually update the database.

5.5 sMaSheD Catalog Setup

To experiment with and visualize the sMaSheD catalog (Figure 3), we followed the instructions provided in the project’s GitHub repository (sMaSheD). However, we encountered difficulties in successfully accessing and interacting with the catalog. The main issue stemmed from the fact that some aspects of the repository have not been updated since the release of the paper in 2018, resulting in outdated dependencies and configurations that hindered our progress.

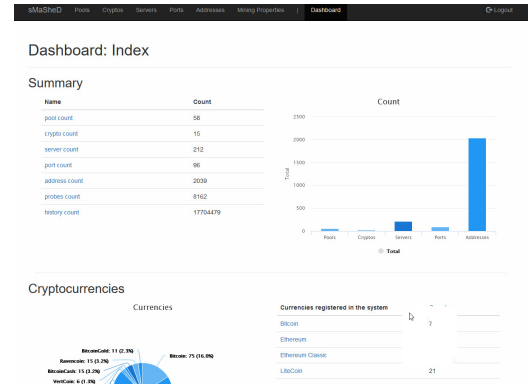


Figure 3: Smashed catalog overview

5.6 Future Work

The authors aim to make this approach more dynamic by automating data collection, exploring additional probing strategies, and ensuring it is continuously updated with new cryptocurrencies, mining pools, and servers as they emerge in public reports of illicit mining activities. Besides this catalog, there are a few other mining pool catalogs, for instance, WhereToMine [14]. WhereToMine is an online platform and mobile application designed to help cryptocurrency miners find mining pools. It allows users to explore mining pools for more than 700 cryptocurrencies, providing key details such as payout schemes, fees, minimum and maximum payout amounts, server locations, connection information, and more. While WhereToMine focuses on helping miners find quality pools, the sMaSheD catalog takes a slightly different approach by aiming to identify unwanted, resource-consuming mining pools, but it also maintains a catalog with some level of detail.

6 ANALYSIS AND IMPLICATIONS

Building on the detection and catalog approaches described in previous sections, this section analyzes their performance (Subsection 6.1) and discusses the implications for network security (Subsection 6.2).

6.1 Results of the detection methods

The sMaSheD catalog verification mechanism demonstrated robust reliability in identifying active mining servers. The catalog has successfully identified mining servers using Stratum and GetBlock-Template protocols, with responses validated via structured JSON field checks. Other protocols like GetWork were not tested.

In terms of passive traffic monitoring techniques, the supervised Machine Learning classifier outperformed the manual classifier with a 2.6% false positive rate (FPR) and 4.7% false negative rate (FNR) compared to the manual classifier that obtained a 10.3% FPR and 8.6% FNR (from Table 2).

This classifier leveraged a rich set of flow-level features (including TCP flag ratios, time of arrival statistics, and byte/packet volume ratios) to generalize effectively across diverse network environments.

The findings from the combined passive-active detection strategy indicate that, despite the relatively high occurrence of false positives following passive analysis, the rate remains low enough to allow for effective active validation of the detected instances. By contrast to the pure catalog approach, passive-active detection is able to discover private or not well-known pools. An overview of the results can be seen in Table 2.

(a) ML-classifier.			
Class	mining	other	
mining	95.3%	4.7%	
other	2.6%	97.4%	
(b) Manual classifier			
Class	mining	other	
mining	89.7%	10.3%	
other	8.6%	91.4%	

Table 2: Confusion matrix

6.2 Implications for network security

The reduction in false positives and negatives delivered by machine learning classifiers enables early, automated detection of unauthorized mining, closing the window for *cryptojacking* before resource depletion occurs.

Blocking known mining pool IPs and ports, through sMaSheD catalog, has been shown to reduce unauthorized mining traffic in enterprise networks, thereby preserving CPU or GPU cycles and preventing early server physical components failures.

Moreover, continuous catalog updates and active probing allow security teams to maintain historical proof of mining service availability, supporting both forensics and regulatory compliance (e.g. the regulatory institutions may want to use the catalog data to check whether there are mining farms under their jurisdiction).

From a defense perspective, integrating Machine Learning based traffic forensics into IDS/IPS (Intrusion Detection Systems) platforms provides real-time alerts on anomalous flows that deviate from legitimate application profiles, even when encryption or protocol obfuscation is employed.

Moreover, since unauthorized mining often occurs with other attacker behaviors (e.g. lateral movement through RCE exploits), cryptomining detection can serve as an early warning “canary in the coal mine”, prompting broader incident response measures to uncover and remediate deeper intrusions.

Finally, these implications underscore that a layered approach is essential for robust mitigation of illicit cryptocurrency mining in modern networks.

7 FORENSIC APPROACHES IN DIFFERENT CRYPTOCURRENCIES INVESTIGATIONS

Forensic analysis plays a crucial role in cryptocurrency investigations, extending beyond the study of unauthorized resource-consuming mining pools explored by the Veselý and Žádník article [13]. Koerhuis *et al.* [3] paper on “Forensic analysis of privacy-oriented cryptocurrencies” investigates the forensic artefacts left behind by the software of privacy-oriented cryptocurrencies, specifically Monero and Verge, on a computer system. These cryptocurrencies have built-in anonymity and privacy features that make it very difficult to trace funds back to a particular user or successfully seize funds present in a cryptocurrency wallet, being a perfect choice for criminals to execute their activities. This article considers digital evidence acquisition from volatile memory, network traffic, and virtual hard disks within a virtual machine environment. Using pre-known values, forensic artefacts were extracted, analyzed, and reported.

The forensic analysis demonstrated that Monero artefacts were present across memory, network, and disk, whereas Verge artefacts were primarily limited to disk and partially to memory. Network traffic for Verge was not useful due to TOR encryption. Both cases highlighted security weaknesses: Monero’s plaintext passphrases in memory and Verge’s plaintext data within encrypted disk files. These findings suggest that forensic analysis can identify software presence and potentially enable wallet access or recovery.

8 CONCLUSIONS

The seminar on detecting cryptocurrency miners using traffic forensics highlighted the critical importance of identifying both legitimate and illicit mining activities within network environments. Cryptocurrency mining, while fundamental to decentralized digital currencies by validating transactions and securing blockchains, is increasingly exploited through cryptojacking. This illicit activity

consumes victim resources and imposes costs, making detection crucial.

The seminar emphasized that understanding mining protocols, particularly Stratum, is key, as these protocols exhibit predictable communication patterns even when encrypted. A hybrid detection approach, which combined passive traffic analysis (using a supervised machine learning classifier trained on flow-level features) with active probing, was presented as an effective strategy. The machine learning classifier, which uses algorithms to learn complex relationships within network traffic data, significantly outperformed manual rule-based methods, demonstrating the power of data-driven approaches in accurately identifying mining traffic. The sMaSheD catalog approach offers a practical and complementary solution, especially for smaller networks, by maintaining a database of known mining pool IP addresses, which aids in distinguishing mining from non-mining connections and identifying the cryptocurrency being mined.

These detection methods offer substantial benefits for network security. They enable the prevention of resource depletion from cryptojacking, facilitate blocking unauthorized mining traffic, and can serve as an early warning for broader security intrusions. Historical data from catalogs such as sMaSheD also provides valuable forensic evidence and supports regulatory compliance.

Future work should focus on automating data collection, exploring new probing strategies, and continuously adapting to the evolving cryptocurrency landscape to ensure robust mitigation of illicit cryptocurrency mining.

REFERENCES

- [1] Cyber Threat Alliance. *Illicit Cryptocurrency Mining: An Increasing Enterprise Security Risk*. 2018. URL: <https://www.cyberthreatalliance.org/wp-content/uploads/2018/09/CTA-Illicit-CryptoMining-Whitepaper.pdf>.
- [2] Brian Johnson. "The Advantages and Disadvantages of the Deep Web, Tor Network, Virtual Currencies and the Regulatory Challenges Thereof". Master's thesis. Master's thesis. USA: Utica College, 2014. URL: [http://refhub.elsevier.com/S1742-2876\(18\)30335-9/sref26](http://refhub.elsevier.com/S1742-2876(18)30335-9/sref26).
- [3] Wiebe Koerhuis, Tahar Kechadi, and Nhien-An Le-Khac. "Forensic analysis of privacy-oriented cryptocurrencies". In: *Forensic Science International: Digital Investigation* 33 (2020), p. 200891. ISSN: 2666-2817. DOI: <https://doi.org/10.1016/j.fsidi.2019.200891>. URL: <https://www.sciencedirect.com/science/article/pii/S1742287619302312>.
- [4] Con Kolivas and contributors. *cgminer - ASIC and FPGA miner in C for Bitcoin*. Accessed: 2025-05-10. 2011. URL: <https://github.com/ckolivas/cgminer>.
- [5] Josef Koumar, Karel Hynek, and Tomáš ejka. "Network Traffic Classification based on Single Flow Time Series Analysis". In: *CNSM 2023–19th International Conference on Network and Service Management*. 2023. URL: https://www.researchgate.net/profile/Josef-Koumar/publication/372625466_Network_Traffic_Classification_based_on_Single_Flow_Time_Series_Analysis/links/64c0d4abb9ed6874a5451f9a/Network-Traffic-Classification-based-on-Single-Flow-Time-Series-Analysis.pdf.
- [6] Boyang Li et al. "A Collaboration Strategy in the Mining Pool for Proof-of-Neural-Architecture Consensus". In: *arXiv preprint arXiv:2206.07089* (2022). URL: <https://arxiv.org/abs/2206.07089>.
- [7] Jordi Zayuelas I Muoz. *Detection of Bitcoin miners from network measurements*. 2018. URL: <https://upcommons.upc.edu/bitstream/handle/2117/133503/137903.pdf>.
- [8] Jordi Zayuelas i Muoz, Jos   Su  rez-Varela, and Pere Barlet-Ros. "Detecting cryptocurrency miners with NetFlow/IPFIX network measurements". In: *2019 IEEE International Symposium on Measurements & Networking (M&N)*. 2019, pp. 1–6. DOI: 10.1109/IWMN.2019.8804995.
- [9] Arvind Narayanan et al. *Bitcoin and cryptocurrency technologies: a comprehensive introduction*. Princeton University Press, 2016.
- [10] Richard Pln  , Karel Hynek, and Tom    ejka. "DeCrypto: Finding Cryptocurrency Miners on ISP Networks". In: *Secure IT Systems*. Ed. by Hans P. Reiser and Marcel Kyas. Cham: Springer International Publishing, 2022, pp. 139–158. ISBN: 978-3-031-22295-5.
- [11] Reza Raeesi. "The Silk Road, Bitcoins and the Global Prohibition Regime on the International Trade in Illicit Drugs: Can This Storm Be Weathered?" In: *Glendon International Journal of Studies / Revue d  tudes internationales de Glendon* 8.1–2 (2015). Accessed 12/05/2025, p. 20. URL: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.960.6973>.
- [12] Michele Russo, Nedim S  rdi  , and Pavel Laskov. "XMR-Ray: Detecting Illicit Cryptomining using Network Metadata". In: *EURASIP Journal on Information Security* 2021.1 (2021), pp. 1–13. DOI: 10.1186/s13635-021-00175-5.
- [13] Vladim  r Vesel  y and Martin Z  dn  k. "How to detect cryptocurrency miners? By traffic forensics!" In: *Digital Investigation* 31 (2019), p. 100884. ISSN: 1742-2876. DOI: <https://doi.org/10.1016/j.diin.2019.08.002>. URL: <https://www.sciencedirect.com/science/article/pii/S1742287618303359>.
- [14] WhereToMine. *WhereToMine: Cryptocurrency Mining Pool Explorer*. Accessed: 2025-05-10. 2025. URL: <https://wheretomine.io/about/>.

Was the clock correct? Timestamps and time anchors across operating systems

Sofija Đukić
Faculty of Computer and Information
Science,
University of Ljubljana
sx91124@student.uni-lj.si

Gregor Perčič
Faculty of Computer and Information
Science,
University of Ljubljana
gp65292@student.uni-lj.si

Anja Pijak
Faculty of Computer and Information
Science,
University of Ljubljana
ap72051@student.uni-lj.si

ABSTRACT

Accurate event reconstruction in digital forensics depends on the integrity of system-generated timestamps, yet operating-system clocks are often misaligned with real time because of drift, battery failure or deliberate tampering. The authors formalise this problem by introducing time anchors: artefacts that embed both a local timestamp supplied by the device and an external timestamp issued by an independent source. By comparing the two we can quantify clock skew at the moment an artefact is created and label events as anchoring (verifiable) or non-anchoring. The authors also define time anomalies — inconsistent skews or impossible temporal sequences — that signal manipulation or hardware faults.

Two controlled Windows 10 experiments validate the approach. First, Google-search artefacts (browser history, cache entries and server logs) were collected with the system clock set correctly and then shifted three hours behind real time. The skew measured in each anchoring artefact precisely matched the imposed offset and exposed the unreliability of non-anchoring events. A second experiment repeated the procedure for local file creation, demonstrating how non-anchoring events can be corrected once neighbouring anchors establish the true skew.

Our addition is a review of methods of timestamp manipulations on Linux system, in which we review how to spot a faked timestamp using the touch command, extended file attributes, and journal history.

KEYWORDS

timestamp, time shift, time anchor, OS forensics

1 INTRODUCTION

Event reconstruction is a key aspect of criminal investigations, especially in digital forensics, where the timestamps are heavily relied upon to establish a sequence of events. These timestamps are usually derived from a computer's internal clock via operating system APIs, using the hardware's Real-Time Clock as the source. However, system time does not always align with real-world time [6]. Variations can result from tampering, natural clock drift, or

hardware issues like a depleted battery [10]. These discrepancies are referred to as clock skew, and they can severely alternate the accuracy of reconstructed timelines in forensic investigations.

An example of importance of correct timestamps: if a suspect searches for information related to the murder before it occurs, it may suggest intent; if the search happens after, it may be unrelated or a reaction to news. Therefore, verifying the correctness of system time is crucial.

To address this, the authors introduce the concept of *time anchors*—combinations of local and external time references that help assess the reliability of timestamps. They also define time anomalies, which describe differences between expected and actual time references. The paper supports these concepts with controlled experiments using timestamped actions like Google searches and file creation, conducted on Windows systems.

The study specifically focuses on identifying and analyzing clock skew at the point of timestamp generation. It excludes manipulation of timestamps after creation or complications arising from transferred files between systems with different time configurations.

2 RELATED WORKS

Previous work in digital forensics has explored different ways of dealing with time-related issues, mostly through two approaches: using clock models and correlating timestamps with external sources. Clock models try to describe how system clocks behave over time, including things like drift or adjustments. Some studies aimed to reconstruct what a clock might have looked like in the past [11], while others suggested forming hypotheses about the clock's behavior and checking if timestamps make sense based on the order of events—for instance, whether one event should have happened before another [15].

Another common method is timestamp correlation. This means comparing system timestamps with ones from outside sources, like web server times saved in cached pages or mobile network times from missed calls or messages. These comparisons can help figure out if a device's clock was accurate at the time of an event. But this approach depends heavily on having access to those external time references, which can be tricky—they might not be saved for long, or legal access might be needed.

3 DEFINITIONS AND TERMINOLOGY

The authors distinguish between three types of time:

- **System clock time** (C_S) – the time reported by the operating system.
- **Real-world time** (C_W) – the actual current time.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

DigFor-25, May 30–June 05 2025, Ljubljana, Slovenia

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
<https://doi.org/XXXXXXX.XXXXXXX>

- **External clock time** (C_E) – a timestamp obtained from a trusted external source, such as an Internet server.

Ideally, these values would be equal ($C_S = C_E = C_W$). If they are not, the difference is referred to as *clock skew* (S).

Since clock skew is not expected to vary significantly during a system’s lifetime, it can be measured at a specific point in time by comparing the following:

- T_S : the system-generated timestamp,
- T_E : the timestamp from an external clock,
- T_W : the timestamp representing the actual time the event occurred.

Definition 1. *Clock skew* S is defined as the difference between real-world time and system time:

$$S = T_W - T_S$$

Minor clock skew is considered tolerable, especially in the context of user activity reconstruction. However, in some cases, a discrepancy of just a few seconds can be significant.

The authors note that for simplicity, time zone differences are ignored, and all timestamps are treated as if they are based on the same time zone. While this assumption simplifies the model, time zone awareness remains important in actual forensic investigations [2].

3.1 Validating Timestamps through Time Anchors and Anomalies

To reliably reconstruct digital events, timestamps must be trustworthy. Since the system clock (C_S) can be inaccurate, *time anchors* [5]—artifacts containing both system timestamps (T_S) and reliable external timestamps (T_E)—help validate clock correctness. If T_S and T_E match closely, the system time was accurate at that point; otherwise, their difference indicates clock skew.

Events inferred from data can be *anchoring* or *non-anchoring*. Anchoring events come from artifacts that serve as time anchors, allowing precise validation of system time when the event occurred. For example, a Google search event inferred from cached files containing both T_S and T_E is self-anchoring. Non-anchoring events lack external timestamps (like file access times) and cannot confirm clock accuracy directly but can be bounded between known time anchors. However, this bounding may be misleading if the system clock was skewed.

Time anomalies indicate inconsistencies suggesting clock errors. These include explicit log entries of clock changes, impossible timestamp sequences (e.g., files created out of order), or irregular synchronization intervals.

Finally, time anchors are either *local*—when both T_S and T_E are stored on the same device—or *remote*, requiring external data (like call records) for comparison. While remote anchors add value, local anchors are more commonly accessible and are the focus here.

4 TIME ANCHORS IN FORENSIC INVESTIGATIONS

Time anchors and anchoring events enable examiners to assess the reliability of a system clock. When system timestamps (T_S) closely match external timestamps (T_E), it is reasonable to conclude

the clock was likely correct at that moment, barring deliberate tampering.

Artifacts that produce anchoring events must involve external time sources, typically triggering remote events. Common examples include browser data (e.g., cache files, cookies with creation and expiry times), messaging apps (e.g., Signal records both sent time from an external source and local receive time), email clients (e.g., Thunderbird stores local send times and server times in email headers), and cloud storage services (e.g., Google Drive synchronization times based on server clocks compared with local file system timestamps). Many web-based apps and network logs may also provide useful external timestamps. This work focuses on Chrome cache, browser history, and Windows event logs, though additional anchors could enhance analyses without changing the underlying approach.

Automatically identifying time anchors is challenging and beyond this paper’s scope. Here, the authors used a manual method, comparing artifact timestamps under both correct and skewed system time conditions. Although timeline tools like Plaso and Timesketch facilitate timeline generation, they often miss external-source timestamps such as those in email headers. Therefore, advancing automation—via specialized plugins or tools that supplement existing frameworks—is essential for scaling analysis, enabling detection of time discrepancies, and automatically tagging anchors within forensic timelines.

5 THE AUTHORS’ EXPERIMENTAL DESIGN AND RESULTS

5.1 Experiment Setup

The authors conducted two controlled experiments to demonstrate their proposed concepts of time anchors, anchoring events, and non-anchoring events for assessing the correctness of system clocks in digital forensic investigations. Both experiments were performed on a Windows 10 system [13].

The second experiment focused on file creation. Here, the team created a file on the same Windows system and recorded the local creation timestamp. They also attempted to obtain an external reference time—using either a synchronized online time server or manual notes—to compare with the local timestamp. As with the first experiment, this was repeated under both correct and skewed clock conditions [13].

Anchoring events are defined as events for which both a local timestamp (from the device’s system clock) and an external timestamp (from a source not influenced by the device, such as a server log) are available. These allow for direct comparison and assessment of clock skew [13]. Non-anchoring events only have a local timestamp, making it impossible to verify their timing independently unless they can be related to an anchoring event [3]. The distinction between these event types is critical, as highlighted by [7], who proposed automated methods for detecting inconsistencies in digital timelines.

5.2 Results and Findings

The experiments produced clear and reproducible findings regarding the detection and correction of system clock skew.

5.2.1 Google Search Experiment. When the system clock was set correctly, the local and external timestamps for the search event matched closely, with only minor differences (usually less than a second), attributable to network latency. However, when the system clock was set 3 hours behind, the local timestamp for the search (as seen in browser history and cache) was, for example, 2023-04-23T15:53:16 (local time), while the Google server recorded the event as 2023-04-23T18:53:16Z (external time). This 3-hour discrepancy was consistent and directly reflected the induced clock skew [13].

The browser cache stored a local copy of the search results page. The timestamp on this cached file was generated by the local system clock. In the skewed scenario, the cache file's timestamp was always 3 hours behind the real event time, perfectly matching the manipulated skew. This demonstrates that cache artifacts reliably reflect the local system time and can be used to measure clock skew when compared to external references. However, cache timestamps alone are non-anchoring and only become useful for verifying clock correctness when paired with an external timestamp [13].

The browser history also recorded the time the search was performed, again using the local system clock. The experiment showed that history entries for the search query reflected the 3-hour lag when the system clock was skewed. History artifacts, like cache, are subject to system clock skew and are only trustworthy for clock verification when matched with an external reference [13].

5.2.2 File Creation Experiment. When the system clock was correct, the local file creation time matched the actual time. When the system clock was skewed, the local file creation time was always 3 hours behind the actual time, as confirmed by the external reference (see Table 2 in the paper). There was no automatic external timestamp for the file creation event, so the authors relied on manual synchronization. This highlights the limitation of non-anchoring events: without an external reference, their timing cannot be independently verified. If an anchoring event is available nearby in the timeline, its measured skew can be used to adjust the timing of non-anchoring events [13].

5.2.3 Time Anomalies. In both the Google search and file creation experiments, the authors did not observe any time anomalies beyond the deliberately induced 3-hour skew. This is expected in a controlled setting, where no further tampering or unpredictable system behavior occurs. All anchoring events (those with both local and external timestamps) showed the exact 3-hour difference, and non-anchoring events (like cache and history alone) could be explained by the same skew [13].

However, the authors emphasize that the methodology they present is designed to detect time anomalies in real-world scenarios. For example, if a user or an attacker changes the system clock between events, or if the system clock drifts unpredictably due to hardware faults, the skew between local and external timestamps would not be consistent across all events. This would manifest as time anomalies—some events would have a different skew than others, or the skew might even change direction (from positive to negative). Such anomalies could indicate tampering, technical faults, or other irregularities that are critical for forensic analysis [14].

6 DISCUSSION AND CRITICAL PERSPECTIVE

The authors' work makes a substantial contribution by formalizing the concepts of time anchors, anchoring events, non-anchoring events, and time anomalies for digital forensic event reconstruction. Their experiments clearly show that when an event produces both local and external timestamps—such as a Google search—investigators can directly assess the correctness of the system clock at that moment. This is a significant improvement over relying on local timestamps alone, which, as their file creation example demonstrates, can be misleading if the system clock is incorrect.

One of the strongest aspects of the paper is the structured approach to distinguishing between anchoring and non-anchoring events. The use of multiple time anchors (for example, cache and history entries in addition to server logs) increases confidence in the findings, especially when those sources are independent and in agreement [13]. This aligns with the principles outlined by [4], who emphasized the need for corroborative evidence in forensic investigations.

However, the paper also exposes some challenges and limitations. First, the entire experimental setup is based on Windows 10, and while the authors claim their concepts are generalizable, this has not been empirically demonstrated. It remains to be seen how well this approach translates to other platforms like Linux or macOS, especially given differences in how timestamps are stored and managed.

Another limitation is the reliance on external time sources. The authors acknowledge that not all external sources are equally trustworthy. Some might be controlled or influenced by the suspect, or simply be out of sync themselves. In practice, investigators may not always have access to reliable or independent external logs, which could limit the applicability of the time anchor method. The authors suggest a probabilistic approach when only a single anchor is available, but this still leaves some uncertainty in the timeline [3].

In summary, the paper provides a well-structured and much-needed formalization of timestamp validation in digital forensics, but its practical effectiveness will depend on broader testing across platforms and scenarios. In our own future work, we plan to execute our own experiments on a Linux system to evaluate whether the time anchor methodology is robust and generalizable.

7 TIMESTAMP METHODS ON LINUX SYSTEMS

Given that the source paper surveys forensic methods relating to timestamps on Windows systems, we add to it a survey of similar methods on Linux systems, with practical examples.

`stat` is a tool on Linux which allows us to investigate detailed information about a file.[8] This can help us spot possible discrepancies in the file data.

For example, suppose that an attacker deletes the file `evidence.txt` and then tries to fake (backdate!) its timestamp by recreating it using

```
user@user:~$ touch -d "2025-04-29 16:21:42" evidence.txt
```

Using only `ls -l`, we get the faked timestamp (modification time):

```
-rw-r--r-- 1 user user 0 Apr 29 16:21 evidence.txt
```

The discrepancy is clear when investigated with `stats`:

```

user@user:~$ stat evidence.txt
  File: evidence.txt
  Size: 0          Blocks: 0          IO Block: 4096
Device: 8,1 Inode: 2359867    Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/   user)
Access: 2025-04-29 16:21:42.000000000 +0100
Modify: 2025-04-29 16:21:42.000000000 +0100
Change: 2025-05-01 12:11:50.989667648 +0100
 Birth: 2025-05-01 12:11:50.989667648 +0100

```

stat lists some of the inode data. For example, Modify is mtime, last change of file *content*, Change is ctime, last change of file *metadata*, and most importantly, Birth is crtime, the time of file's creation as recorded in the inode. Therefore, since in this case, the last modification of the file *precedes* the creation time of the file, this is evidence of deliberate backdating. An additional giveaway is that the modification time 16:21:42.000000000 is whole (without fractions of a second), which seems artificial compared to the actual last time of modification (which just is the time of creation in this case), 12:11:50.989667648. An uncautious attacker might only modify the timestamp up to a second's precision and therefore leave such an additional trace.

Suppose now that the attacker is able to manipulate all of the inode attributes as well. Finding fakes is still possible in some cases, via inspection of extended file attributes. For example, if a document doc.odt is connected to a cloud platform, it may be periodically synced with the cloud, which is recorded in one of its extended attributes, which modifies the file:

```
user.cloud.sync_time="2025-05-01T12:11:50"
```

Therefore, if mtime precedes the timestamp in this attribute, this is again evidence of backdating. We can inspect such attributes with the getfattr utility.[1] We can dump all hidden attributes with

```
getfattr -d -m - doc.odt
```

Based on our testing, not many services use extended attributes today, so this approach is unlikely to succeed. For example, 10 years ago, files downloaded with Firefox had extended attributes, but we are unable to reproduce this behavior today.[12]

Lastly, we can utilize journaling to find out whether the timestamps pertaining to a file (again, let us say) evidence.txt is authentic. If the first activity on this file precedes file creation time (crtime), this is evidence of forward dating. And conversely, if any of the inode time attributes precede the earliest journal entry, this is evidence of backdating. ext4magic is a program which allows us to inspect the journal history of each inode (in our case, the inode tied to evidence.txt).[9]

```
sudo ext4magic /dev/sda1 -f user/evidence.txt
```

We have to specify the file system /dev/sda1 and an absolute path from its mount point to the file of interest (in this case, the mount point would be /home, so we provide user/evidence.txt). A possible complication is that, at some point in the past, there was another unrelated file named evidence.txt tied to the same inode on the inspected file system, which is unlikely, but not impossible. To check this, we must trace the history of all files with this specific name tied to the specific node of the currently investigated file. If the journal entries on the last instance of evidence.txt with anomalous timestamps on our specific node do not end with its

deletion, we can be (more) certain that these journal entries refer to the file under investigation, since an inode must be “deleted” (or perhaps a better expression would be “freed”) in order to be reused. If, on the other hand, the anomalous entries do end with deletion, the anomalous data is explained without reference to deliberate tampering with timestamps.

8 CONCLUSION

Understanding whether a system clock is accurate is crucial in digital forensic investigations because it affects how events are reconstructed. The original paper [13] explored this by introducing the concept of “time anchors,” which use events with both local and external timestamps to detect clock changes, focusing mainly on Windows 10. In our overview, we summarized their main findings and extended the discussion to Linux system by surveying similar forensic methods with practical examples. We demonstrated how Linux's stat command reveals multiple file timestamps-including modification time (mtime), change time (ctime), and creation time (crtime), which can be compared to detect timestamp manipulation, such as deliberate backdating using the touch command. Furthermore, we explored using filesystem journaling tools like ext4magic to compare file timestamps with the system's recorded activity. This Linux-focused survey complements the original Windows based approach, showing that despite differences in tools and filesystem features, time anchors remain a powerful concept for verifying timestamp integrity in digital forensic investigations.

REFERENCES

- [1] Andreas Grünbacher. 2025. *getfattr(1) — Get extended attributes of filesystem objects*. <https://man7.org/linux/man-pages/man1/getfattr.1.html>.
- [2] C. Boyd and P. Forster. 2004. Time and date issues in forensic computing—a case study. *Digital Investigation* 1 (2004), 18–23. <https://doi.org/10.1016/j.diin.2004.01.002>
- [3] Florian Buchholz and Brett Tjaden. 2007. A brief study of time. *Digital Investigation* 4 (2007), 31–42. <https://doi.org/10.1016/j.diin.2007.06.004>
- [4] Eoghan Casey. 2020. Standardization of forming and expressing preliminary evaluative opinions on digital evidence. *Forensic Science International: Digital Investigation* 32 (03 2020), 200888. <https://doi.org/10.1016/j.fsidi.2019.200888>
- [5] Frederick B. Cohen. 2010. *Digital forensic evidence examination*. Asp Press Livermore.
- [6] H. Marouani and M.R. Dagenais. 2008. Internal clock drift estimation in computer clusters. *Journal of Computer Systems, Networks, and Communications* (2008), Article ID 583162. <https://doi.org/10.1155/2008/583162>
- [7] Andrew Marrington, Ibrahim Baggili, George Mohay, and Andrew Clark. 2011. CAT Detect (Computer Activity Timeline Detection): A tool for detecting inconsistency in computer activity timelines. *Digital Investigation* 8 (2011), S52–S61. <https://doi.org/10.1016/j.diin.2011.05.007> The Proceedings of the Eleventh Annual DFRWS Conference.
- [8] Michael Kerrisk. 2025. *stat(1) — Display file status*. <https://man7.org/linux/man-pages/man1/stat.1.html>.
- [9] Roberto Maar. 2011. *ext4magic — Recover deleted files from ext3/4 partitions*. https://ext4magic.sourceforge.net/manpage_en.html.
- [10] J.P. Sandvik and A. Arnes. 2018. The reliability of clocks as digital evidence under low voltage conditions. *Digital Investigation* 24 (2018), S10–S17. <https://doi.org/10.1016/j.diin.2018.01.014>
- [11] M.W. Stevens. 2004. Unification of relative time frames for digital forensics. *Digital Investigation* 1 (2004), 225–239. <https://doi.org/10.1016/j.diin.2004.07.003>
- [12] unknown. 2014. Why doesn't getfattr -d show anything? <https://unix.stackexchange.com/questions/180019/why-doesnt-getfattr-d-show-anything>. Unix & Linux Stack Exchange, accessed 2025-05-01.
- [13] Céline Vanini, Christopher J. Hargreaves, Harm van Beek, and Frank Breiting. 2024. Was the clock correct? Exploring timestamp interpretation through time anchors for digital forensic event reconstruction. *Forensic Science International: Digital Investigation* 49 (2024), 301759. <https://doi.org/10.1016/j.fsidi.2024.301759> DFRWS USA 2024 - Selected Papers from the 24th Annual Digital Forensics Research Conference USA.

- [14] Svein Willassen. 2008. Hypothesis-Based Investigation of Digital Timestamps. *International Federation for Information Processing Digital Library: Advances in Digital Forensics IV*; 285. https://doi.org/10.1007/978-0-387-84927-0_7
- [15] S.Y. Willassen. 2008. Timestamp evidence correlation by model based clock hypothesis testing. In *Proceedings of the 1st International Conference on Forensic*

Applications and Techniques in Telecommunications, Information, and Multimedia and Workshop. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), Brussels, BEL, 1–6.

Received 18 May 2025

Enhancing speaker identification in criminal investigations through clusterization and rank-based scoring

Lan Vukušić
lv8541@student.uni-lj.si
Fakulteta za računalništvo in
informatiko
Ljubljana, Slovenia

Enei Sluga
es6878@student.uni-lj.si
Fakulteta za računalništvo in
informatiko
Ljubljana, Slovenia

Luka Pantar
lp6715@student.uni-lj.si
Fakulteta za računalništvo in
informatiko
Ljubljana, Slovenia

ABSTRACT

Speaker identification is a crucial task in criminal investigations, but analyzing large volumes of noisy audio and comparing them with extensive databases often leads to a high rate of false positives when using standard methods.

This paper presents an approach to improve the identification of forensic speakers by combining state-of-the-art speaker embeddings with clustering and a rank-based scoring mechanism. The method extracts embeddings using an ECAPA-TDNN model, groups audio segments using HDBSCAN, and applies a rank-adjusted scoring system.

Experiments demonstrate that this approach significantly reduces the number of potential candidates requiring manual review compared to baseline methods, making the investigation process more efficient and manageable in real-world forensic scenarios.

KEYWORDS

Digital forensic, Speaker identification, Clusterization, Audio analysis, Scoring

1 INTRODUCTION

Speaker identification is vital in criminal investigations, yet standard methods struggle with the large, noisy audio datasets and extensive databases typical in forensic contexts, resulting in numerous false positives. This challenge limits the practical application of advanced speaker recognition technologies for investigators.

This paper presents a method to enhance forensic speaker identification by integrating state-of-the-art speaker embeddings with clusterization and a rank-based scoring system. This approach aims to significantly reduce false positives and improve the efficiency of identifying potential suspects. We detail the proposed method, discuss its evaluation, and present our implementation of key aspects, highlighting its potential utility in real-world forensic analysis.

2 PROPOSED APPROACH

The paper by Moura et al. introduces an approach designed to support the identification of speakers in criminal investigations [5]. The complete proposed pipeline can be focused on the picture 1 and will be divided into two stages: *cluster scoring* and *speaker identification*.

2.1 Cluster scoring

At its core, the method extracts voice embeddings (vector representation) using a state-of-the-art model ECAPA-TDNN¹ [2]. Voice samples are gathered and later compared from Target database, which represents a cell phone with multiple voice utterances and Enrollment database - a voice collection of the prison system, that connects individuals with their voices. This model is robust in various languages, but tends to produce many false positives in large databases, requiring additional processing.

Cosine similarity (degree of vectors) in equation 1 is used for scoring to measure the similarity between the target and enrollment speaker embeddings, as seen in equation. While effective, it tends to generate many false matches in large datasets.

$$\text{CosineSimilarity}(A,B) = \frac{A \cdot B}{\|A\| \|B\|} \quad (1)$$

In order to reduce false positives, raw scores are adjusted by incorporating rank and a special parameter α for decay rate as expressed in the equation 2. $score_{i,s}$ represents similarity between target audio utterance i (target database) and audio from speaker s (enrollment database). On the other hand $rank_{i,s}$ denotes the speaker rank s for the target audio i , ranging from 0 to *total number of individuals* - 1. Therefore, this adjustment emphasizes top-ranked matches more strongly and penalizes lower-ranked ones.

$$\text{AdjustedScore}(score_{i,s}, rank_{i,s}, \alpha) = score_{i,s} \cdot \frac{\alpha}{rank_{i,s} + \alpha} \quad (2)$$

To reduce variability and noise in individual audio samples, audio clips are proposed to be grouped using the HDBSCAN² clustering algorithm. This simplifies analysis to per-cluster basis and also enhances match reliability.

The last step in the first part of pipeline determines an overall score (ClusterScore in equation 3) for each cluster c by averaging the adjusted scores of all its recordings i against each speaker s in the enrollment database, as defined in the following equation. This favors speakers with consistently high individual scores. [5]

$$\text{ClusterScore}_{c,s}(c, s, \alpha) = \frac{1}{\text{length}(c)} \sum_{i \in c} \text{AdjustedScore}(score_{i,s}, rank_{i,s}, \alpha) \quad (3)$$

¹Emphasized Channel Attention, Propagation and Aggregation in Time Delay Neural Network

²Hierarchical Density-Based Spatial Clustering of Applications with Noise

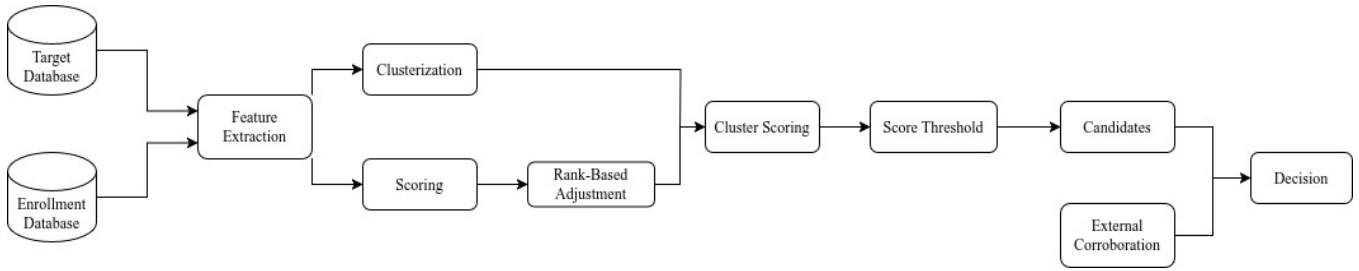


Figure 1: Complete pipeline with cluster scoring and speaker identification

2.2 Speaker identification

The speaker identification pipeline uses a cluster scoring approach, employing both absolute and relative score thresholds to narrow down potential speaker candidates. These thresholds often need to be adjusted individually for each cluster to optimize performance. Following the scoring of each audio cluster against the enrollment database, candidates with low scores or those substantially below the highest score are eliminated. This significantly reduces the number of potential matches per cluster.

Notably, audio from a single speaker can sometimes be divided into multiple clusters, or a cluster may contain voices from different individuals. Given the inherent limitations of voice identification, even with advanced techniques, supplementary verification is required. Final candidate selection relies on external corroboration and human judgment, acknowledging that subtle similarities in voice characteristics necessitate additional evidence for accurate identification. [5]

2.3 Experiments

We will take a look at the experiments performed in the studied paper with proposed approach.

2.3.1 Common voice experiment. First experiments used a synthesized dataset derived from the multilingual Common Voice collection (Portuguese, French, Spanish, Italian). In the effort of replicating real-world scenario, each speaker was enrolled with one utterance, and the rest were used to simulate 30 virtual cell phones, with limitation of 300 utterances per speaker to ensure balance and 3000 cumulative samples.

ECAPA-TDNN generated 192-dimensional speaker embeddings. Visualizations of high-dimensional data can be done using UMAP³ [4] showed clear grouping of genders, although age-based separation was less distinct.

Next, HDBSCAN was applied to group utterances by speaker. While clustering reduced the Equal Error Rate (EER), many utterances were left unclustered. ERR is a key metric to determine performance of classification in biometric systems [7]. It represents a value where *false acceptance rate* equals *false rejection rate*.

By changing the *minimum cluster size* parameter, a balance between coverage (proportion of samples in clusters) and cluster predominance (distribution of recordings connected to primary speaker) can be achieved. Optimal cluster size was determined to be **30**, which is used in further sections. Baseline approach (each

utterance scored independently) was compared to clustering, but the clustering approach didn't significantly improve EER.

Introducing a rank-based score adjustment in combination with clustering significantly lowered and therefore improved EER, showing statistical improvements over clustering alone. Decay rate parameter value $\alpha = 10$ was based on experiments and observations. [5]

2.3.2 Applied experiment. Next set of experiments more closely resembles the real world scenario, using the data set, supplied by Brazilian government agencies MPCE and SAP-CE.

MPCE provided audio data from 67 seized prison cell phones, resulting in over 400,000 utterances. 86 individuals were externally validated through non-audio evidence.

SAP-CE contributed prisoner enrollment data collected upon admission. Information includes speech samples and biometric details. Though the audio data contains noise and inconsistencies, it covers 69453 verified unique speakers.

Both the baseline and the proposed clustering-based approach were tested. Couple of different variables were manipulated to find more optimal solution: maximum (absolute) threshold and the relative threshold to the maximum threshold. While the baseline identified more validated candidates (87%), it generated an overwhelming number of potential matches (more than 300,000) including numerous false positives. Potential candidates can be considered as matches between an audio cluster (or recording) and a prisoner from the enrollment database. The proposed method (clusterization) identified much fewer validated candidates (35%), but significantly reduced the number of potential candidates (approximately 300), hence making manual investigations more manageable. [5]

2.4 Comparison to State of the Art

The work leverages the ECAPA-TDNN model, which is acknowledged as a state-of-the-art framework for speaker verification in realistic scenarios within the machine learning and AI community [5]. However, the paper highlights a critical distinction between general speaker verification and the specific demands of forensic speaker identification. While powerful, directly applying these state-of-the-art models to the large, often low-quality, and noisy datasets typical of criminal investigations results in an unmanageable number of false positives when compared against vast enrollment databases. This limitation hinders their practical utility for forensic analysts. Recent research continues to build upon or propose alternatives

³Uniform Manifold Approximation and Projection

to models like ECAPA-TDNN, aiming for improved performance or robustness in various challenging conditions, such as the CRET model which fuses Conv2D, ResNet, and ECAPA-TDNN components [3], or models designed for specific noisy environments like air-ground communication [6].

The proposed approach directly addresses this gap between general AI/ML state-of-the-art in speaker recognition and the practical requirements of forensic speaker recognition. By integrating clusterization and a rank-based scoring adjustment with the ECAPA-TDNN embeddings, the method creates a system specifically tailored to the forensic context. The clusterization helps to consolidate voice patterns from potentially fragmented or variable recordings of the same speaker, while the rank-based scoring prunes the vast number of potential matches generated by direct comparison against a large database.

Experimental results presented in the paper demonstrate that this combined strategy offers a significant improvement over a baseline approach that relies solely on individual audio comparisons using the state-of-the-art embeddings [5]. While the baseline produced an overwhelming number of candidates, making manual verification impractical, the proposed method substantially reduced the candidate pool while maintaining a reasonable identification rate for validated speakers. This demonstrates that the paper's contribution lies not just in using a state-of-the-art model, but in developing a system around it that effectively adapts it to the unique challenges and scale of forensic investigations, a capability often lacking in general speaker recognition systems or existing forensic tools focused on other data types.

2.5 Novelty

Although clusterization and rank-based scoring are not novel AI techniques, their specific *combination and application* with a state-of-the-art speaker recognition model to enhance performance and practicality in the identification of forensic speakers within criminal investigations is a key contribution [5].

2.6 Potential for Future Improvement

Enhancements could involve exploring different classification and clustering algorithms, possibly drawing inspiration from fields such as image processing or bioinformatics that handle complex data analysis. Integrating other forensic data modalities and leveraging speech-to-text and NLP for content analysis are also promising avenues. Improving robustness to data variations and incorporating iterative learning from identified cases could further refine the system [5].

3 OUR IMPLEMENTATION

3.1 Experimental Setup and Data

Our implementation builds upon the concept of using speaker embeddings for identification, as explored in recent work [5]. For our experiments, we utilized a subset of the Common Voice dataset, comprising approximately 900 speakers, which allowed us to replicate initial clustering analyses. We processed audio files to extract fixed-duration chunks associated with their speaker IDs. Additionally, we prepared a small dataset of our own voice recordings, organized

by speaker, to test the system's performance on custom audio data [1].

3.2 Embedding Extraction and Initial Clustering

Speaker embeddings were extracted from the audio chunks using a pre-trained ECAPA-TDNN model from the SpeechBrain library. These embeddings are 192-dimensional vector representations for each audio segment. To visualize the structure of these embeddings and assess initial clustering, we reduced their dimensionality to 2D using UMAP, employing cosine distance, which is suitable for speaker embeddings. Plots of embeddings derived from the Common Voice subset showed discernible groupings corresponding to individual speakers [1]. Figure 2 shows Umap visualization of the embeddings and is color coded by speakers. It presents that implementation correctly clusters speakers into visible clusters.

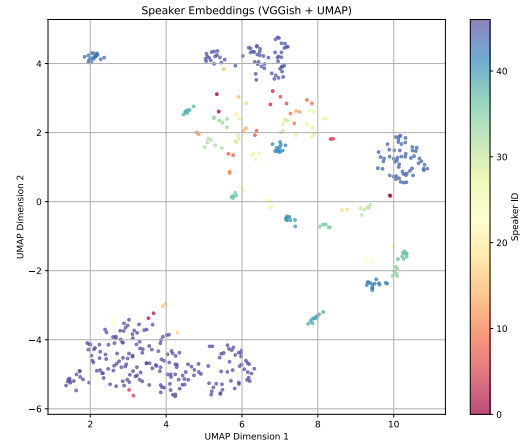


Figure 2: Umap visualization of the embeddings.

3.3 Contrastive Model for Improved Clustering

To enhance the discriminative quality of the embeddings and achieve more well-defined clusters, we developed and trained a custom Speaker Contrastive Network [1]. The purpose of this network is to refine the embedding space by learning to pull embeddings from the same speaker closer together while pushing embeddings from different speakers further apart. The network takes pairs of embeddings as input and is trained using positive (same speaker) and negative (different speaker) examples. The architecture includes a series of linear layers, BatchNorm1d, and ReLU activation functions, followed by dropout, projecting the initial 192-dimensional embeddings down to a 64-dimensional refined embedding space. A contrastive head then processes pairs of these refined embeddings. Applying this trained network to the ECAPA-TDNN embeddings resulted in refined embeddings that, when visualized with UMAP, exhibited more distinct and tighter clusters for individual speakers compared to the initial embeddings (figure 3). Simmilar groups are also less scattered and merged with other groups now, since the contrastive learning helped pulling them apart.

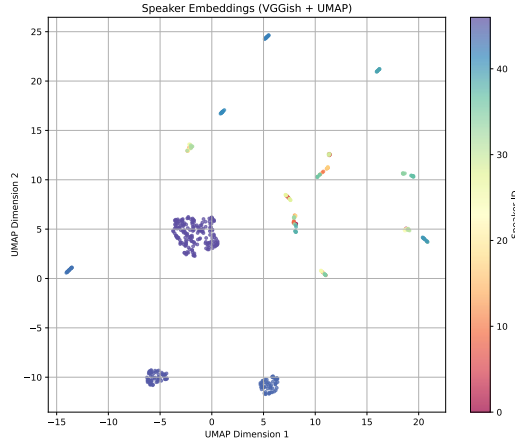


Figure 3: Umap visualization of the contrastive models embeddings.

3.4 Testing with Custom Voice Recordings

A crucial step in validating our implementation was testing it with voice recordings outside the training data. We processed our own voice samples, extracting embeddings and then passing them through the trained contrastive model for refinement. Subsequent UMAP visualization of these refined embeddings successfully clustered the custom recordings according to speaker, demonstrating the practical capability of our implemented approach to accurately group novel audio data by speaker (figure 4) [1].

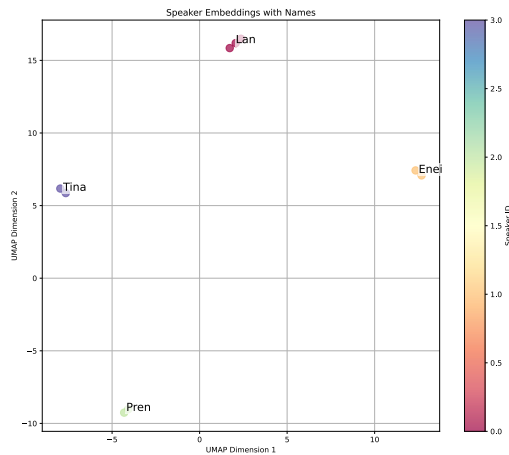


Figure 4: Testing with our own voice recordings.

4 CONCLUSION

Speaker identification in criminal investigations presents significant challenges, particularly when dealing with large volumes of noisy audio data and extensive databases. While state-of-the-art speaker recognition models offer powerful embedding capabilities, their direct application in forensic contexts often leads to an overwhelming number of false positives, hindering practical analysis. The approach presented in this paper tackles this limitation by integrating a clusterization technique with a rank-based scoring mechanism on top of embeddings extracted from a state-of-the-art ECAPA-TDNN model.

This novel combination, specifically tailored for forensic speaker identification, allows for the effective grouping of audio segments and a refined scoring process that prioritizes more reliable matches. Experimental results, particularly those utilizing real-world data from criminal investigations, demonstrate the practical value of this method. While identifying a smaller absolute number of potential speakers compared to a naive baseline, the proposed cluster-based approach significantly reduced the number of false positives and the overall pool of candidates requiring manual verification. This substantial reduction in potential matches makes the analysis process considerably more manageable and efficient for forensic investigators. Therefore, the proposed methodology represents a promising step towards enhancing the accuracy and feasibility of speaker identification in complex forensic scenarios.

Future work could explore further refinements in clustering, scoring parameters, and the integration of additional data modalities to improve performance and robustness.

REFERENCES

- [1] 2025. Our Implementation Code. Jupyter notebook and associated code for speaker clustering experiments. Based on experiments described in this document.
- [2] Brecht Desplanques, Jenthe Thienpondt, and Kris Demuyne. 2020. ECAPA-TDNN: Emphasized Channel Attention, Propagation and Aggregation in TDNN Based Speaker Verification. In *Interspeech 2020*. 3830–3834. <https://doi.org/10.21437/Interspeech.2020-2650>
- [3] Pinyan Li, Lap Man Hoi, Yapeng Wang, Xu Yang, and Sio Kei Im. 2025. Enhancing Speaker Recognition with CRET Model: a fusion of CONV2D, RESNET and ECAPA-TDNN. *EURASIP Journal on Audio, Speech, and Music Processing* 2025, 1 (14 Feb. 2025), 9. <https://doi.org/10.1186/s13636-025-00396-4>
- [4] Leland McInnes, John Healy, and James Melville. 2020. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. [arXiv:1802.03426 \[stat.ML\]](https://arxiv.org/abs/1802.03426) <https://arxiv.org/abs/1802.03426>
- [5] Antonio Artur Moura, Napoleão Nepomuceno, and Vasco Furtado. 2024. Enhancing speaker identification in criminal investigations through clusterization and rank-based scoring. *Forensic Science International: Digital Investigation* 49 (2024), 301765. <https://doi.org/10.1016/j.fsidi.2024.301765>
- [6] Weijun Pan, Shenhao Chen, Yidi Wang, Sheng Chen, and Xuan Wang. 2025. The Speaker Identification Model for Air-Ground Communication Based on a Parallel Branch Architecture. *Applied Sciences* 15 (03 2025), 2994. <https://doi.org/10.3390/app15062994>
- [7] Ahmad Zairi Zaidi, Chun Yong Chong, Zhe Jin, Rajendran Parthiban, and Ali Safaa Sadiq. 2021. Touch-based continuous mobile device authentication: State-of-the-art, challenges and opportunities. *Journal of Network and Computer Applications* 191 (2021), 103162. <https://doi.org/10.1016/j.jnca.2021.103162>

4 IoT in avtomobili / IoT and cars

Oblačna forenzična analiza Amazon iRobot Roomba sesalca

Jakob Adam Šircelj
Faculty of Computer and
Information Science
University of Ljubljana
js51840@student.uni-lj.si

Faris Delić
Faculty of Computer and
Information Science
University of Ljubljana
fd4786@student.uni-lj.si

Primož Gabrovec
Faculty of Computer and
Information Science
University of Ljubljana
pg1499@student.uni-lj.si

1. POVZETEK

Članek predstavlja možnost uporabe avtonomnih naprav, kot je Amazonov-ova iRobot Roomba, za preiskovanje zločinov. Kot mnogo drugo IoT naprav tudi Roomba zbira in hrani kup podatkov o uporabi, lokaciji, sobah in pohištvu v stanovanju, časovne žige, metapodatke in drugo. Ti podatki bi lahko bili zelo koristni pri raziskavi zločinov, saj nam bi lahko podali dragocene informacije o poteku dogodkov. Avtorji so najprej analizirali vse API dostopne točke, preko katerih Roomba komunicira s strežnikom, nato pa jih uporabili za razvoj nove odprtokodne aplikacije PyRoomba, ki izboljša uradno Roombino aplikacijo, tako da doda več podrobnosti in izriše bolj natančne 2D zemljevide prostorov, v katerih je sesalec aktiven. Aplikacija je bila nato testirana v več različnih prostorih z različno velikostjo in številom objektov, nato pa je bila še simulirana scena umora, kjer so na tla položili truplo in več nožev, ter testirali, ali jih bo Roomba zaznala. Preizkusi so se izkazali za uspešne, saj je sesalec identificiral dodatne ovire, ki se povezane z umorom.

2. KLJUČNE BESEDE

Digitalna forenzika, Forenzika interneta stvari, Oblačna forenzika, Amazon iRobot Roomba

3. UVOD

Z razvojem samostojnih naprav, kot so robotski sesalniki, se je odprlo več novih tem, najpogostejše so to vprašanja glede zasebnosti (naprave se povezujejo na internet, ter hranijo in pošiljajo veliko količino informacij - Amazon iRobot Roomba sesalec zbira natančne podatke o postavitvi pohištva in sob znotraj hiše). Avtorji članka, ki ga predstavljamo, pa so se lotili drugega vprašanja - ali je možno te podatke uporabiti za reševanje zločinov. To ni nekaj povsem novega, saj so preiskovalci v preteklosti že uporabljali podatke o govoru, ki so bili pridobljeni s strani Amazonove Alexa in shranjeni v oblaku, za rešitev umora Angie White. Še en tak primer je bil uporaba podatkov, zbranih z pametnega zvočnika Echo, leta 2018. Glavna razlika med tema

dvema primeroma in našim člankom pa je to, da so avtorji namesto uporabe mikrofонов izrabili podatke, ki jih ima Roomba o prostoru, in iz njih poskušali rekonstruirati sceno zločina in zaznati spremembe v okolici. Primer so trupla ali pa orožje, ki bi jih Roomba identificirala kot novo oviro. V ta namen so avtorji analizirali Roombino oblačno infrastrukturo ter njen API, napisali odprtokodno aplikacijo PyRoomba, ki to infrastrukturo uporablja, ter vse testirali na simuliranem kraju zločina.

4. SORODNA DELA

V zadnjih letih je naraščal interes za uporabo IoT naprav v digitalni forenziki, saj te naprave zbiraajo veliko količino podatkov, ki bi lahko bili koristni pri preiskovanju zločinov. Enako pomembna je tudi analiza citatov in literatura v zvezi z različnimi metodami plagiatstva, kar predstavlja ključno področje pri raziskovanju avtentičnosti dokumentov [4].

Ena izmed bolj znanih uporab pametnih naprav v forenziki je bila uporaba Amazonove Echo naprave v primeru reševanja umora, kjer so preiskovalci uporabili podatke, zbrane s pomočjo glasovne asistentke Alexa, za ugotavljanje časovnih zaporedij dogodkov. Podoben primer je bil uporabljen v letu 2018, ko so preiskovalci uporabili podatke pametnega zvočnika za rekonstrukcijo dogodkov [3]. V nasprotju s tem se v predstavljenem članku namesto mikrofонов uporablja navigacijski podatek sesalnika Roomba za rekonstrukcijo scene zločina.

Poleg tega obstajajo številne publikacije, ki se ukvarjajo z analizo prisotnosti objektov v prostoru in njihovo identifikacijo na podlagi senzorskih podatkov. Primer takega pristopa je analiza zamenjave obraza na slikah osebnih dokumentov, kjer so avtorji razvili algoritem za detekcijo sprememb v okolju in identifikacijo novih objektov [3]. To je zelo podobno metodologiji, ki jo uporabljajo avtorji predstavljenega članka za identifikacijo ovir, kot so trupla ali orožje.

Dodatno so bili izvedeni različni sistemi za preverjanje prisotnosti, ki temeljijo na senzorjih in biometričnih podatkih. Eden takšnih sistemov je predstavljen v [2], kjer avtorji opisujejo uporabo senzorskih naprav za avtomatizirano beleženje prisotnosti. Kljub različnim pristopom je cilj podoben - določiti prisotnost ali spremembe v prostoru na osnovi podatkov, zbranih s pametnih naprav.

Na področju obdelave podatkov iz IoT naprav pa so bile razvite tudi metode za dostop do oblačnih storitev in ek-

strakcijo forenzično relevantnih podatkov. Primer takega pristopa je PyRoomba, odprtokodna aplikacija, ki omogoča dostop do API-jev Roomba sesalnika in generira natančnejše zemljevide prostorov, kar je bistveno izboljšalo analizo [3]. Takšna orodja omogočajo večjo preglednost in natančnost pri rekonstrukciji dogodkov.

Zaradi hitrega razvoja pametnih naprav in povezljivosti medtem postaja vedno bolj pomembno, da se vzpostavi standardiziran pristop k obravnavanju podatkov iz teh naprav v forenzične namene. Obstajajo predlogi smernic za nadaljnje raziskave na tem področju, kot jih predlaga [5], kjer so predstavili okvir za digitalne kompetence državljanov in dodatna sorodna dela, usmerjena v izobraževanje in kapaciteto digitalne forenzike.

5. METODOLOGIJA

Avtorji članka [1] so delo razdelili na 4 dele: Priprava, identifikacija API, izdelovanje orodij in validacija.

5.1 Priprava

Uporabljen je bil sesalnik Roomba (Roomba serije J) in pametni mobilni telefon (Samsung Galaxy S10+), na katerem je bila nameščena uradna aplikacija iz Googlove Trgovine. Roomba je bila resetirana po proizvajalčevih navodilih. Poleg tega so na Windows računalnik namestili orodji apk-mitm in mitmproxy, ter ga povezali z Roombo.

5.2 Identifikacija API

Z uporabo mitmproxy orodja so prestrezali HTTPS promet med mobilno aplikacijo in Amazonovimi strežniki ter na ta način našli vse API zahteve, kjer so jih zanimala imena teh APIjev ter JSON odgovori. Če je odgovor vseboval zapise o času, uporabniške metapodatke, koordinate ali pa kakršnekoli podatke o prostoru, je bil ta API za raziskovalce relevanten, zato so ga dokumentirali.

V tem koraku je bilo najdenih več API zahtev, ki sploh nimajo dokumentacije. Najbolj relevantni so:

- Roomba's platform configuration: Iz nje lahko pridobimo podatke o regiji, osnovni URL naslov in informacijo, da Roomba uporablja MQTT protokol.
- Account provider API: vsebuje podatke o času registracije, kdaj se je uporabnik nazadnje prijavil, kako se je uporabnik prijavil (skozi socialno omrežje, skozi glavno spletno stran), ali je uporabnik na novo registriran in ali je email verificiran.
- Mission History: podatek o polnjenju, kako je bila zagnana (ročno, iz aplikacije...), koordinate zaznane umazanije, čas čiščenja, površina prostora, ki je bil počiščen, ter tudi navigacijski podatki.
- Map Detail Coordinates: metapodatki o zemljevidu, koordinate mej prostora, ki ga čisti, regije, ki jih je Roomba našla, tip in koordinate objektov, ki jih je Roomba našla med delovanjem, ovire, koordinate vrat, in več.
- Obstacle Detection: Vrača kriptirane slike, ki so poimenovane z imenom objekta, prikazan na njih

- Image Retrieval Link za nalaganje slik. Pomembno si je zapomniti, da je link na voljo samo za približno 36 ur.

5.3 Izdelovanje orodja PyRoomba

V tem koraku so prej omenjene APIje izkoristili za izdelavo nove spletne aplikacije PyRoomba, ki je napisana v Python 3.11. Ta aplikacija to stori brez direktne interakcije s samo napravo, kar zagotovi forenzično integriteto. Aplikacija lahko generira zemljevide iz podatkov o navigaciji, prikaže časovne žige in prikaže objekte, ki jih je sesalec našel.

5.4 Validacija

Za testiranje sta bili uporabljeni dve sobi, ena večja in ena manjša, v katero je bilo postavljeno več tipičnih hišnih pripomočkov. Skozi večjo sobo je Roomba šla štirikrat, skozi manjšo pa dvakrat. Na koncu so primerjali rezultate orodja PyRoomba z dejansko sobo in pa z zemljevidom, ki ga je generirala uradna Amazonova aplikacija.

6. PYROOMBA

Orodje PyRoomba je bilo narejeno zaradi pomankljivosti podatkov standardne mobilne aplikacije. Le-te PyRoomba pridobi direktno iz oblaka, brez da bi aktiviral samo Roombo, kar bi lahko vodilo do okvare podatkov. Aplikacija nam ponuja natančne informacije o zgodovini in podrobnostih, ki jih Roomba zapiše med čiščenjem, poleg tega pa tudi izriše 2D zemljevid, na katerem so vsi podatki o objektih in časovnih žigih. Ta zemljevid je bistveno bolj natančen kot tisti, ki je na voljo v uradni aplikaciji. PyRoomba podpira tudi prenos vseh podatkov o identificiranih objektih, kar lahko pomaga pri natančni analizi. Avtorji tudi verjamejo, da bo program na voljo več časa, saj ob začetku preveri API spremembe in se nanje prilagodi. Za avtentikacijo uporablja Amazon AWS, kar zagotovi, da so podatki zanesljivi in varni. Druga prednost pred uradno aplikacijo je dostop do vseh preteklih Roombinih misij (Amazonova aplikacija hrani le zadnjih 30), edina izjema je v primeru, da je bila Roomba ponastavljena na tovarniške nastavitve. V tem primeru do podatkov ni mogoče dostopati. Podatki, ki so na voljo, vključujejo: začetno točko, dolžino misije, čas začetka in zaključka, status, pokrito področje, detekcijo umazanije in več.

7. ANALIZA

Analiza je bila narejena v dveh prostorih: Manjši spalnici in večjem stanovanju. Na podlagi teh dveh prostorov je bila narejena primerjava med rezultati, pridobljenimi z uporabo PyRoomba in uradne aplikacije.

V večjemu stanovanju je bilo kar nekaj večjih objektov, med drugim postelja, hladilnik, kuhinjske omare, pečica ter več manjših objektov, kot so kabli, dva para čevljev in vrečke iz blaga. Skozi ta prostor je Roomba šla štirikrat, dvakrat na dan v dveh dneh. V manjši spalnici je bila postelja, miza, omara z vrati, en par čevljev in nekaj kablov. Skozi ta prostor je sesalec šel le dvakrat.

Glavni cilj je bil ugotoviti, kako natančno in konsistentno lahko Roomba prepozna različne objekte. Poleg tega so avtorji želeli tudi odgovoriti na vprašanje, ali bo več zaporednih ponovitev izboljšalo rezultate, našlo prej še neodkrita

Roomba endpoint APIs and forensic relevance.

API Category	Undocumented Roomba APIs	Description	Forensic Relevancy
Platform Configuration Account Provider	https://disc-prod.iot.irobotapi.com/v1/discover/endpoints?country_code=US https://accounts.us1.gigya.com/accounts.getAccountInfo	Configuration details about the iRobot IoT platform User account and profile information	Provides insight into the infrastructure and potential data storage locations. Offers a timeline of account activity, login methods, and verification status, which can be used to track user interactions and potential unauthorized access.
Mission History	https://auth3.prod.iot.irobotapi.com/v1/robotid/missionhistory?filterType=app_id	Comprehensive dataset about Roomba's activities	Offers insights into Roomba's operations, including Wi-Fi connections, charging times, and cleaning durations. This can help establish timelines and identify anomalies.
Roomba Map Detail Coordinates	https://auth3.prod.iot.irobotapi.com/v1/robotid/pmaps/pmap_id/versions/pmapv_id/	Information regarding Roomba's navigational coordinates to reconstruct the map	Offers a digital footprint, enabling the reconstruction of events, verification of objects, detection of environmental changes.
Obstacle Detected	https://auth1.prod.iot.irobotapi.com/v1/robots/340EE4928078487E853BF9F3180A3898/imageupload/metadata	Details about the type of obstacle	Forensically relevant in reconstructing events, understanding environmental dynamics, and establishing potential evidence in investigations.
Image Retrieval	https://auth1.prod.iot.irobotapi.com/v1/robots/340EE4928078487E853BF9F3180A3898/imageupload/imagesurl	Link to download Encrypted Images captured by Roomba	Images to see the Obstacles presented in an Environment.

Figure 1: V tabeli so prikazani zanimivi APIji, ki jih Roomba uporablja med čiščenjem in nam dajo zanimive podatke za digitalno forenziko

ovire ali pa celo neuspešno našel objekte, ki jih je v prejšnjih ponovitvah že našel.

8. PREDSTAVITEV REZULTATOV

Za sesanje večjega stanovanja je Roomba potrebovala med 26 in 43 minut. Med prvima dvema testiranjema je delovala 43 in 26 minut, v obeh primerih je našla le 5 objektov. Med tretjim testom je našla 8 objektov v 36 minutah, v zadnjem pa kar 11 v 33 minutah. V manjši spalnici je najprej delovala 22 minut in ni našla nobenega objekta, v drugem poizkusu pa je v 15 minutah našla 4 objekte.

Na koncu so avtorji želeli simulirati še umor. Najprej so Roombo spustili v sobo in pustili, da jo je dokumentirala. To je trajalo 29 minut, sesalec pa je pokrtil približno 14,4 kvadratnih metrov. V tem času je našla 1 blago, pet kablov, dva čevlja, 2 vrečki iz blaga in skledo za domače živali. Po zaključku pa so avtorji za simulacijo zločina v sobo dodali truplo ter okoli njega več nožev, ter še enkrat aktivirali sesalec. Tokrat je Roomba v 27 minutah pokrila slabih 12 kvadratnih metrov, dodala pa je dve dodatni "oviri": te dve sta bili povezani s truplom in dodanimi noži.

Iz preizkusov je sledilo več zaključkov:

- V vsaki iteraciji je Roomba našla več objektov.
- V manjših sobah je bila Roomba bolj učinkovita, saj je dosegla boljšo pokritost in posledično boljše našla razne objekte.
- Če primerjamo generiran zemljevid po uporabi PyRoomba aplikacije s tistim, ki ga nariše uradna aplikacija, se izkaže, da je PyRoombin bistveno boljši in prikaže več detajlov.

Avtorji verjamejo, da se lahko PyRoombine zemljevide uporabi za rekonstrukcijo scen zločina ter zaradi hranjenja časovnih žigov tudi za prikaz zaporedja dogodkov. Zazna lahko

tudi dodajanje ali odstranitev objektov iz sobe ob nekem času. Tudi dejstvo, da se je pokritost med dvema testoma zmanjšala za dva in pol kvadratna metra, je dober indikator za večjo oviro oz. v našem primeru truplo. Kljub vsemu pa potrebujemo tudi človeka, ki bo ročno šel čez rezultate, saj sesalnik vse le označi kot "oviro", natančna interpretacija pa se je delala na roke.

9. OMEJITVE

Ena izmed glavnih omejitev je potreba, da imamo uporabniške podatke, s katerimi se lahko prijavimo v Amazonov sistem za Roombo. Poleg tega se uporablja Amazonov oblak, namesto da bi podatke prebral iz same naprave. Ker niso imeli glavnega ključa za dekripcijo, jim ni uspelo dekriptirati slik Roombe, vendar so bile že oznake slike zelo koristne. Poleg tega pa je pomembno še poudariti, da so avtorji vse testirali z Roombinim modelom J, in niso testirali nobenega drugega modela.

10. PRILOGE

Priložili smo še dve sliki, ki prikazujeta delovanje aplikacije PyRoomba. Na sliki 2 so prikazane vse štiri iteracije sesalnika, ko deluje v stanovanju, na sliki 3 pa je prikazana simulirana lokacija umora, ter zemljevid, ki je bil generiran s pomočjo PyRoombe po tem, ko je sesalec opravil misijo v sobi pred in po umoru.

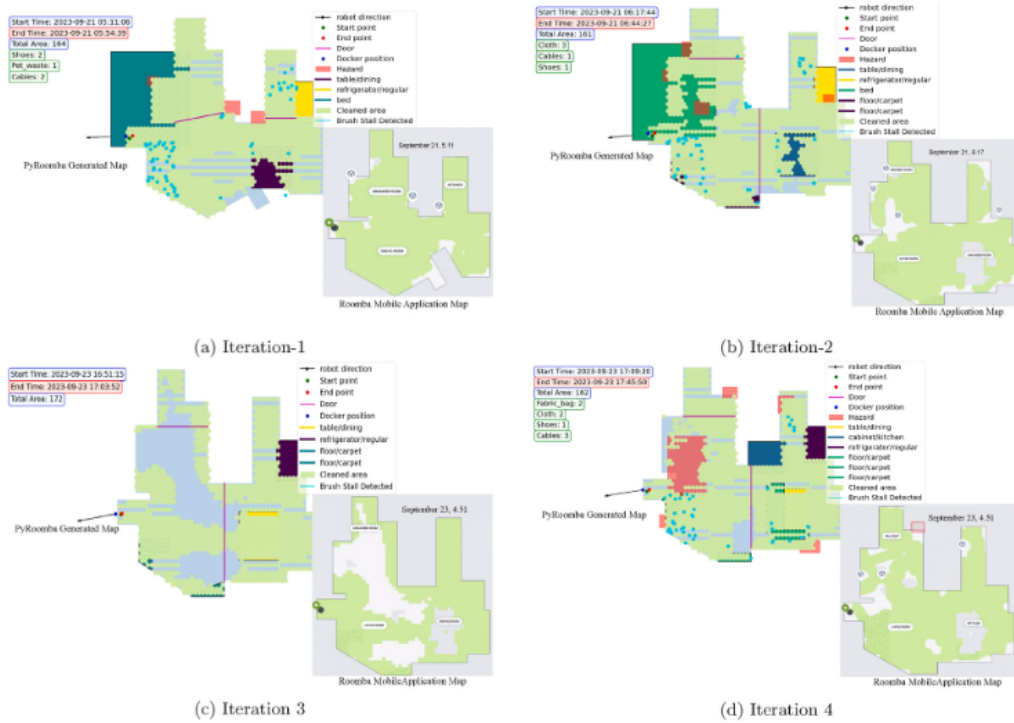


Figure 2: Na sliki so prikazani zemljevidi, ki jih je PyRoomba ustvarila po vsaki iteraciji v večjem prostoru.

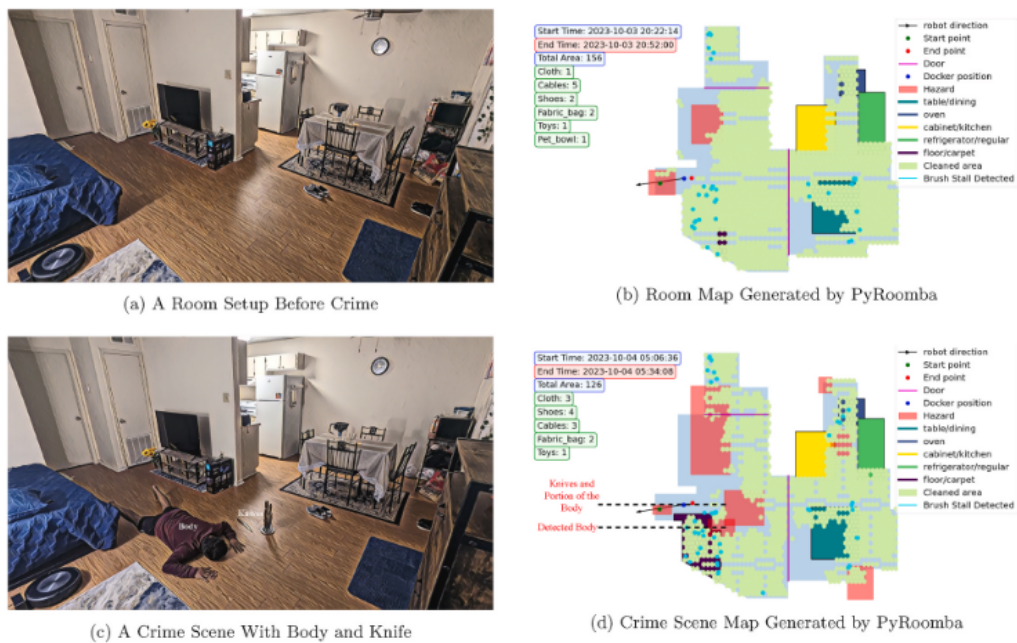


Figure 3: Na sliki so prikazani rezultati pred in po simuliranem umoru.

11. REFERENCES

- [1] I. B. A. M. W. Abdur Rahman Onik, Ruba Alsmadi. So fresh, so clean: Cloud forensic analysis of the amazon irobot roomba vacuum. *Forensic Science International: Digital Investigation*, 2024.
- [2] Arvind and K. S. Chethan. Face recognition based smart attendance system. In *International Conference on Intelligent Engineering and Management (ICIEM)*, pages 354–357, June 2020.
- [3] S. Friedl and G. Pernul. Device forensics in smart homes: Insights on advances, challenges and future directions. In *Transactions on Large-Scale Data- and Knowledge-Centered Systems LVII*, pages 68–98. Springer, Berlin, Heidelberg, 2024.
- [4] A. K. M. N. Islam, M. A. Al-Faruque, M. A. Sayem, M. M. Rahman, M. A. Kaium, M. R. Hasan, and M. R. Islam. A comprehensive digital forensic and incident response management model for smart agriculture networks. *Frontiers in Computer Science*, 4, 2022.
- [5] V. R. Kebande and A. I. Awad. Industrial internet of things ecosystems security and digital forensics: Achievements, open challenges, and future directions. *ACM Computing Surveys*, 2024.

Hit and run: Forensic vehicle event reconstruction through driver-based cloud data from Progressive’s snapshot application

Veljko Dudić

David Aristovnik

Miha Frangež

ABSTRACT

This paper provides a summary and critical analysis of a pioneering study by Onik et al. (2024) on the forensic analysis of the *Snapshot* application from *Progressive Insurance*. Driving Insurance Applications (DIAs) gather extensive data on driver behavior, yet their cloud-based data repositories remain a largely untapped resource for forensic investigations. The original study developed *PyShot*—an open-source Python tool—to extract granular data from the Progressive cloud, much of which is inaccessible via the standard user interface. It investigated the accuracy of Snapshot’s location and speed data, its resilience against GPS spoofing, and its ability to provide detailed event information for reconstructing a simulated hit-and-run scenario. This seminar paper evaluates the methodology and findings of that work, contextualizes it within the broader field of vehicle and cloud forensics, and discusses its implications. The findings confirm that telematics cloud data offers a robust and reliable new avenue for investigating traffic incidents and crimes, even when faced with user tampering.

General Terms

Digital forensics, Forensic investigation techniques, Event Reconstruction, Cloud data analysis, Evidence integrity and reconstruction

Keywords

Driving insurance application (DIA), Vehicle forensics, Digital forensics, Snapshot, Driving patterns, Cloud forensics

1. INTRODUCTION

Digital forensics is undergoing rapid evolution due to the increasing integration of Internet of Things (IoT) devices into everyday applications. One notable development is the use of driving insurance applications (DIAs), such as Progressive’s Snapshot, which collects extensive data on driving behavior and vehicle performance through telematic devices [1]. Originally designed to reward safe driving, the under-

lying cloud data repository presents significant potential for forensic investigations.

This seminar paper presents a summary and analysis of a novel methodology introduced by Onik et al. [1] for reconstructing vehicular incidents using cloud data from the Snapshot application. Their work introduces *PyShot*, an open-source Python tool, to extract a comprehensive set of driving data from Progressive’s cloud servers, going beyond the limited details available through the mobile application. The dataset includes precise information on trip start and end points, speed variations, hard braking events, and indicators of distracted driving.

The original study, and therefore our analysis, is structured around four key research questions:

1. **RQ1:** How accurate is the location and speed data from Snapshot?
2. **RQ2:** What additional forensic details can be extracted from the Progressive Cloud?
3. **RQ3:** Can fake GPS data alter trip records?
4. **RQ4:** Can a hit-and-run scenario be reconstructed from the trip event details?

By examining the methods used to answer these questions, this paper evaluates the validity of using DIA cloud data as a reliable source of digital evidence and discusses its position within the wider landscape of digital forensics.

2. RELATED WORK

Prior efforts in this domain fall into two main categories that inform this study.

- **Location-tracking devices and apps.** Bluetooth tags, navigation services, and family-sharing platforms collect and store rich GPS history that can be parsed with open-source tools. Forensic analyses of the Tile tracker, smartphone mapping apps, and Life360/iSharing demonstrate recovery of routes, coordinates, and user metadata, with experiments showing meter-level accuracy for positioning logs.[2, 3, 4, 5]
- **Automotive digital evidence.** Beyond event data recorders, investigators tap vehicle data from multiple layers:

man-in-the-middle interception of vehicle-to-infrastructure traffic, artifacts left on paired phones by Android Auto or Apple CarPlay, and cloud records kept by OEM companion apps. Studies reveal trip details, driver actions, and remote-control events, while also noting the proprietary, non-standard nature of in-vehicle formats.[6, 7, 8, 9]

Building on these trends, Onik et al. (2024) apply IoT-style network interception to Progressive Snapshot telematics, bridging mobile/cloud forensics, and vehicular event reconstruction, as in their earlier cloud work on the Roomba vacuum.[1, 10]

3. METHODOLOGY

3.1 Research Questions

The forensic analysis presented in the article addresses four critical research questions:

1. **Accuracy of Location and Speed Data:** How precise is the location and speed information recorded by the Snapshot application compared to traditional GPS data sources?
2. **Extraction of Forensically Relevant Data:** What additional forensic details can be retrieved from Progressive’s cloud that are not accessible via the mobile interface, and how do these details enhance the investigative process?
3. **Resilience Against Anti-Forensic Techniques:** Can anti-forensic techniques, such as the injection of fake location data, effectively manipulate trip details, and does the cloud-based data maintain its integrity against such attempts?
4. **Reconstruction of Hit-and-Run Scenarios:** Is it feasible to reconstruct a hit-and-run incident by analyzing detailed trip event data, thereby establishing a clear timeline and spatial map of the event?

3.2 Preparation

The preparation phase involved acquiring the necessary hardware and installing the required software (see Table 1). An Android smartphone (Samsung Galaxy S10+) was obtained and the Progressive Insurance Snapshot app was downloaded from the Google Play Store. The apk-mitm tool was used to modify the APK - decoding it, adjusting its network configuration, disabling certificate pinning, and re-encoding it so that the modified app could be installed on the test phone (see Fig 1). A network connection was then established between the test phone and a Windows PC via a mobile hotspot, and *mitmproxy* was installed on the PC. The *mitmproxy* certificate was integrated on both devices to allow monitoring of encrypted traffic between the app and its servers.

Hardware/Software	Use	Company	Version
Galaxy S10+	Progressive Android Application	Samsung	Android 13.0.0
iPhone 6s	Progressive iOS Application	Apple	iOS 15.7.1
Galaxy Tab A7 Lite	GPS Logger	Samsung	Android 13.0.0
Windows PC	MITM Web	Microsoft	Windows 10.0.22621
Progressive App	Application Analysis	Progressive	Version-3.92
APK-MITM Tool	APK modification for HTTPS	Open Source	v1.2.1
MITMProxy Tool	HTTPS traffic interception	Open Source	mitmproxy 10.1.0
GPS Logger App	Tracks location and speed	BasicAirData	Version-129
Fake GPS Spoofer	Simulates fake location data	IncorporateApps	Version 5.8.1

Table 1: List of apparatus

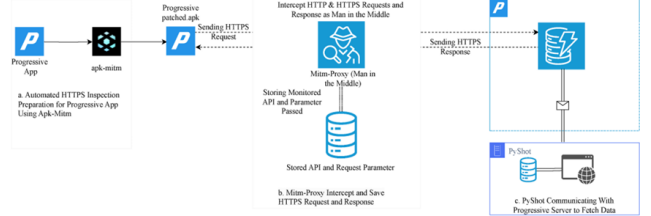


Figure 1: Network traffic analysis of progressive application

3.3 Identification of Undocumented APIs

Mitmweb, the web interface for *mitmproxy*, was used to monitor HTTPS communications between the Snapshot application and its server. *Mitmproxy* is a tool for analyzing network traffic using a Man-in-the-Middle, which is necessary to decrypt the TLS-encrypted traffic of modern mobile applications. API requests were identified based on JSON responses that contained timestamps, user metadata, coordinates, or related information. Each significant API request was recorded along with its headers, query parameters, and responses as sequential "Flows." These insights were later utilized in the development of the forensic tool, PyShot.

3.4 Validation

Controlled experimental drives were conducted in an empty parking area using a vehicle and a mobile device. Three distinct trips were performed: one under standard driving conditions, and two simulating phone usage with varied maneuvers such as turns, hard braking, and non-braking turns. The maps generated by PyShot were compared against both the ground truth data and those produced by the Snapshot application to verify accuracy and detail.

3.5 PyShot: A Forensic Tool

PyShot is a forensic tool developed in Python 3.11 that extracts detailed driving data from Progressive’s cloud using the discovered APIs. The tool collects comprehensive trip information—including start and end coordinates and specific incident events such as hard braking, dangerous maneuvers, and phone usage—with precise timestamps and locations. This approach enhances forensic investigations by providing a more detailed account of driving behavior than the native Snapshot application.

4. UNDOCUMENTED SNAPSHOT APIS

This section describes the undocumented APIs discovered during the analysis of the Progressive Snapshot application. The APIs return JSON-formatted data containing essential forensic information, such as timestamps, coordinates, and user metadata, which are critical for reconstructing driving events.

4.1 Login API

The Progressive API login process follows a structured procedure. Initially, a POST request is sent to the Login API (refer to Table 2, row 1) to authenticate the user. This request includes key authorization headers such as X-PGRIdentifier, User-Agent, api-key, and Host—to ensure secure identification. The login credentials are provided in a JSON-formatted body, and the API's JSON response delivers an accessToken along with links for further interactions. This accessToken is essential for accessing user-specific resources within the Progressive system.

4.2 Telematics Registration API

The Telematics Registration API initiates the registration process by accepting a mobile phone number. Upon submission, the API sends a One-Time Password (OTP) to the provided number, thereby beginning the authentication process and enabling access to the Snapshot data.

4.3 Telematics Authentication API

After receiving the OTP, the Telematics Authentication API validates the phone number and OTP via a secure POST request. The response yields key data elements such as an API token, the user's first name, a mobile participant ID, and a list of associated vehicles, which are necessary for subsequent authentication steps.

4.4 Device and Driver Authentication API

Utilizing information from the previous authentication steps, the Device and Driver Authentication API sends a JSON request that includes device details and the access token. The response returns new identifiers, including a device ID and driver ID, along with a renewed access token, all of which are essential for retrieving detailed trip data.

4.5 Trip Details API

The Trip Details API is used to obtain comprehensive trip metrics. By providing the driver ID, start and end times (as Unix timestamps), and a valid access token, the API returns detailed information such as trip coordinates, timestamps, event logs (including hard braking, acceleration, and distractions) and other critical data for forensic reconstruction of driving events.

4.6 PyShot: A DIA Forensic Tool

PyShot is an open-source forensic tool developed in Python 3.11 to extract detailed driving data from Progressive's Snapshot cloud service. Designed for forensic accuracy and data integrity, PyShot retrieves comprehensive trip information including start and end coordinates, timestamps, and a variety of driving events such as hard braking, acceleration, and distractions.

Key features of PyShot include:

- **Telematics Registration and OTP Authentication:** Securely authenticates users through one-time password validation.
- **Local Database Management:** Organizes and stores trip data locally to reduce reliance on continuous cloud connectivity.

- **Comprehensive Trip History:** Captures detailed logs of all trips, including specific events not available via the native Snapshot application.
- **Timeline Analysis:** Constructs a chronological sequence of driving events, facilitating accurate forensic reconstruction.
- **Map Visualization:** Leverages the Google Maps API to visually correlate trip events with precise geographical locations. Fig. 2 presents a visual representation of the PyShot interface.

These features enable enhanced forensic analysis by providing a robust framework for reconstructing vehicular incidents with high precision.

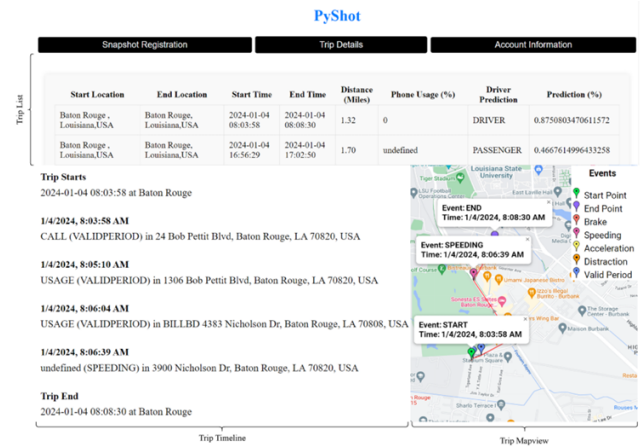


Figure 2: Visualization of the main features of the PyShot application.

5. EVALUATION AND DISCUSSION

This section presents experiments that demonstrate the forensic capabilities of Snapshot and the Progressive cloud data accessed via PyShot. The analysis addresses four research questions: the accuracy of location and speed data, the forensic relevance of cloud data from undocumented APIs, resilience against anti-forensic techniques such as location spoofing and the feasibility of reconstructing specific driving incidents.

5.1 RQ1: Location Data and Speed Accuracy Analysis

Experimental drives were conducted to compare the geospatial and velocity data recorded by the Snapshot application with the data obtained from a dedicated GPS logger. Detailed graphical analyses revealed that the recorded location and speed values closely matched the ground truth measurements. The study considered factors such as data sampling frequency and environmental variability, and the minimal discrepancies observed validate the high accuracy and forensic reliability of the Snapshot data.

5.2 RQ2: Forensic Data Beyond Mobile Application Access

Cloud data was analyzed using PyShot, and the resulting trip events were compared with those extracted from the Snapshot app on Android and iOS platforms. The iOS version provided only basic details (start and end points), while the Android version included additional information, such as braking and acceleration events with timestamps. In contrast, PyShot captured a broader range of events, including hard braking, acceleration, dangerous maneuvers and distracted driving, all recorded on Progressive’s servers (see Table 2).

Evidence Type	iOS	Android	PyShot
Start Point	✓	✓	✓
End Point	✓	✓	✓
Hard Brakes	✗	✓	✓
Accelerations	✗	✓	✓
Speeding	✗	✗	✓
Distractions	✗	✗	✓
Phone Usage & Calls	✗	✓	✓

Table 2: Evidence of Trip Events: iOS vs. Android vs. PyShot.

5.3 RQ3: Resilience Against Anti-Forensic Techniques

The resilience of the system was evaluated by simulating GPS spoofing to inject false location data. Despite these deliberate manipulation attempts, the cloud-based records accessed via PyShot consistently maintained accurate trip details, including precise event timings and correct geolocation data. This robustness under adversarial conditions confirms the strong integrity of the data and its suitability for forensic analysis, even in the face of anti-forensic techniques.

5.4 Transit Mode Analysis

The analysis of transit modes revealed that the Snapshot application reliably distinguishes between various travel states, such as differentiating between a driver and a passenger. Each trip is assigned a transit mode classification accompanied by a confidence value, which further substantiates the accuracy of the prediction. This reliable mode classification enriches the overall forensic assessment by providing additional context to the driving behavior, thereby supporting more nuanced interpretations in investigations.

6. CASE STUDY: HIT-AND-RUN

A simulated hit-and-run scenario was designed to evaluate the forensic application of cloud data accessed via PyShot. In this experiment, a mannequin was positioned at a pre-determined location to represent a collision victim. The cloud data recorded a sequence of events including sudden hard braking at the impact point followed by rapid acceleration as the vehicle fled the scene. Mapping precise timestamps and geolocation coordinates enabled a chronological reconstruction of the incident, which correlated well with supplementary video evidence and physical measurements.

7. LIMITATIONS

Several limitations are acknowledged in the current study. The focus on a single insurance application restricts the generalizability of the findings across other platforms. Addition-

ally, reliance on undocumented APIs introduces potential vulnerabilities, as future updates to the application could alter data structures and impede forensic data retrieval. The controlled experimental environment may also not fully reflect the complexities encountered in real-world scenarios, suggesting the need for further research under diverse conditions.

8. CONCLUSIONS

The investigation by Onik et al. [1] demonstrates that Progressive’s Snapshot cloud data, accessed via the custom PyShot tool, is a highly detailed and reliable resource for vehicular forensic analysis. The study successfully confirms that cloud-based records capture critical driving events with high accuracy and are resilient against common anti-forensic techniques. The effective reconstruction of a simulated hit-and-run scenario powerfully underscores the practical forensic value of this approach. By shifting the focus from volatile on-device data to more robust cloud repositories, this work contributes significantly to the advancement of digital forensic methodologies and opens promising avenues for the investigation and reconstruction of vehicular incidents.

9. REFERENCES

- [1] Onik, Abdur Rashed, Spinosa, Trevor T., Asad, Abdulla M., and Baggili, Ibrahim, “Hit and run: Forensic vehicle event reconstruction through driver-based cloud data from progressive’s snapshot application,” *Forensic Science International: Digital Investigation*, vol. 49, p. 301762, 07 2024.
- [2] Pace, Lauren R., Salmon, Lynn A., Bowen, Christopher J., Baggili, Ibrahim, and Richard III, Golden G., “Every step you take, i’ll be tracking you: Forensic analysis of the tile tracker application,” *Forensic Science International: Digital Investigation*, vol. 45, p. 301559, 2023.
- [3] J. Moore, I. Baggili, and F. Breiteringer, “Find me if you can: Mobile gps mapping applications’ forensic analysis & snavp, the open source, modular, extensible parser,” *Journal of Digital Forensics, Security and Law*, vol. 12, no. 1, 2017.
- [4] J. Bays and U. Karabiyik, “Forensic analysis of third-party location applications in android and ios,” in *Proceedings of the IEEE INFOCOM Workshops*, 2019, pp. 524–529.
- [5] G. A. Listi, M. H. Manhein, and M. Leitner, “Use of the global positioning system in the field recovery of scattered human remains,” *Journal of Forensic Sciences*, vol. 52, no. 1, pp. 11–15, 2007.
- [6] X. Feng, E. S. Dawam, and D. Li, “Autonomous vehicles’ forensics in smart cities,” in *Proceedings of IEEE Smart-World/SCALCOM/UIC/ATC/CBDCCom/IOP/SCI*, 2019, pp. 1688–1694.
- [7] A. Mahr, R. Serafin, C. Grajeda, and I. Baggili, “Auto-parser: Android auto and apple carplay forensics,” in *Digital Forensics and Cyber Crime (ICDF2C 2021)*. Springer, 2022, pp. 52–71.
- [8] S. Ebbers, F. Ising, C. Saatjohann, and S. Schinzel, “Grand theft app: Digital forensics of vehicle assistant apps,” in *Proceedings of the 16th International Conference on Availability, Reliability and Security*

- (*ARES*), 2021, pp. 1–6.
- [9] D. Kopencova and R. Rak, “Issues of vehicle digital forensics,” in *Proceedings of the XII International Science-Technical Conference — Automotive Safety 2020*, 2020, pp. 1–6.
- [10] Onik, Abdur Rashed, R. Alsmadi, I. Baggili, and A. Webb, “So fresh, so clean: Cloud forensic analysis of the amazon irobot roomba vacuum,” *Forensic Science International: Digital Investigation*, vol. 48S, p. 301686, 2024.

Grand Theft API: A Forensic Analysis Of Vehicle Cloud Data

Luka Maček

Faculty of Computer and Information Science
Ljubljana, Slovenia
lm9231@student.uni-lj.si

Dimitrije Stefanović

Faculty of Computer and Information Science
Ljubljana, Slovenia
ds35686@student.uni-lj.si

Ana Bračić

Faculty of Computer and Information Science
Ljubljana, Slovenia
ab1165@student.uni-lj.si

Sara Mihalič

Faculty of Computer and Information Science
Ljubljana, Slovenia
sm0770@student.uni-lj.si

Abstract

Modern cars collect and send meaningful amount of data to manufacturer cloud systems. This situation creates new possibilities for modern digital forensic investigations. In this paper, we look at how investigators can get vehicle cloud data directly through manufacturer APIs, using login details taken from suspects' mobile devices. We explain the basics of vehicle forensics, look at current methods for getting vehicle cloud data, and show how API-based methods work in practice. After analyzing 23 different vehicle apps and manually testing with Mercedes-Benz and BMW cars, we found that a significant amount of vehicle data can be accessed through these APIs. This includes current car status, location, and personal info. Our research shows that using APIs to get vehicle cloud data gives valuable information for forensic investigations. This approach could change how evidence is collected from modern vehicles. It represents a step forward in digital vehicle forensics and allows access to both real-time and historical vehicle data while following proper forensic principles.

Keywords

Vehicle Forensics, API-based Acquisition, Digital Forensics, Telemetry Data, Cloud Forensics

1 Introduction

Modern cars have evolved into advanced connected computing systems. In addition to mechanical systems, they feature built-in computers and sensors that continuously collect data regarding driving behavior, vehicle status, location, or fuel use. While some of this data remains stored in the car, much of it is stored in manufacturer cloud systems. Car owners can view this data through Vehicle Assistant Apps (VAA) on their smartphones. This data proves especially useful for law enforcement purposes. It can be useful in locating stolen cars, identifying suspects connected to vehicles, verifying witness statements, investigating insurance claims, and analyzing car accidents. Digital evidence of this nature can significantly influence the outcome of criminal investigations.

Traditionally, there are two methods for obtaining vehicle data. The first method involves extracting information directly from the car systems. This requires physical access to the car and only works on data that is stored locally. The second method regards the information stored on the manufacturer cloud systems. This approach involves cooperation from the manufacturer and often

takes a long time, even with legal permission. Both methods have significant limitations and can slow down investigations.

Our paper analyzes Ebbers et al.'s [2] research on API-based forensic acquisition of vehicle cloud data for forensic purposes and explores a third approach: accessing vehicle cloud data directly through manufacturer APIs using login details or tokens taken from a suspect's smartphone. This method could give investigators immediate access to vehicle data when they legally have temporary access to a suspect's phone containing valid credentials.

The contributions of the paper include a general method for accessing vehicle cloud data using credentials from analyzed smartphones, analyzing vehicle assistant apps to identify accessible data categories, providing actual test results from Mercedes-Benz and BMW vehicles, and discussing forensic implications and potential applications in criminal investigations.

2 Related Work

A lot of research already lays down the groundwork for automotive forensics, like Gomez Buquerin, Corbett and Hof (2021) [3] have stated, there are many different types of data forensics that would be interested in when analyzing a vehicle. While the process and data classification is something that has already been determined, the reliability of acquiring and securing this data has always been a challenge.

2.1 Traditional tooling and work

With many researchers such as Litman (2017) [6] stating the complexity and the possibility that the computer systems inside cars could become much more valuable, there is a growing demand for different and generalized approaches to data collection.

With many different car brands and case studies (Jacobs et al. (2017) [5]; Stabili et al. (2025) [10]), there is an interest in creating and collecting comprehensive tools and wide field research to be able to successfully extract from many different systems.

Classic tools often rely on access to the internal systems inside the car (Nilsson and Larson (2009) [8]) such as the On-Board diagnostics or the Infotainment system to extract important information about the vehicle and its usage. However, with the many complexities of modern systems (Gomez Buquerin, Corbett and Hof (2021) [3]) and their rapid evolution, we have to consider the systems that can communicate to the car, directly or indirectly, and include them in the scope of the investigation.

2.2 Expanding the Scope

Communication between the vehicle and its components, the drivers' phone (Hossain et al. (2017) [4]) or even directly with the company backend system is something that is important to consider. Ebbers et al. (2024) [2] ("Grand Theft API") showed how smartphone Vehicle Assistant Apps (VAAs) contain driving data: by extracting application data and OEM subject-access requests. They reconstructed trips, fuel logs, and lock/unlock events from eight different brands.

Separately, cloud-forensic research and automotive cybersecurity highlight new data flows and potential risks. On the other hand, cloud-native digital artifacts (Roussev et al. 2016 [9]) suggest inertness of the data being investigated, given they are accessed remotely, while still undermining the potentially critical information on the evolution of the file in question.

Law-enforcement interest is high – as Ebbers et al., 2024 [2] note, vehicles continuously upload mileage, GPS data, tire pressure, etc., to OEM clouds. But obtaining this cloud data via official channels is slow and cumbersome.

2.3 Papers relevance

The paper in question of the analysis, as well as our coverage of the topic, fills the gap in prior work. Unlike Jacobs et al. (2017) [5] or Stabili et al. (2025) [10], which needed physical extraction, Grand Theft API shows law enforcement can now pull historic telemetry remotely using the suspect's credentials. In this way, we present a new forensic methodology that leverages vehicle cloud services: continuing earlier studies of in-car and smartphone evidence.

3 Methodology

The methodology described by Ebbers et al. [2] for accessing vehicle cloud data follows a structured approach that integrates elements from mobile, cloud, and vehicle forensics. The process involves:

- (1) Identifying VAAs and their capabilities
- (2) Understanding the authentication mechanisms
- (3) Locating and extracting tokens or credentials
- (4) Accessing cloud APIs directly for data acquisition

This approach allows investigators to access the same data available to vehicle owners through their applications, but in a forensically sound manner that documents all interactions and preserves data integrity.

3.1 Vehicle-User Connection Process

To access a vehicle's data through a VAA, users must first create an account on the manufacturer's platform. This requires providing personal information such as an email address, phone number, name, and sometimes address. The user must then link their vehicle to this account through a process that varies between manufacturers.

The vehicle linking process typically involves methods such as scanning a QR code displayed on the vehicle's infotainment system, entering the Vehicle Identification Number (VIN), confirming a transaction number on the infotainment system, submitting proof of ownership, or interacting with vehicle controls.

These authentication processes are designed to ensure that only legitimate owners can access vehicle data, but they also create potential access points for forensic investigators when legally authorized.

3.2 Vehicle Assistant App Analysis

The research took into account 23 different VAAs from major manufacturers, including BMW, Dacia, Ford, Hyundai, Mercedes, and Tesla. This analysis identified the capabilities of each application and the types of data potentially accessible.

The analysis process involved:

- (1) Installing applications on test devices
- (2) Setting up Man-in-the-Middle proxies to capture API traffic
- (3) Bypassing security measures like certificate pinning and root detection
- (4) Mapping API endpoints and authentication mechanisms

This process revealed significant variations in authentication methods, API structures, and data availability across manufacturers.

3.3 Token Extraction Methods

Most vehicle apps use OAuth 2.0 for API access, saving access and refresh tokens on the user's phone. These tokens are what is valuable to investigators, since they allow API access without the need for passwords. The study found several places where tokens are stored on Android phones: plain text files in app folders, XML preference files, and encrypted databases. The difficulty of tokens' extraction depends on the car brand. BMW's app stored tokens as plain text, making them easy to find, while other brands used encryption to protect them. When tokens were too hard to extract, researchers used the account username and password to create new tokens through the authentication API instead.

3.4 Proof of Concept Tool Development

Based on API analysis, researchers created a Python tool called `gta.py` (Grand Theft API) to demonstrate API-based acquisition for data acquisition from vehicle cloud data from different car manufacturers. The tool handles authentication for various brands and offers a simple text user interface for making API requests. To keep the process forensically sound, it records all API interactions and saves the collected data with timestamps and integrity hashes. This ensures the evidence remains valid for court cases while helping investigators collect and analyze vehicle data properly.

4 Vehicle Cloud Data Categories

Based on the analysis of vehicle assistant applications and research by Ebbers et al. [2], we identified seven categories of data, accessible through vehicle APIs. Each category provides unique forensic value for investigators. An overview of which manufacturers provide which data categories is summarized in Table 1.

4.1 User Data

User data includes personal information related to accounts: email addresses, phone numbers, names, and sometimes addresses or birth dates. This information links people to vehicles and helps identify who has access to a vehicle.

Table 1: Data Categories Available Through Manufacturer APIs.

Manufacturer	User Data	Status Data	Health Data	Remote Control	Trip Data	Charging Data
BMW	✓	✓	✓	✓	✓	✓
Dacia	✓	✓	✓	✓		
Ford	✓	✓	✓	✓	✓	✓
Hyundai	✓	✓	✓	✓	✓	
Mercedes	✓	✓	✓	✓	✓	✓
Tesla	✓	✓	✓	✓	✓	✓

All tested APIs provided basic authentication data. Mercedes' system revealed detailed personal information including home addresses and birth dates, while BMW showed specific user permissions for remote vehicle control. In our implementation, we successfully accessed a BMW user profile containing sensitive information like full name, email address, and account identifiers.

4.2 Descriptive Data

Descriptive data includes vehicle identifiers such as color, model, VIN, and connectivity features. This information helps identify vehicles when only mobile devices are available for analysis.

In the findings, BMW's API exposed the ICCID (SIM card identifier) and head unit identifier—technical information that could be linked to telecommunications records. Dacia and Ford provided URLs to stock images, while Ford and Tesla showed user-assigned vehicle nicknames. These details are valuable when investigators have access to suspects' devices, but not to their vehicles.

4.3 Current Status

Current status information provides real-time vehicle data including mileage, location, fuel/charging state, and door status. This category allows vehicle tracking and timestamp verification for establishing timelines.

Testing revealed that all APIs provided mileage and range information, with BMW, Dacia, Hyundai, and Tesla showing geographic coordinates suitable for tracking. BMW offered detailed status of windows, doors, and charging connections. All status data included timestamps not visible in app interfaces, providing precise timing information for investigations.

4.4 Health Data

Health records include maintenance information like tire pressure, oil status, and service history. This data shows vehicle maintenance patterns and can link vehicles to specific service locations.

During testing, Tesla provided tire pressure readings relevant to accident investigations, while BMW offered more comprehensive monitoring messages for tires and oil levels. BMW and Mercedes listed upcoming service appointments, with Mercedes including specific workshop information.

4.5 Remote Control Features

Remote capabilities encompass vehicle interaction functions: locking/unlocking doors, climate control, lights, and engine starting

features. These functions represent potential investigative tools when properly authorized.

Tesla's remote control features were tested, including opening charging ports and locking/unlocking doors. This shows that manufacturer APIs allow not only reading vehicle data but also controlling vehicle systems remotely.

4.6 Trip Data

Journey information includes start/end points, routes, speeds, and distances traveled. This data proves useful for verifying alibis or establishing movement patterns during investigations.

Contrary to expectations, only Hyundai provided trip information via API in the testing, including journey times, distances, and speeds for the previous three months. Detailed route information with waypoints was not available. Mercedes vehicles use a separate app for trip logging, making trip data inaccessible through their main API.

4.7 Charging and Refueling Data

Charging and fueling records document energy consumption events with times, locations, and amounts. These records help identify user habits and place vehicles at specific locations.

The BMW X1 hybrid provided available detailed charging data, including timestamps, durations, and location addresses. This information could be accessed for the entire lifespan of the vehicle, allowing the creation of a complete historical record of charging events and locations - which is crucial for confirming the vehicle's presence at specific times.

5 Practical Implementation

To demonstrate the practical feasibility of API-based vehicle cloud forensics, we tested an implementation focusing on Mercedes-Benz and BMW vehicles.

5.1 Creating accounts

To test the implementation, we acquired temporary access to accounts for two vehicles: a Mercedes-Benz B220 and a BMW. These were entrusted to us by family members for research purposes only, and we did not have physical access to any of the vehicles.

5.1.1 Mercedes-Benz vehicle. The Mercedes-Benz ID account must first be created on the website. To confirm ownership of a used vehicle, a user must physically present the relevant documentation at a licensed Mercedes-Benz dealership. While this is a tedious task, the vehicle can still be driven without an account, however,

existing accounts cannot be removed from the vehicle without the dealership. Publicly available documentation also describes the process of pairing via the MBUX infotainment screen using a one-time QR-code; however, we were unable to test this.

5.1.2 BMW vehicle. To successfully test the BMW workflow, we first created a dedicated *BMW ConnectedDrive* account through the *MyBMW* mobile application. After registering, we added the vehicle's VIN to the account, which established a logical link between the backend services and the target car. The application then requested in-vehicle confirmation: on the iDrive infotainment screen, we accepted the pairing prompt, completing BMW's two-factor association process.

5.2 Acquiring Mercedes-Benz Cloud Data

The `gta.py` utility, which was released as part of [2], provides a simple to use interface for extracting data from manufacturer-provided APIs. The tool uses a modular approach to facilitate future improvements. At the time of writing, the Mercedes-Benz module was, however, outdated. The domain `bff-prod.risingstars.daimler.com`, used for the login and authentication API, was not resolvable, which produced an error when attempting to login using an email and password.

We attempted to update the tool by extracting the new API address by inspecting the DNS traffic between the mobile device and the Internet. This was achieved by configuring a Linux computer as a wireless access point and router and inspecting the packets using Wireshark. This effort proved worthwhile, as we were able to extract several relevant domain name queries, listed in Table 2.

This approach, however, yielded no useful results, so we resorted to inspecting the application code. We used *ADB* to extract the `.apk` file from the device and *jadx* to decompile the source code. When inspecting the decompiled code, we discovered the method for selecting a domain name. The class

```
mercedesbenz.data.common.configuraiton.Endpoint
constructs it using the base address, the user's region and the stage
(prod, non-prod, local):
```

Listing 1: The code for dynamic building of endpoint address.

```
public String getBffUrl() {
    if (this.stage == Stage.LOCAL) {
        return LOCAL_HOST;
    }
    return BFF_HOST
        + this.region.getRegionSuffix()
        + this.stage.getStageSuffix()
        + ".mobilesdk.mercedes-benz.com";
}
```

This can help us identify the relevant API servers, however, due to encryption, we are unable to ascertain exactly which endpoint address on that server the request is directed at. To find out more, we decided to recreate Ebbers et al.'s [2] method of using a man-in-the-middle proxy [7] to inspect the encrypted traffic, as shown in Figure 1.

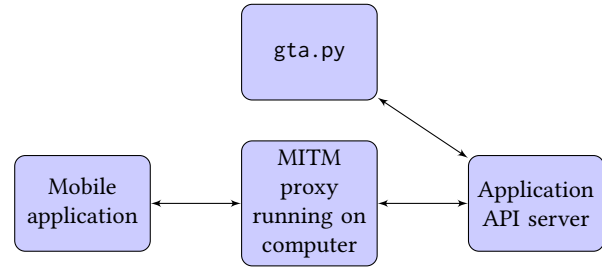


Figure 1: Operation of the MITM proxy.

This did not yield any useful results, as the app's communication is secured using certificate pinning, and does not use the operating system's own certificate store. To bypass that, we injected the proxy SSL certificate into the app using `apk-mitm`. This caused the application to exit after sending several requests to the aforementioned address. Due to the lack of access log files, we were unable to locate the mechanism behind that. However, since the app was able to successfully communicate with the servers through our proxy, we believe this may be an anti-tampering measure.

Furthermore, the server did respond to the request

```
GET https://bff.emea-prod.mobilesdk.mercedes-benz.com/v1/config
```

by returning, among other data, a JSON document that included the following snippet:

Listing 2: The intercepted JSON response

```
{
  "forceUpdate": {
    "status": "OK",
    "storeUrl":
      "https://play.google.com/store/apps/details?id=com.daimler.ris.mercedesme.ece.android",
    "updateTime": "0001-01-01T00:00:00.000Z"
  },
}
```

This suggests the presence of a mechanism to verify the integrity of the installed application package. Furthermore, the application did not crash once internet connectivity was disabled on the device, further confirming this hypothesis.

5.2.1 Risks. Having inspected the contents of the `.apk` file we extracted from the Android phone, we discovered several expired `.pem` and `.crt` files with expiration dates as far back as 2019. These should, in line with best practices of public key cryptography, be denied in all cases. Keeping them in a publicly available application is therefore a security oversight, and should be removed.

When exploring the extracted `.apk` archive, we confirmed the use of certificate pinning for several domains. This is however done by a wildcard such as `*.risingstars.daimler.com`. We believe this is a potential risk, as the configuration lists several valid signatures for a potentially infinitely large set of valid domain names. Since DNS spoofing is a known attack vector, this reduces the effort required to find a signature collision. While there is a low

Table 2: Domains identified in the DNS-traffic interception method.

Domain name	Triggering events	Notes
bff.emea-prod.mobiledsk.mercedes-benz.com	—	uses the pinned certificate
id.mercedes-benz.com	login, logout	—
me-connect-tou-emea.mercedes-benz.com	login, listing user data	—
websocket.emea-prod.mobiledsk.mercedes-benz.com	opening app	—
content.emea-prod.mobiledsk.mercedes-benz.com	opening app, switching tabs	—
api.oneweb.mercedes-benz.com	—	—
docms.i.mercedes-benz.com	login	does not use a public SSL certificate on port 443

likelihood of such an attack, we would suggest specifying exact domain names with a single signature and pushing an app update on any changes when necessary.

5.3 Acquiring BMW Cloud Data

To access the *MyBMW* backend APIs, we need the *access* and *refresh* tokens that the mobile app receives after a successful login.

5.3.1 API prerequisite. BMW's back-end accepts only OAuth 2.0 bearer tokens. Consequently, a single tuple: (access_token, refresh_token, vin, expiry_epoch) is sufficient for every forensic query.

5.3.2 Credential-acquisition strategy. Because the *MyBMW* iOS client completes its PKCE flow over an *unpinned* TLS channel, we again intercepted the exchange with mitmproxy[7]:

a) *Proxy deployment* – start mitmproxy -p 8080 on macOS; set the iPhone's Wi-Fi proxy to the Mac; install and trust the proxy CA. b) *Login replay* – sign out/in once in the *MyBMW* app. The resulting /oauth/token response (Listing 3) contains the required credentials.

Listing 3: JSON payload with BMW

```
{
  "access_token": "iSAao...Iku",
  "refresh_token": "lBuoB...Dko",
  "expires_in": 3599
}
```

The lifetime value expires_in (SI3599s) is converted to an epoch (1747481197) and, together with the vin, stored in config/BMW_tokens.csv:

```
vin,access_token,refresh_token,expiry_epoch
WBA1A2B3C4D...,iSAao...Iku,lBuoB...Dko,1747481197
```

After we had prepared the BMW_token module and written the token tuple to config/BMW_tokens.csv, we ran the gta.py and tried to connect to the BMW Cloud. However, we did not successfully connect to the cloud. So the next step was adjusting the author's code.

5.3.3 Regional endpoint versioning (EU). As we observed, BMW operates multiple API versions in parallel. In the EU back-end the canonical paths were migrated from /eadrax-vcs/v1/ to /eadrax-vcs/v5/; attempts to query v1 return 404. We tried to patch the tool with:

```
sed -i 's/eadrax-vcs/v1/eadrax-vcs/v5/g' src/*.py
```

After this region-specific adjustment, all requests were accepted with 200.

The same refresh_token remains valid for approximately 90 days, enabling periodic collections without repeating the interception procedure.

5.3.4 Retrieving the user profile. With the access_token in hand, we can call /my/user/v5/identity (or any profile endpoint) against the BMW backend. The response below proves that we have reached the customer record stored in the cloud and extracted key attributes such as gcid, surname, homeMarket, and the registration e-mail.

Listing 4: JSON response from BMW /identity endpoint showing user-profile fields

```
{
  "businessPartner": {
    "gcid": "0137b7...",
    "ucid": "T0...",
    "surname": "Br...",
    "givenName": "An...",
    "salutation": "SAL_MS",
    "correspondLanguageISO": "EN",
    "homeMarket": "SI"
  },
  "userAccount": {
    "mail": "ana.br...",
    "status": "ACTIVATED",
    "authenticationLevel": "ONE_FACTOR"
  }
}
```

5.3.5 Attempted Vehicle Data Retrieval and Troubleshooting. The next step in our workflow was to retrieve live vehicle telemetry via the gta.py interface. Upon selecting the "2 -vehicles" option, the tool consistently failed to reach the endpoint:

```
Choose Request: 2
('Status Code: 404',)
Bmw: API error - most likely invalid token or
insufficient vehicle capabilities
```

Because we were able to reach the user data, we confirmed that the access token was valid. We tried to diagnose the problem and performed the following probes:

- (1) **API version sweep** Tested endpoints /v1, /v2, /v3, /v4, /v5, and /emea/v5 on both cocoapi.bmwgroup.com and

b2vapi.bmwgroup.com; all returned HTTP 404 or no response.

- (2) **Endpoint cluster probe** Queried /vehicle_list on:
 - https://cocoapi.bmwgroup.com/eadrax-vcs/v5/vehicle_list
 - https://cocoapi.bmwgroup.com/eadrax-vcs/emea/v5/vehicle_list
 - https://b2vapi.bmwgroup.com/eadrax-vcs/v5/vehicle_list
 All returned HTTP 404.
- (3) **VIN verification** A manual /vehicle_list query returned no VINs, indicating the back-end did not recognise any vehicle for the account on the tested clusters.
- (4) **Privacy and provisioning checks** Confirmed that *Remote Services* was active in the MyBMW app and that data sharing was set to “All data & position” in iDrive.

Despite a valid OAuth token (successful retrieval of users profile) and correct VIN, all vehicle-specific URL on public and legacy clusters returned HTTP 404.

5.3.6 Conclusion. These symptoms strongly indicate that BMW has migrated or deprecated the REST vehicle endpoints for Europe, which were valid at the time of writing of Ebbers et.al.[2] work. The vehicle data API no longer responds on any known path (/v1-/v5, /emea/v5, legacy gcdm), preventing further automated retrieval. This aligns with user reports in the Home Assistant core issue #116145 [1], where community-maintained integrations broke for the same reason and required patches to point at the new EU cluster paths.

6 Discussion

In this study we performed live cloud-forensic tests on two major OEMs: Mercedes-Benz and BMW. Firstly, we focused on Mercedes. We have successfully paired the vehicle and intercepted DNS/API traffic, but when we tried to use the GTA's gta.py, it didn't work. The module was outdated and unable to reach any vehicle endpoints without extensive DNS and pinning work. For BMW, we acquired valid OAuth tokens and were able to retrieve user-profile data. However, when we tried to access the vehicle data, we were not successful. These failures underscore the fragility of unofficial forensic tools due to the rapid evolution of manufacturer cloud APIs.

6.1 Mercedes-Benz Findings

- **Certificate pinning and domain retirement.** The original bff-prod.risingstars.daimler.com host was unreachable, and certificate pinning, along with other safety measures, prevented simple man-in-the middle traffic inspection. We extracted new hostnames via DNS inspection and bypassed pinning by patching the app files, but still could not locate stable vehicle-data paths.
- **Outdated client module.** The Mercedes module in gta.py targeted outdated endpoints and failed to authenticate, demonstrating that without continuous maintenance, forensic tools quickly lose functionality.

6.2 BMW Findings

- **User-profile still accessible.** Identity endpoints under /my/user/v5/identity returned HTTP 200, confirming that access tokens are valid.
- **Vehicle-data migration.** Status, trips, charging, and commands endpoints on /eadrax-vcs/v1-v5, /emea/v5, and gcdm all returned HTTP 404, indicating BMW has relocated or deprecated these APIs without public notice.

6.3 Challenges Consistency

Despite its robust architectural foundation, the gta.py tool currently confronts a critical “chicken-and-egg” dilemma. The tool's limited adoption has led to insufficient maintenance of its modules, rendering them outdated and effectively unusable for the tested vehicle brands. This lack of practical utility, in turn, discourages broader adoption, creating a self-reinforcing feedback loop that inhibits community engagement and impedes further development.

Two potential solutions to this challenge can be identified. First, increased adoption within the digital forensics and research communities could foster collaborative maintenance efforts. In such a scenario, investigators, security researchers, and commercial partners would contribute updated API modules, feature enhancements, and expanded platform support. The tool's modular architecture makes it particularly well-suited for this type of distributed, open-source development model.

Secondly, a more comprehensive solution would involve establishing legal or regulatory frameworks that mandate manufacturers to provide standardized API interfaces for forensic and lawful data access. While incredibly ambitious in scope, this approach would eliminate the current dependence on undocumented endpoints and reverse engineering techniques, offering investigators predictable, auditable, and stable access to vehicle telemetry data across different manufacturers and jurisdictions.

Until one of these pathways materializes, API-based vehicle cloud forensics remains a technically promising yet inherently fragile methodology—one that demonstrates clear feasibility but remains critically dependent on dynamic, opaque, and rapidly evolving backend infrastructures.

6.4 Moral Implications and Privacy

As we have seen, the method proposed proves itself to be quite an efficient way to access personal data of an individual, such as their name, email address, and vehicle information. While forensically valuable, this raises important privacy and ethical questions. Modern vehicles collect personal data such as location history and contact lists, and accessing this data, even with legal authorization, can bypass user consent and challenge regulatory norms.

From a legal standpoint, using reverse-engineered APIs may conflict with data protection laws like the GDPR, especially if the access methods are not endorsed by manufacturers. Even when used under lawful conditions, such as with a warrant, the opaque nature of these methods could undermine public trust and accountability.

Ethically, the potential for misuse, we think, underscores the need for clear oversight. In addition, there is the risk of normalizing such invasive practices in the name of forensic utility. Without proper boundaries and tools, such as the one we presented, could be

repurposed by third parties in ways that cannot be simply controlled by law enforcement.

We recommend the establishment of clear legal and technical guidelines to govern the responsible use of vehicle cloud data access methods. These rules should make sure data is accessed openly and that forensic methods respect both the needs of investigations and the rights of individuals. Collaboration between technologists, legal experts, and ethicists will ensure that forensic advancements respect these privacy rights.

A responsible approach keeps technological capabilities at pace with ethical safeguards, so that access to sensitive vehicle data remains justified, fair, and accountable.

7 Conclusion

Our analysis shows that vehicles generate significant telemetry data, including user identities, trip histories, and vehicle status, all accessible via manufacturer APIs. To evaluate how API-based data collection works in practice, we tested two well-known car brands: Mercedes-Benz and BMW. With Mercedes-Benz, the forensic tool `gta.py` proved ineffective because many of the addresses were outdated and the security measures were exceptionally robust. Even after adjusting the settings, we were unable to access the required data. In the case of BMW, the APIs were somewhat more accessible, but challenges remained due to frequent changes in endpoint versions and unclear documentation. In both instances, the rapid evolution of the manufacturers' systems outpaced open-source forensic tools, making it difficult to obtain the necessary data.

Findings underscore three key challenges for future work. First, manufacturers frequently change their API infrastructures without notice, leaving community-developed tools outdated almost as soon as they are released. Second, security controls differ widely; what works for one brand may be blocked entirely by another. Third, our study was necessarily limited in scope—other brands may present even greater obstacles. Moving forward, the future of vehicle cloud forensics will depend on sustained research, active collaboration, and possibly regulatory action to create stable, standardized access pathways. Only through such collective and adaptive efforts can the full forensic value of vehicle cloud data be reliably unlocked for investigators, while also ensuring proper safeguards for privacy and legal compliance.

References

- [1] home-assistant/core contributors. *Issue #116145: BMW integration broken after API migration*. 2024. URL: <https://github.com/home-assistant/core/issues/116145> (visited on 05/14/2025).
- [2] Simon Ebberts et al. "Grand theft API: A forensic analysis of vehicle cloud data". In: *Forensic Science International: Digital Investigation* 48 (2024). DFRWS EU 2024 - Selected Papers from the 11th Annual Digital Forensics Research Conference Europe, p. 301691. ISSN: 2666-2817. DOI: <https://doi.org/10.1016/j.fsidi.2023.301691>. URL: <https://www.sciencedirect.com/science/article/pii/S266628172300210X>.
- [3] Kevin Klaus Gomez Buquerin, Christopher Corbett, and Hans-Joachim Hof. "A generalized approach to automotive forensics". In: *Forensic Science International: Digital Investigation* 36 (2021). DFRWS 2021 EU - Selected Papers and Extended Abstracts of the Eighth Annual DFRWS Europe Conference, p. 301111. ISSN: 2666-2817. DOI: <https://doi.org/10.1016/j.fsidi.2021.301111>. URL: <https://www.sciencedirect.com/science/article/pii/S2666281721000056>.
- [4] Md Mahmud Hossain, Ragib Hasan, Shams Zawoad, et al. "Trust-IoV: A Trustworthy Forensic Investigation Framework for the Internet of Vehicles (IoV)". In: *ICIOT*. 2017, pp. 25–32.
- [5] Daniel Jacobs et al. "Volkswagen car entertainment system forensics". In: *2017 IEEE Trustcom/BigDataSE/ICSS*. IEEE. 2017, pp. 699–705.
- [6] Todd Litman. "Autonomous vehicle implementation predictions". In: (2017).
- [7] mitmproxy is a free and open source interactive HTTPS proxy. URL: <https://mitmproxy.org/>.
- [8] Dennis K Nilsson and Ulf E Larson. "Conducting forensic investigations of cyber attacks on automobile in-vehicle networks". In: *International Journal of Digital Crime and Forensics (IJDCF)* 1.2 (2009), pp. 28–41.
- [9] Vassil Roussev and Shane McCulley. "Forensic analysis of cloud-native artifacts". In: *Digital Investigation* 16 (Mar. 2016), S104–S113. DOI: [10.1016/j.diin.2016.01.013](https://doi.org/10.1016/j.diin.2016.01.013).
- [10] Dario Stabili, Filip Valgimigli, and Mirco Marchetti. "I know where you have been last summer: Extracting privacy-sensitive information via forensic analysis of the Mercedes-Benz NTG5*2 infotainment system". In: *Forensic Science International: Digital Investigation* 53 (2025), p. 301909. ISSN: 2666-2817. DOI: <https://doi.org/10.1016/j.fsidi.2025.301909>. URL: <https://www.sciencedirect.com/science/article/pii/S2666281725000484>.

5 Datoteke in sistemi / Files and systems

Applying digital stratigraphy to the problem of recycled storage media

Nejc Jager
nj65755@student.uni-lj.si
FRI
Ljubljana, Slovenia

Matija Krigl
mk39539@student.uni-lj.si
FRI
Ljubljana, Slovenia

Urša Kumelj
uk29151@student.uni-lj.si
FRI
Ljubljana, Slovenia

Abstract

This report provides an overview of a study by Schneider et al. (2024) that applied digital stratigraphy to recycled storage media in a forensic context. The main goal of that research was to determine whether this technique can distinguish older residual data from new data on a reused device, especially when such fragments are found in unallocated space without metadata. The methodology relies on the concept of digital stratigraphy – adapted from geology – to analyze how file fragments are layered on disk over time. Schneider et al. implemented an automated file system simulation framework that systematically creates, modifies, and deletes files using real file system drivers, generating realistic layers of data for analysis. Experiments showed that examining these data layers and observing file system behavior can reveal stratigraphic markers indicating a fragment’s likely origin. These markers help investigators infer whether a recovered file fragment originated from the device’s previous usage or the current user’s activities. While the digital stratigraphy approach cannot definitively attribute every fragment, it significantly improves data provenance assessment on recycled storage media when conventional file metadata is absent.

Keywords

Digital stratigraphy, File System Upper Bound (FSUB), Data layering, Recycled Storage Media, Forensic analysis

1 Introduction

The proliferation of recycled storage media has created significant challenges in digital forensics, particularly when determining data ownership. Investigators often find residual data from previous users persisting in non-allocated space, making it difficult to establish whether recovered files belong to the current owner or are artifacts from prior usage. This problem extends to both second-hand devices and new equipment containing recycled components.

Existing forensic methods struggle to reliably attribute data in such cases, as conventional analysis cannot always distinguish between intentionally deleted content and leftover fragments. This research addresses this gap by applying principles of stratigraphy—traditionally used in geology and archaeology—to analyze data layers in storage media. By examining how files are written, modified, and overwritten, we develop a systematic approach to determine data provenance in recycled storage devices.

This work provides investigators with new tools to assess whether recovered evidence originated from a suspect’s usage or was inherited from previous ownership. The methodology is particularly valuable for cases involving USB drives, SSDs, and other removable media where data attribution is critical for legal and investigative purposes.

2 Digital stratigraphy

Digital stratigraphy applies geological layering principles to analyze how data is stored and modified in file systems. The core of this approach examines the physical arrangement of files on storage media to establish their relative creation order. A key concept is the File System Upper Bound (FSUB), defined as the highest storage block containing data written by the current file system. The FSUB serves as a critical reference point:

- Data found below the FSUB likely belongs to the current file system.
- Data above the FSUB may represent residual information from prior usage.
- The boundary helps evaluate whether recovered files were part of active usage.

Our methodology analyzes three fundamental behaviors:

- (1) How file systems allocate space for new data.
- (2) How they reuse space from deleted files.
- (3) How residual data persists after reformatting.

This method works well for used devices, but it’s not perfect. If the device is very full or files are scattered everywhere, it gets harder to tell when files were created. Still, when we use this along with other detective tools, it helps us piece together the story of what happened on a storage device.

3 Automation Framework Development

The ‘File System Activity Simulator’ (FSAS), a Python script (`fsas.py`), was developed to create realistic file system states and programmatically execute operations via OS interfaces, preserving state and enabling visualization. It comprises three commands:

3.1 Generating File System Operations

The `generate` command creates a static yet randomly generated flowchart of file system operations, stored as an XML file for repeatable execution on Windows and Linux. It supports file (`create`, `shrink`, `enlarge`, `delete`) and directory (`create`, `delete`) operations. Parameters are randomly selected under defined conditions, with consistency checks to ensure logical sequences. The resulting XML file details the operations to be performed.

3.2 Executing File System Operations

The `execute` command runs a previously generated XML file on a file system. It allows selection of output types to preserve the file system state before and after each operation, potentially using external tools. Specialized outputs (`alloc`, `nonzero`, `pattern`) leveraging The Sleuth Kit (TSK) Python bindings (`pytsk`) are available for supported file systems. The script executes commands from

the XML, applying selected tools to the initial state and logging subsequent states after each operation.

3.3 Generating Visualizations

The plot command generates visual representations of the experimental data, offering casey, scatter (using Matplotlib), and heatmap (using Plotly) plot types. Auxiliary files are created based on the chosen plot, potentially large depending on the file system size.

4 Experimental Results

This section presents the experimental findings addressing the research questions, focusing on file system driver behavior, file system specifics across operating systems, the File System Upper Bound (FSUB), and the persistence of old data.

4.1 File System Driver Behaviour

Experiments investigated sequential write patterns and block reuse upon deletion for FAT32, exFAT, and NTFS on Windows and Linux. Casey plots and newly developed scatter plots visualizing block status changes were used. Sequential writes were confirmed for FAT32 and exFAT on both platforms, while NTFS showed non-contiguous sequential allocation. Upon deletion, FAT32 on Linux showed delayed block reuse, filling gaps only after reaching the end of the file system, whereas Windows exhibited immediate reuse. exFAT showed immediate gap filling with no OS-specific differences. NTFS also filled gaps, but precise behavior was harder to determine from plots.

4.2 File System Specifics

Analysis revealed OS-dependent file system specifics, further explored using dynamic heatmaps visualizing actual block status and combinations. FAT32 on Linux had larger and differently located FATs compared to Windows, likely due to differing default cluster sizes. NTFS on Linux showed an initial reserved area for new data and a second reserved area containing \$MFTMirr, while Windows had these structures swapped. Sequential allocation in NTFS was less frequent on Linux.

4.3 File System Upper Bound (FSUB)

The concept of a calculable FSUB, representing the highest block with current file system data, was investigated. A helpful tool, called (`get_fsub_info.py`) was developed to calculate FSUB based on live and deleted files using `pytsk`. Experiments comparing the highest written block with the calculated FSUB showed correspondence for live files. However, scenarios with significant file deletion demonstrated that the calculated FSUB can be a conservative lower estimate.

4.4 Investigating Previous File System Data Prior to FSUB

Experiments explored the presence of old data (written before formatting) below the FSUB. For FAT32, old data was found above the FSUB until the volume end on both OSes, but within the data area below the FSUB, it was limited to cluster slack on Windows;

Linux zeroed cluster slack. NTFS on Windows retained old data in file system files and potentially significant amounts in the \$MFT reserved area. On Linux, old data was at the volume start and in the \$MFT reserved area, with potential for large areas due to \$MFTMirr placement. exFAT on Windows showed old data in reserved areas, system file slack, and the FAT; below the FSUB, it was limited to cluster slack, similar to FAT.

5 Citations

5.1 Recovery of Deleted Files in the NTFS File System using Python and PyTSK3

The researchers referenced this study [6] because it helped solve a key problem in their file recovery work - telling whether deleted files actually belonged to the current user or were just leftover data from previous owners. This was especially important for their NTFS recovery tool since:

- It showed how to analyze where and how files are stored to determine their origin.
- It provided methods to identify when data was likely from past usage rather than current activity.
- It helped them improve their tool's ability to recover the right files in cases where storage devices had been reused.

The stratigraphy approach gave them a scientific way to handle these tricky situations, making their recovery tool more accurate for forensic investigations involving used drives and devices.

5.2 SOLVE-IT: A proposed digital forensic knowledge base inspired by MITRE ATT&CK

The SOLVE-IT paper [3] used this study to solve a common problem in digital forensics: telling whether recovered files actually belong to the current user or are just leftovers from previous device owners. When forensic tools recover deleted files (called "file carving"), they sometimes get this wrong - especially on used devices like second-hand hard drives or USB sticks.

The SOLVE-IT team added this as a solution in their guidebook for investigators (they call it "mitigation M1061"). Now when forensic experts recover files from used devices, they can use Schneider's method to check if they're looking at current evidence or just old data that shouldn't be used in their case.

This was important because it gave investigators a scientific way to avoid mistakes when working with recycled storage devices, making their findings more reliable in court. The reference shows how SOLVE-IT connects useful research to real investigative work.

6 Comparison with Prior Work

6.1 Challenges of Residual Data

The challenge of residual data on recycled storage media has been highlighted in recent forensic research. [4] demonstrated through a large-scale study that even brand-new USB flash drives can contain leftover user data, due to the reuse of memory chips. In their analysis of 614 "new" drives, over 12% were found to house non-trivial data from previous usage. This poor sanitization practice creates a serious attribution problem: a suspect might plausibly claim that

incriminating content discovered in the device’s unallocated space was already present when the device was purchased.

6.2 Digital Stratigraphy as a Solution

The main paper by Schneider et al. (2024) [5] directly addresses this problem by exploring whether *digital stratigraphy* can help discern the true provenance of such data remnants. Digital stratigraphy, originally adapted to forensic science by [2], involves analyzing how a file system writes and overlays data layers over time. By analyzing the physical placement of file fragments on disk, investigators can infer their relative ages (newer vs. older). This process depends heavily on understanding file system behavior—how data is allocated, modified, and deleted under systems like FAT or NTFS. It also requires awareness of potential pitfalls that can mislead interpretation, such as file initialization artifacts, tunneling effects, or irregular allocation patterns. Paper [2] is crucial as an initial theoretical groundwork.

Building on [2] foundational concept, [5] applied digital stratigraphy specifically to the scenario of recycled media. The methodology in their paper differs from earlier studies because they created an automatic simulation of file system activity. The aim is to find markers that show if leftover files belong to the current or previous owner. In essence, [5] seek to link "content-only" artifacts (files recovered without metadata) to the correct usage layer of the device. This approach directly tackles the attribution question raised by [4]. Can incriminating content in a device’s free space be confidently tied to the current file system or did it likely originate from earlier use of the recycled component?

6.3 Context and Complementary Techniques

Prior studies offer important context and techniques that complement this investigation. [2] work first showed that file allocation can reveal the sequence of past events on a disk, forming the basis for digital stratigraphy. Other researchers have since expanded these concepts. For example, [1] developed a method to estimate the *time window* when a deleted file fragment was originally written by examining the timestamps of its neighboring allocated blocks. In their approach, the known creation times of the k -nearest neighboring fragments provide lower and upper bounds for the orphan fragment’s creation date. This approach is similar to archaeological dating methods that use surrounding evidence to determine age. However, [1] solution is focused on establishing temporal bounds for a fragment’s existence, rather than distinguishing ownership. In contrast, Schneider et al. (2024) concentrate on the *source* of the fragment by leveraging how data layers from different usage periods might be discerned on a recycled device.

Crucially, none of the prior works explicitly tackled the unique forensic challenge posed by recycled storage media before. [2] and related studies explored timeline reconstruction and relative file dating in general scenarios, and methods like [1] targeted fragment dating in isolation. These studies showed ways to estimate when files were created but did not deal with data ownership on reused devices. Schneider et al. (2024) fill this gap by applying the established principles of digital stratigraphy to a new context. By combining insights from earlier research with their simulation-driven experiments, they demonstrate a way to approach the provenance

question for content found in non-allocated space on recycled media.

7 Conclusion

This study demonstrates that digital stratigraphy and FSUB analysis provide valuable forensic tools for investigating recycled storage media, offering systematic methods to assess data provenance when traditional metadata is unavailable. While the technique cannot definitively attribute all recovered data due to limitations with heavily fragmented devices, it significantly improves investigators’ ability to distinguish between current and residual data. Future research should extend this work to Apple’s filesystems (APFS, HFS+), as their unique allocation behaviors may require different stratigraphic approaches compared to NTFS and FAT32 systems. The developed framework and automated tools advance forensic capabilities for analyzing file system behaviors across Windows and Linux environments, ultimately strengthening evidentiary assessments involving reused storage devices.

References

- [1] Abdul Bahjat and Andrew Jones. 2019. Deleted file fragment dating by analysis of allocated neighbors. *Digital Investigation* 31 (2019), 1–10.
- [2] Eoghan Casey. 2017. Digital Stratigraphy: Contextual Analysis of File System Traces in Forensic Science. *Journal of Forensic Sciences* 63, 4 (2017), 1063–1071.
- [3] Christopher Hargreaves, Harm van Beek, and Eoghan Casey. 2025. SOLVE-IT: A Proposed Digital Forensic Knowledge Base Inspired by MITRE ATT&CK. *Forensic Science International: Digital Investigation* 52 (March 2025), 301864. doi:10.1016/j.fsidi.2025.301864
- [4] Johannes Schneider, Andreas Dewald, and Felix C Freiling. 2021. In Search of Lost Data: A Study of Flash Sanitization Practices. *Forensic Science International: Digital Investigation* 36 (2021), 301118.
- [5] Janine Schneider, Maximilian Eichhorn, Lisa Marie, and Christopher Hargreaves. 2024. Applying digital stratigraphy to the problem of recycled storage media. *Forensic Science International* (2024).
- [6] K. Venkatesh, Tharshan RS, N. Karthik, P.Shahith Nazeem, and B. Maharaja. 2024. Recovery of Deleted Files in the NTFS File System Using Python and PyTSK3. In *2024 International Conference on IoT Based Control Networks and Intelligent Systems (ICICNIS)*. 165–171. doi:10.1109/ICICNIS64247.2024.10823366

Forenzični vidiki zloženih datotečnih sistemov

Lara Anžur

Fakulteta za računalništvo in informatiko, Univerza v
Ljubljani
Ljubljana, Slovenija
la4710@student.uni-lj.si

Žiga Leskovec

Fakulteta za računalništvo in informatiko, Univerza v
Ljubljani
Ljubljana, Slovenija
zl5596@student.uni-lj.si

Tina Poštuvan

Fakulteta za računalništvo in informatiko, Univerza v
Ljubljani
Ljubljana, Slovenija
tp1735@student.uni-lj.si

Maksimiljan Vojvoda

Fakulteta za računalništvo in informatiko, Univerza v
Ljubljani
Ljubljana, Slovenija
mv7846@student.uni-lj.si

POVZETEK

Članek obravnava izzive forenzične analize zloženih datotečnih sistemov, ki so postali bolj kompleksni zaradi večslojnih arhitektur in naprednih funkcij, kot je npr. šifriranje. Tradicionalne forenzične tehnike niso primerne za te sisteme, saj zahtevajo analizo tako zgornjih kot spodnjih slojev datotek. Predlagan je nov model, ki vključuje fazo za prepoznavanje indikatorjev in analizo časovnih žigov. Za preverjanje učinkovitosti modela so avtorji izvedli študijo primerov na sistemih MooseFS, GlusterFS in eCryptfs, pri čemer so prepoznali ključne forenzične izzive teh sistemov, kot sta fragmentacija in prisotnost skritih podatkov. Model omogoča izboljšanje forenzične preiskave z obnovo izbranih datotek in celovitejšo analizo datotečnih sistemov.

1 UVOD

Tradicionalna forenzična analiza temelji na predpostavki, da datotečni sistemi zapisujejo podatke neposredno na diskovne nosilce (angl. *volumes*). To omogoča neposredno analizo datotečne strukture in iskanje izbranih ali skritih podatkov [5]. Vendar pa v sodobnih sistemih, zlasti v okoljih, ki uporabljajo oblak in velike podatkovne baze, podatki pogosto niso več neposredno vezani na nosilec. Zloženi datotečni sistemi (angl. *stacked file systems*) uvajajo dodatno kompleksnost, saj podatke zapisujejo na drug datotečni sistem. Zaradi tega tradicionalne forenzične metode niso več neposredno uporabne, saj se s tem pojavlja zahteva po novi metodologiji in razumevanju večslojnih arhitektur [10].

V nadaljevanju so v poglavju 2 povzete bistvene ugotovitve članka izvirnega članka [10], tega pa smo v poglavju 3 nagradili še z našo analizo datotečnega sistema OverlayFS.

2 PREGLED IZVORNEGA ČLANKA

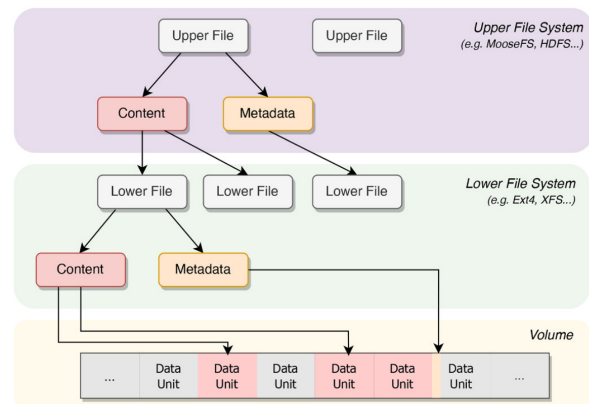
2.1 Zloženi datotečni sistemi

Zloženi datotečni sistemi so posebna vrsta datotečnih sistemov, ki jih sestavljata zgornji (uporabniški) sistem (s pripadajočimi zgornjimi datotekami) in spodnji sistem (s pripadajočimi spodnjimi datotekami), kamor zgornji sistem hrani svoje podatke in metapodatke. Zloženi sistemi, katerih struktura je prikazana na sliki 1, omogočajo:

- **Lokalno uporabo** – V lokalnih okoljih zloženi sistemi omogočajo nadgradnjo funkcionalnosti, kot je šifriranje, brez potrebe po spremembi spodnjega sistema. Tak primer je

eCryptfs, ki šifrira datoteke, ne da bi vplival na osnovni ext4 [7]. Za lokalno okolje gre, kadar se zgornji in spodnji datotečni sistem nahajata na isti napravi.

- **Distribuirano uporabo** – V distribuiranih okoljih, kot so oblačne storitve, omogočajo, da so podatki razdeljeni med več strežnikov, kar povečuje razpoložljivost, a hkrati otežuje forenzično analizo zaradi fragmentacije [2]. Sistemi se še naprej delijo na upravljane in neupravljane.



Slika 1: Pregled zloženega datotečnega sistema, ki uporablja tradicionalni spodnji datotečni sistem za shranjevanje podatkov.

2.2 Pregled dosedanjih raziskav

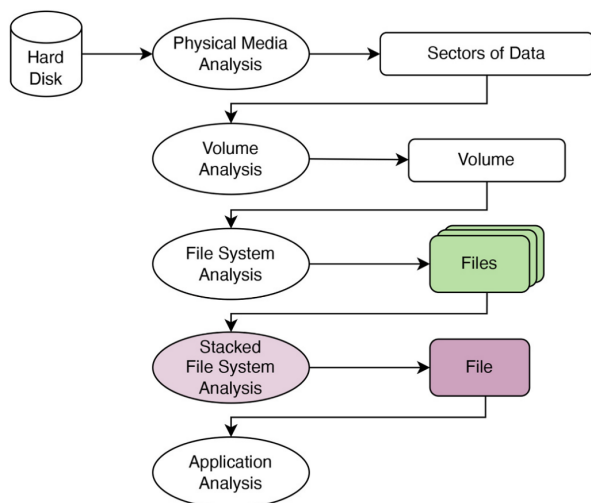
Razdelan koncept zloženih datotečnih sistemov se je prvič pojavil v članku avtorjev Heidemann in Popek [9], in sicer s primarno nalogo, da omogoči lažji razvoj novih funkcionalnosti s pomočjo obstoječih datotečnih sistemov. V nadaljnjih raziskavah se je na podlagi tega razvilo še nekaj preostalih rešitev, ki so datotečnim sistemom omogočale dodatne funkcije, kot so npr. šifriranje, preprečevanje izbrisa datotek, omogočanje varnega brisanja ter sledenje dostopov do datotek [1, 3, 14]. Kljub vsemu pa se omenjene raziskave niso posvetile tudi forenzičnemu vidiku. Le nekaj novejših del, kot sta Asim in drugi [2] ter Harshany in drugi [8], je začelo raziskovati implikacije za forenziko v sistemih, kot je Hadoop Distributed File

System, vendar niso obravnavali širših posledic strukture spodnjega sistema.

2.3 Predlagani model analize

Omejitve se pojavljajo tudi na področju forenzičnih orodij, saj ta lahko analizirajo le spodnje plasti datotečnih sistemov, nimajo pa funkcionalnosti za povezovanje z zgornjimi plastmi. S tem razlogom obstoječi modeli forenzične analize (npr. Carrierjev model [5], prikazan na sliki 2) niso več ustrezni za zložene datotečne sisteme, ker preprosto ne obvladujejo njihove kompleksnosti. Avtorji v članku predlagajo model z dodatno fazo analize (prikazano v vijolični barvi na sliki 2), ki vključuje naslednje glavne razširitve:

- (1) **Identifikacijo indikatorjev** – analizo spodnjega sistema za prepoznavanje značilnih struktur (npr. direktorijev 00–FF v MooseFS).
- (2) **Korelacijo zgornjih in spodnjih datotek** – povezovanje preko inode števil, UUID-jev ali metapodatkov, kjer je to mogoče.
- (3) **Analizo časovnih žigov** – primerjavo žigov v zgornjem in spodnjem sloju za rekonstrukcijo časovnic in odkrivanje manipulacij.



Slika 2: Razširitev Carrierjevega modela za uporabnost v zloženih datotečnih sistemih.

2.4 Eksperimentalni sistem in primeri

Za pridobitev čim bolj celovitih rezultatov so avtorji v raziskavi uporabili tri različne zložene datotečne sisteme, ki vsebujejo različne arhitekture in ponujajo različne izzive:

- **MooseFS** – Distribuiran sistem, kjer glavni strežnik (angl. *Master Server*) vodi evidenco metapodatkov, medtem ko strežniki kosov (angl. *Chunk Servers*) hranijo razdeljene dele datotek.
- **GlusterFS** – Prav tako porazdeljen sistem, ki uporablja peer-to-peer arhitekturo brez glavnega strežnika. Vsak izmed

strežnikov hrani dele podatkov in ti se povezujejo drug na drugega, s čimer ustvarijo zaupanja vreden sklop (angl. *trusted pool*).

- **eCryptfs** – Lokalni sistem, ki uporabnikom omogoča šifriranje posameznih datotek brez vpliva na spodnji ext4 sistem [7].

Omenjeni datotečni sistemi so podrobneje primerjani v tabeli 1. Kot spodnji datotečni sistem je bil zaradi svoje razširjenosti in primerljivosti rezultatov v vsakem od naštetih zloženih sistemov uporabljen ext4.

Tabela 1: Primerjava testiranih zloženih datotečnih sistemov

Lastnost	MooseFS	GlusterFS	eCryptfs
Vrsta	Upravljan	Neupravljan	Lokalni
Distribucija	Da	Da	Ne
Glavni strežnik	Da	Ne	Ne
Podpora šifriranju	Ne	Možno	Da
Glavna prednost	Redundanca	Prilagodljivost	Šifriranje

2.5 Identifikacija sistema na spodnjem sloju

Za uspešno identifikacijo zloženega datotečnega sistema je ključno, da se najprej opravi analiza spodnjega datotečnega sistema, pri čemer je treba prepoznati morebitne kazalce. Ti se med posameznimi sistemi razlikujejo:

- **MooseFS** – Prepoznavni elementi vključujejo značilno strukturo direktorijev od 00 do FF ter 0x2000 bajtne glave kosovnih (angl. *chunk*) datotek, ki vsebujejo podpis (MFSC 1.0 ali 1.1), identifikator kosa in njegovo verzijo [10].
- **GlusterFS** – Ob vzpostavitvi in zagonu nosilca se ustvari skrita mapa **.glustersfs** s podmapami od 00 do FF, poleg tega se za vsako zgornjo datoteko ustvari enolični identifikator (GFID), ki poimenuje pripadajočo spodnjo datoteko. Metapodatki se shranjujejo v različnih razširjenih atributih.
- **eCryptfs** – Tu ne obstaja enolična hierarhija na spodnjem nivoju, hrani pa se enako na spodnjem in zgornjem datotečnem sistemu. Če je šifriranje omogočeno, se za vsako spodnjo datoteko uporabi posebna predpona ECRYPTFS_FNEK_ENCRYPTED. Spodnje datoteke prav tako vsebujejo posebne magične označevalce, ki jih je mogoče prepoznati z izvedbo operacije XOR.

2.6 Korelacija imen datotek

V procesu je ključno ugotoviti, kakšna je povezava med imeni zgornjih in ustreznimi spodnjimi datotekami ter preučiti, kako se hierarhija zgornjega datotečnega sistema odraža v spodnjem. Za posamezne sisteme se to izvede na naslednje načine:

- **MooseFS** – Ker niti glava (angl. *header*), ime datoteke in niti preostali metapodatki spodnje datoteke ne vsebujejo sklicev na izvorno zgornjo datoteko, je za korelacijo je treba analizirati metapodatke glavnega strežnika. Orodje **mfsmetadump** omogoča razčlenitev zapisov **EDGE** in **NODE**, kjer **EDGE** predstavlja relacijo med imenom in inode številko, **NODE** pa povezave do kosovnih datotek [10]. Izsek delovanja orodja je prikazan na sliki 3.

```
$ mfsmetadump metadata.mfs.back

# section header: NODE 1.4 (4E4F444520312E34) ; length: 145
# section type: NODE ; version: 0x14
# maxinode: 6 ; hashelements: 2
NODE|k:~|i:4|#:2|e:0x00|w:0x00|m:00644|u:0|g:0
|a:1695910450,m:1695910551,c:1695910551|t:24h|l:10|c:(00000000000000007)
NODE|k:~|i:3|#:2|e:0x00|w:0x00|m:00644|u:0|g:0
|a:1695910260,m:1695910427,c:1695910427|t:24h|l:526592|c:(0000000000000006)
-----
# section header: EDGE 1.1 (4544474520312E31) ; length: 79
# section type: EDGE ; version: 0x11
# nextedgeid: 7FFFFFFFFFFFFFFF2
EDGE|p:~|l:~|c:~|i:0|x:7FFFFFFFFFFFFFFF3|n:testfile
EDGE|p:~|l:~|c:~|i:0|x:7FFFFFFFFFFFFFFF4|n:chunkfile
```

Slika 3: Izsek iz orodja mfsmetadump, ki prikazuje metapodatke datotečnega sistema MooseFS.

- **GlusterFS** – GlusterFS za ugotavljanje hierarhije uporablja razširjene atribute in mehke povezave, pri čemer razširjeni atributi spodnje datoteke vsebujejo sklic na izvirno ime datoteke ter imenika GFID, v katerem je bila shranjena [12]. Ena izmed možnosti je tudi deljena številka inode med dvema datotekama, kar je razvidno na sliki 4.

```

$ file -r -b -o 2048 chunkserver1.hdd
# d/d 131074:      .glusterfs
[...]
# d/d 131082:      00
+++ d/d 131398:    00000000-0000-0000-0000-000000000001
+++ l/l 131339:    1477be7c-7a84-4c41-932b-5e0740c5e809
# d/d 131083:      01
+++ d/d 131084:      02
+++ d/d 131085:      03
# d/d 131106:      18
+++ d/d 131363:      47
+++ r/r 131362:    1847be7c-7a84-4c41-932b-5e0740c5e809
# d/d 131158:      4c
+++ d/d 131360:      db
+++ l/l 131361:    fcdcbcf-a8ab-4c3c-bd63-061445b3c380
+++ d/d 131322:      f0
+++ d/d 131352:      2b
+++ l/l 131353:    fcdcbcf-a8ab-4c3c-bd63-061445b3c380
# r/r 131346:      1MG0090.jpg
+++ d/d 131351:    documents
# r/r 131357:      cvc.docx
+++ d/d 131359:    Private
+++ r/r 131362:    data.txt
# d/d 131354:      Downloads

```

Slika 4: Primer hierarhije na spodnjem datotečnem sistemu v GlusterFS.

- **eCryptfs** – Korelacija med zgornjim in spodnjim datotečnim sistemom poteka preko inode števil (to že omogoča orodje `ecryptfs-find`). Če so imena datotek šifrirana, je potrebno dekodirati FNEK (File Name Encryption Key) iz šifriranega imena [7].

2.7 Rekonstrukcija vsebine datotek

2.7.1 Fragmentacija. V tradicionalnih datotečnih sistemih fragmentacija datotek povzroča izzive pri forenzičnih preiskavah, saj je potrebno spremljati podatkovne enote in njihov vrstni red. Za potrebe članka so avtorji ustvarili več večjih zgornjih datotek, da bi preučili doslej premalo raziskane vzorce fragmentacije v zloženih datotečnih sistemih. Prišli so do naslednjih zaključkov:

- **MooseFS** – Datoteke, večje od 64 MiB, se samodejno razdelijo na več kosov, ki so razporejeni med različne strežnike. Za rekonstrukcijo je potrebno pridobiti podatke iz glavnega

strežnika, kjer v datoteki z metapodatki shranjen seznam identifikatorjev kosov, iz katerih je datoteka sestavljena [10].

- **GlusterFS** – Fragmentacija in podvajanje spodnjih datotek se razlikujeta glede na vrsto uporabljenega volumna; v porazdeljenem načinu (angl. *distributed*) datoteke niso razdrobljene, temveč razpršene po vseh gradnikih (angl. *bricks*), medtem ko način podvajanja (angl. *replica*) ustvari več kopij iste datoteke, ki jih je mogoče prepoznati po istem GFID. Razpršeni način povzroči fragmentacijo zgornjih datotek preko več strežnikov, pri čemer se za identifikacijo kosov uporabljajo razširjeni atributi in je za rekonstrukcijo datoteke potrebna uporaba Erasure kodiranja [2].
- **eCryptfs** – Zaradi lokalne narave sistema fragmentacija ni prisotna. Vsaka zgornja datoteka ima eno ustrezno spodnjo šifrirano datoteko.

2.7.2 Transformacija podatkov. Zaradi različnih vrst transformacij podatkov, kot sta npr. kodiranje in kompresija, vsebina datoteke na disku ni enaka izvirni. To še dodatno oteži neposredno analizo:

- **MooseFS** – Doda glavo kosu, ki vključuje metapodatke in kontrole CRC. Ker je vsebina datoteke nešifrirana, jo je enostavno pridobiti [10].
- **GlusterFS** – Pri porazdeljenih in podvojenih nosilcih se vsebina spodnjih datotek ne spremeni, pri razpršenih nosilcih so podatki zaščiteni z naprednimi kodirnimi tehnikami, zato je za rekonstrukcijo potrebna ponovna vzpostavitev prvotne GlusterFS konfiguracije. Pomembno je, da se ohranijo razširjeni atributi in pravilni vrstni red gradnikov [12].
- **eCryptfs** – Podatki v spodnjih datotekah so v celoti šifrirani. Pred dejanskimi podatki je dodana posebna glava z velikostjo najmanj 8192 bajtov, ki vključuje kriptografski kontekst posamezne datoteke, nešifrirano informacijo o velikosti izvirne datoteke, različico sistema in šifriran sejni ključ.

2.8 Časovni žigi

Časovni žigi se v tradicionalnih sistemih shranjujejo kot del metapodatkov datotek in vključujejo podatke o dostopu, spremembi vsebine datoteke, spremembi metapodatkov ter občasno tudi o nastanku datoteke. V primeru zloženih datotečnih sistemov so prisotni žigi iz obeh nivojev, kar zahteva dodatno razumevanje medsebojnega vpliva in posodabljanja. V nadaljevanju je opisano, na kakšen način se posodablajo časovni žigi na obeh ravneh sistema:

- **MooseFS** – Časovni žigi se beležijo v metapodatkih na glavnem strežniku ali v datoteki Metalogger. Ob spremembi datoteke se posodobita časa spremembe vsebine in metapodatkov (angl. *Modification Time*, *Change Time*), medtem ko se čas dostopa (angl. *Access Time*) lahko spreminja glede na način predpomnjenja [10].
- **GlusterFS** – Časovni žigi se hranijo v razširjenih atributih v **glusterfs** direktoriju, kodirani v Base64 formatu (struktura razširjenega atributa je prikazana na sliki 5). Branje metapodatkov omogoča natančno določitev časa kreacije, spremembe in dostopa [12].
- **eCryptfs** – Časovni žigi so neposredno prevzeti iz spodnjega datotečnega sistema (npr. ext4). To pomeni, da je njihova analiza enaka analizi tradicionalnih sistemov.

00000000	01	00 00 00 00 00 00 00 00	00 00 00 00 00 65 0b 24
00000010	c3	00 00 00 00 36 57 96 54	00 00 00 00 65 0b 24
00000020	aa	00 00 00 00 0d 80 9b 59	00 00 00 00 65 0b 24
00000030	aa	00 00 00 00 0d 52 47 a5	

Change Modification Access

Slika 5: Struktura razširjenega atributa `trusted.glusterfs.mdata`, ki vsebuje časovne žige v GlusterFS.

2.9 Neuporabljeni prostor in načini skrivanja podatkov

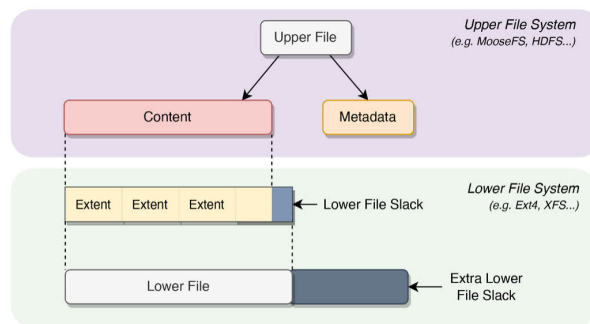
Navadno lahko neuporabljen prostor (angl. *slack*) med koncem datoteke in koncem podatkovne enote vsebuje ostanke prejšnjih podatkov, ali pa se celo uporablja za skrivanje novih. V zloženih sistemih najdemo dva tipa neuporabljenega prostora, ki sta razvidna tudi iz slike 6:

- **Spodnje datotečni neuporabljen prostor (angl. *lower file slack*)** – Pri poravnavi podatkov na velikosti blokov lahko nastane neizkoriščen prostor. V MooseFS kosih, ki so večji od vsebine datoteke, ostaja neizkoriščen prostor, ki ga je mogoče uporabiti za skrivanje podatkov [11]. V porazdeljenem in načinu podvajanja skrivanje podatkov ni mogoče, saj spodnje datoteke točno ustrezajo zgornjim datotekam, medtem ko v razpršenem uspešno skrivanje podatkov prav tako ni zagotovljeno zaradi spreminjanja podatkov na zgornji plasti. Pri eCryptfs je mogoče skriti podatke v spodnjem datotečnem prostoru, vendar se ti podatki izgubijo, ko se zgornja datoteka spremeni, saj se celotna vsebina dopolnilnega prostora (angl. *padding*) prepiše.
- **Dodatni spodnji datotečni neuporabljen prostor (angl. *extra lower file slack*)** – Če napadalec ročno poveča velikost spodnje datoteke brez vpliva na zgornji sistem, lahko v ta dodani del shrani skrite podatke, ki niso vidni v običajni analizi [6]. V MooseFS lahko podatke skrijemo tako, da jih dodamo obstoječi spodnji datoteki, pri čemer je shranjevanje podatkov zaščiteno pred prepisovanjem zaradi sprememb zgornje datoteke. Vseeno pa skriti podatki izginejo, ko je kos prenesen na drug strežnik. V GlusterFS se tak način izkaže za nepraktičnega, saj so skriti podatki vidni ob branju zgornje datoteke, izbrisani ali pa povzročijo napako pri vhodno-izhodnih operacijah. Pri eCryptfs se skriti podatki v spodnji datoteki ohranijo tudi po spremembah, vsaj dokler ni bila presežena velikost dopolnilnega prostora, zato je za njihovo varnost potrebno dovolj veliko prostora.

2.10 Možnosti za obnovo izbranih datotek

Dinamika brisanja datotek je odvisna od posamičnega datotečnega sistema. V poskusih tega članka je bil za proces brisanja uporabljen ukaz `rm`, s čimer so avtorji zaobšli "koš za smeti" operacijskega sistema. Ugotovitve so zapisane spodaj:

- **MooseFS** – Implementiran je vgrajen Trash sistem, ki omogoča obnovo izbranih datotek v 24-urnem obdobju. Poleg tega lahko analiza ostankov kosov na strežnikih omogoči obnovo še po preteku tega časa. Ko je datoteka popolnoma izbrisana, rekonstrukcija ni več mogoča [10].



Slika 6: Pregled možnosti neuporabljenega prostora v zloženih datotečnih sistemih.

- **GlusterFS** – V privzeti nastavitvi ne uporablja funkcionalnosti koša, ob brisanju odstranijo tako zgornje kot spodnje datoteke. V kolikor je Trash translator omogočen, premakne izbrisane datoteke v imenik `.trashcan`, s čimer omogoči njihovo naknadno obnovo brez dostopa do vseh strežnikov hkrati [12].
- **eCryptfs** – Takšen mehanizem ni implementiran. Izbrisana datoteka je odstranjena tudi iz spodnjega sistema, zato je možna obnova le preko tradicionalnega "file carving" postopka [13].

3 DATOTEČNI SISTEM OVERLAYFS

Kot dodatek prvotnemu članku, katerega ugotovitve smo zapisali v prejšnjem poglavju, je v nadaljevanju podana analiza zloženega datotečnega sistema OverlayFS [4]. Glavna lastnost tega sistema je, da iz tako zgornjega kot iz spodnjega sloja skupaj sestavi nov datotečni sistem. Slednja združi tako, da se ob ujemanju imena datoteke upošteva vsebina zgornjega sloja. Spodnji sloj sicer deluje v bralnem načinu in se posledično ne spreminja, zgornji sloj pa v načinu pisanja, ki se uporablja za shranjevanje sprememb na naloženem sistemu.

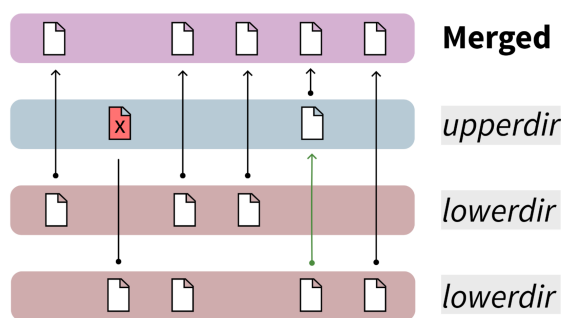
OverlayFS vzpostavimo z ukazom `mount -t overlay`, ki podpira naslednje parametre (slika 7):

- **lowerdir** – pot do datotečnih sistemov spodnjega sloja (vsaj ena).
- **upperdir** – pot do datotečnega sistema zgornjega sloja (največ ena, lahko tudi nobena).
- **workdir** – pot do imenika, ki bo imel vlogo delovnega imenika. Uporabljen je za pripravo datotek za atomično transakcijo do združenega datotečnega sistema. V kolikor **upperdir** ni bil definiran, je parameter neobvezen, sicer pa se mora imenik nahajati v zgornjem sloju.

3.1 Analiza datotečnega sistema

V tem podpoglavju sistem analiziramo na podoben način kot so avtorji analizirali preostale datotečne sisteme v izvornem članku.

- **Identifikacija sistema na spodnjem sloju:** OverlayFS sam ne pušča sledi oziroma ne hrani metapodatkov. Ker je del Linuxovega jedra, lahko njegovo konfiguracijo zasledimo preko datotek kot je `/etc/fstab`.



Slika 7: Primeri prekrivanja datotek v OverlayFS.

- **Korelacija imen datotek:** Za korelacijo datotek ni potrebne nobene dodatne analize.
- **Rekonstrukcija vsebine datotek:** Uporabi se pristop datotečnih sistemov spodnjega in zgornjega sloja. Vreden je ogled **workdir** direktorija, saj lahko vsebuje datoteke, za katere je spodletela atomična transakcija (izpad elektrike ipd.).
- **Časovni žigi:** Pomembno je, da spremljamo časovne žige datotek v zgornjem in spodnjem sloju. Ker se spodnji sloj ne spreminja, ostanejo tam časovni žigi nespremenjeni, medtem ko se ti v zgornjem sloju spreminjajo na običajen način. Če datoteka obstaja samo v spodnjem sloju, imamo podatek o tem, da se ta preko OverlayFS ni spreminjala, na drugi strani pa nimamo podatka za najnovejšo čase dostopa.
- **Neuporabljeni prostor in načini skrivanja podatkov:** Pomembno je, da pregledamo neuporabljeni prostor na datotečnih sistemih zgornjega in spodnjega sloja posamezno.
- **Možnosti za obnovo izbranih datotek:** OverlayFS nima implementiranega sistema za izbrisane datoteke. Za obnovo je potreben tradicionalni postopek na vseh datotečnih sistemih, uporabljenih v slojih OverlayFS.

Analiza brez zavedanja prisotnosti OverlayFS konfiguracije oteži preiskovalni postopek in povzroči, da pride do dvoumnosti pri beleženju vsebine in časovnih žigov posameznih datotek. Tako nam manjka tudi kontekst datotek, saj so na zgornjem sloju lahko zapisane zgolj spremenjene datoteke spodnjega sloja.

4 ZAKLJUČEK

Z naraščajočo uporabo oblčnih storitev, distribuiranih omrežij in šifriranja je poznavanje delovanja zloženih datotečnih sistemov ključno za vsakega digitalnega forenzičnega preiskovalca. Analiza mora zajemati tako zgornje kot spodnje sloje sistema, upoštevati transformacije, fragmentacije in omogočiti rekonstrukcijo časovnic na osnovi metapodatkov. Avtorji opozarjajo, da trenutna forenzična orodja še vedno pogosto niso prilagojena za obravnavo takšnih arhitektur, zato je nadaljnji razvoj nujen.

LITERATURA

- [1] Akshat Aranya, Charles P Wright, and Erez Zadok. 2004. Tracefs: A File System to Trace Them All.. In *FAST*. 129–145.

- [2] Muhammad Asim et al. 2019. Big data forensics: Hadoop distributed file systems as a case study. In *Handbook of Big Data and IoT Security*. Springer, 179–210.
- [3] W. A. Bhat and S. M. K. Quadri. 2012. restFS: Secure data deletion using reliable & efficient stackable file system. In *2012 IEEE 10th International Symposium on Applied Machine Intelligence and Informatics (SAMI)*. 457–462. <https://doi.org/10.1109/SAMI.2012.6209010>
- [4] Neil Brown. 2025. *Overlay Filesystem*. <https://www.kernel.org/doc/html/latest/filesystems/overlayfs.html> Dostop: 9.6.2025.
- [5] Brian Carrier. 2005. *File System Forensic Analysis*. Addison-Wesley Professional.
- [6] Tobias Göbel et al. 2019. Revisiting data hiding techniques for apple file system. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*. ACM, 1–10.
- [7] Michael Halcrow. 2005. An enterprise-class encrypted filesystem for Linux. In *Proceedings of the Linux Symposium*, Vol. 1. 201–218.
- [8] Edward Harshany, Ryan Benton, David Bourrie, and William Glisson. 2020. Big Data Forensics: Hadoop 3.2.0 Reconstruction. *Forensic Science International: Digital Investigation* 32 (2020), 300909. <https://doi.org/10.1016/j.fsidi.2020.300909>
- [9] John S Heidemann and Gerald J Popek. 1994. File-system development with stackable layers. In *ACM Transactions on Computer Systems (TOCS)*, Vol. 12. 58–89.
- [10] Jan-Niclas Hilgert, Martin Lambert, and Daniel Baier. 2024. Forensic implications of stacked file systems. *Forensic Science International: Digital Investigation* 48 (2024).
- [11] Ewa Huebner et al. 2006. Data hiding in the NTFS file system. *Digital Investigation* 3, 4 (2006), 211–226.
- [12] Ben Martini and Kim-Kwang Raymond Choo. 2014. Distributed filesystem forensics: XtremFS as a case study. *Digital Investigation* 11, 4 (2014), 295–313.
- [13] Vincent van der Meer et al. 2021. A contemporary investigation of NTFS file fragmentation. *Forensic Science International: Digital Investigation* 38 (2021), 301125.
- [14] Erez Zadok. 1999. *Stackable file systems as a security tool*. Technical Report. Citeseer.

Forenzična analiza naprave Steam Deck

Predstavitev in razširitev članka Well Played, Suspect!

Eva Gabersček

Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija

Miha Mitič

Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija

Jan Ljubič

Fakulteta za matematiko in fiziko,
Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija

Aljaž Škof

Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija

Povzetek

V članku analiziramo področje digitalne forenzike na igralnih konzolah in igrah. Osredotočamo se predvsem na članek Well Played, Suspect! [4], ki je bil osnova za našo raziskavo. V prvem delu predstavimo postopek diferencialne forenzike in opišemo napravo Steam Deck ter operacijski sistem SteamOS, ki sta glavno področje naše raziskave. V drugem delu naredimo pregled področja, kjer opišemo sorodne članke [1, 2, 11] ter njihove ugotovitve, poleg tega še podrobneje predstavimo članek [4], ki je osnova za naše delo. Nato opišemo praktični del našega članka, kjer pridobimo slike fizične in virtualne naprave in na njih preverimo delovanje dodatka oziroma razširitve za orodje Autopsy, ki so jo napisali avtorji predhodnega članka. Razširitev tudi popravimo. Na koncu predstavimo rezultate in primerjamo delovanje razširitve na obeh napravah.

CCS Koncepti

• Računalništvo v praksi → Digitalna forenzika; Sistemska forenzika.

Ključne besede

Digitalna forenzika, Diferencialna forenzika, SteamOS, Steam Deck, Nintendo 3DS, PlayStation 4, Autopsy

1 Uvod

Nenehni razvoj in miniaturizacija računalnikov nam omogoča prenosno računaljivost kjerkoli in kadarkoli. Ta razvoj nam poleg delovnih pripomočkov omogoča tudi prenosne digitalne vsebine in digitalne igre. Leta 1979 je podjetje Milton Bradley izdalo prvo prenosljivo konzolo Microvision. Druga podjetja so se hitro pridružila razvoju svojih konzol. Zaradi tehnoloških omejitev se je na napravo včasih shranjevalo zelo malo uporabnikovih podatkov, kar je omejilo uporabnost naprav v forenzičnih preiskavah. Z napredkom v zmogljivosti in učinkovitosti majhnih procesorjev se želja po vedno bolj naprednih in računsko potratnejših igrah povečuje. Današnje konzole zelo pogosto zaradi olajšanega razvoja uporabljajo vnaprej pripravljene operacijske sisteme, najpogosteje Linux. Tej operacijski sistemi hranijo ogromno količino zapisov, kot so čas igranja, čas prižiga in lokacijo, kar pripomore k forenzični preiskavi. Ena izmed teh naprav je Steam Deck, ki ga v tem članku podrobneje analiziramo. Napravo pregledamo s pomočjo Autopsy razširitve in ocenimo učinkovitost uporabljenih pristopov.

1.1 Diferencialna forenzična analiza

Diferencialna analiza je način pregledovanja forenzičnih artefaktov, kjer med seboj primerjamo različne posnetke nekega sistema v času in pregledamo razlike med njimi. Na tak način je lažje izluščiti aktivnost uporabnika v tem časovnem obdobju. Razliko med slikami je pogosto najbolje izraziti kot zaporedje operacij, potrebnih za spremembo. Razvitih je bilo več različnih orodij, ki uporabljajo diferencialno analizo. Pojavlja se v tekstovni analizi datotek, kot tudi v ročnih in avtomatiziranih orodjih. Izven forenzike je takšna analiza prisotna tudi pri sinhronizaciji sistemov in v sistemih za nadziranje različic [8].

1.2 Steam Deck

Leta 2017 je podjetje Nintendo revolucionariziralo dostopnost video iger z izdajo svoje igralne konzole Nintendo Switch. Zaradi večjega zanimanja za ročne igralne konzole je leta 2022 na ta trg vstopilo tudi podjetje Valve z izdajo naprave Steam Deck, ki ga lahko vidimo na sliki 1. S to napravo so omogočili dostop do večjega števila računalniških iger s poudarkom na igrah s platforme Steam na prenosni konzoli. Naprava uporablja operacijski sistem SteamOS, katerega podrobneje opišemo v poglavju 1.3. Naprava ima AMD APU, ki vsebuje Zen2 CPE (2.4–3.5 GHz, 4 jedra, 8 niti) in 8 RDNA GPE (1.0–1.6 GHz), ter 16 GB LPDDR5 RAM spomina. Na voljo je več modelov z različnimi velikostmi diskov od 64 GB do 1 TB in z različnimi tipi zaslonov. Povezavo med napravami omogočata tehnologiji Bluetooth in Wi-Fi [12].



Slika 1: Konzola Steam Deck v igralnem načinu [12].

1.3 SteamOS

Operacijski sistem SteamOS je podjetje Valve razvilo za konzole Steam Deck in predhodno za naprave Steam Machine. Pretekle

verzije so bile zgrajene na osnovi Linux distribucije Debian, trenutna verzija pa je zasnovana na distribuciji Arch [4, 12]. Gre za tretjo verzijo sistema, imenovano Holo, ki je naložena na napravah Steam Deck. Podpirajo ga tudi druge konzole kot sta Legion Go S in ASUS ROG Ally. SteamOS podpira dva grafična načina: igralni, kjer lahko uporabnik enostavno dostopa do naloženih iger in svoje knjižnice na programu Steam, ter namizni način, ki omogoča normalno uporabo operacijskega sistema.

2 Pregled področja

V naslednjem poglavju si ogledamo nekaj sorodnih tem v povezavi z video igrami, sistemi za video igre in odkritji avtorjev člankov. Najprej predstavimo kako se področje forenzike spoprijema z uporabniškimi dodatki in razširitvami priljubljenih video iger, opišemo pa tudi forenzične postopke pri preiskovanju drugih igralnih sistemov, kot sta Nintendo 3DS in PlayStation 4.

2.1 Članek Emerging cyberworld attack vectors: Modification, customization, secretive communications, and digital forensics in PC video games [2]

V članku [2] avtorja analizirata video igre s stališča varnosti in možnosti novih vektorjev napadov z njihovo uporabo.

Mnoge igre omogočajo kreacijo svojih nivojev ali svetov v sami igri, kar omogoča veliko vsebine, ustvarjene po meri. To predstavlja izziv za forenzično preiskovanje, saj odpira nove kanale za komunikacijo in skrivanje informacij.

Igra World of Warcraft ima veliko možnosti za dodajanje in spreminjanje mnogih aspektov igranja, kot tudi spreminjanje modelov likov ali tekstur. Ena izmed glavnih nevarnosti preži v sistemu dodatkov (ang. *add-ons*), saj avtorji igre ne preverjajo, ali so dodatki, narejeni s strani igralcev, varni. To bi tudi bil kar precejšen izziv, saj je dodatkov zelo veliko. Mnogi dodatki so narejeni z namenom spreminjanja uporabniškega vmesnika, saj na ta način igralci lahko v igro zase izpostavijo mnoge uporabne podatke, ki omogočajo analizo spopadov s sovražniki ali pokažejo na nevarnosti. Dodatki so narejeni s pomočjo programskega jezika Lua in XML. Čeprav je Lua interpretiran jezik, večina igralcev ne zna ali ni pripravljena brati kode pred zagonom dodatka. Lua omogoča potencialnemu napadalcu popoln dostop do sistema, kar lahko izpostavi igralce nevarnosti, ko na slepo nalagajo dodatke s spleta.

Pri neuradnih dodatkih in po meri narejeni vsebini je mnogokrat težje zaznati zlonamerno vsebino, saj je vanje mogoče zakriti ali zakodirati skrivna sporočila, katera je težko zaznati. Z računanjem zgoščevalnih funkcij osnovne ter modificirane različice igre in primerjavo vrednosti lahko najdemo spremenjene datoteke in jih lažje analiziramo. Pomembno je tudi pregledati dodatke in modificirane zaganjalnike iger.

Moderne igre predstavljajo posebne izzive, saj jih raziskovalci in forenziki mnogokrat spregledajo kot vir informacij oziroma jih nimajo za vredne pregleda, kljub temu da se v njih morda skrivajo možni dokazi. Pogoste posodobitve otežujejo delo, saj pri igrah z velikim številom posodobitev lahko nekdo izgubi voljo do rednega pregledovanja datotek, ki jih posodobitve prinesejo. Seveda pa igre dandanes omogočajo veliko načinov skrivanja vsebine, ki jih mnoga forenzična orodja ne znajo odkriti.

2.2 Članek A forensic methodology for analyzing Nintendo 3DS devices [11]

Članek A forensic methodology for analyzing Nintendo 3DS devices [11] na kratko predstavi metodologijo pri analizi igralne konzole Nintendo 3DS.

Članek na začetku opiše in popiše funkcije in zmogljivosti konzole Nintendo 3DS kot so povezovanje preko omrežij Wi-Fi, vgrajen spletni brskalnik ter komunikacijski sistem, podoben e-pošti. Naprava ima možnost izmenjave podatkov z drugimi Nintendo 3DS konzolami v bližini, tako v budnem stanju kot tudi v stanju spanja. Različica konzole Nintendo 3DSi, ki vsebuje tudi digitalno kamero, s katero lahko uporabnik ustvari slike in videoposnetke.

Konzola Nintendo 3DS za shranjevanje podatkov primarno uporablja igralne kartice in SD kartice, vsebuje pa tudi vgrajeni NAND flash pomnilnik, ki je del matične plošče. Ta čip je v članku [10] tudi natančneje forenzično preučen.



Slika 2: Zapisi na konzoli Nintendo 3DS [11].

Konzola za povezovanje na Wi-Fi omrežja uporablja protokol 802.11b/g in hrani podatke o največ 3 dostopnih točkah naenkrat. Podatke o povezavi lahko pregledamo v nastavitvah naprave, kjer vidimo podatke o trenutni povezavi in MAC naslovu.

Dnevnik aktivnosti spremlja in shranjuje podatke, kot so na primer čas prve in zadnje uporabe aplikacije na sistemu in čas igranja, kot lahko vidimo na sliki 2. Spletni brskalnik shrani zgodovino iskanja do 32 obiskanih strani in do 64 zaznamkov.

Avtorji članka opišejo metodologijo forenzične analize konzole Nintendo 3DS:

- (1) Pričnemo s snemanjem za dokumentacijo vseh interakcij s konzolo.
- (2) Preverimo status konzole (povezava na internet, ali je prižgana, ipd.). V primeru, da konzola še ni ugasnjena, si ogledamo možne zanimive točke in zapišemo njihovo stanje, nato konzolo izklopimo s pritiskom in držanjem gumba za vklop. S tem je konzola izklopljena, dnevnik aktivnosti pa se posodobi.
- (3) Previdno odstranimo SD kartico in jo analiziramo ločeno od ostalih komponent. Naredimo njeno kopijo in na njej nato

Predstavitev in razširitev članka Well Played, Suspect!

izvedemo analizo. Podatki na SD kartici niso kriptirani, zato njihovo branje ne predstavlja težav.

- (4) Odstranimo igralno kartušo in shranimo podatke z nje. Preverimo, ali hrani video igranje ali aplikacijo.
- (5) Vključimo Nintendo 3DS. Onemogočimo povezavo na omrežje z izklopom Wi-Fi s premikom drsnika ob strani konzole ali s postavitvijo konzole v Faradayevo kletko. S tem preprečimo, da konzola zazna možne posodobitve programske opreme, ki lahko onemogočijo dostop do nekaterih lokacij in datotek.
- (6) Pregledamo dnevnik aktivnosti. To naredimo kot eno prvih stvari, saj se v dnevnik zapiše tudi že sam dostop do njega.
- (7) Pregledamo seznam dodanih prijateljev.
- (8) Preverimo podatke o omrežju. V podmeniju najdemo tudi MAC naslov konzole.
- (9) Pregledamo spletni iskalnik. Z uporabo spletnega iskalnika lahko potrdimo trenutno stanje povezave na omrežje in nastavitve posredovanja.
- (10) Pregledamo aplikacijo Game Notes. Tu so shranjeni ročno napisani zapiski, ki jih ustvari uporabnik.
- (11) Pregledamo aplikacijo Camera. Prek nje lahko dostopamo do slik in videoposnetkov, ki so shranjeni na SD kartici.
- (12) Pregledamo komunikacije aplikacije StreetPass, preko katere si bližnje Nintendo 3DS lahko izmenjuje podatke, kot so na primer liki Mii.

V članku [11] avtorji sklenejo, da je forenzična analiza sistemov znamke Nintendo zahtevna, saj morajo forenziki zelo natančno poznati delovanje in njegovo sestavo, da lahko pridobijo podatke, ki niso hitro dostopni povprečnemu uporabniku.

2.3 Članek Investigating The Security and Digital Forensics of Video Games and Gaming Systems [1]

V članku Investigating The Security and Digital Forensics of Video Games and Gaming Systems [1] avtorji pregledajo varnost nekaterih popularnejših računalniških iger in igralne konzole PlayStation 4, vidni na sliki 3, ter različne poglede nanje s strani digitalne forenzike.



Slika 3: Konzola PlayStation4 [5].

Avtorji članka so pregledali napravo PlayStation 4 in nato testirali, na katere vrste napadov je odporna. Najprej so poizkusili ARP

zastupljanje tako, da so v isto LAN omrežje povezali računalnik ter PlayStation 4. Z uporabo orodja Cain and Abel na operacijskem sistemu Windows so skenirali MAC naslove v omrežju in našli IP naslov, ki je pripadal PlayStationu. Nato so izvedli še sam napad in uspešno prevzeli pakete, ki so bili namenjeni za PlayStation.

Poizkušen je bil tudi napad odpovedi storitve. Za to so uporabili orodje t50 na 64-bitnem sistemu z nameščenim Kali operacijskim sistemom. Z namenom napravo upočasniti, so njen IP naslov zasipali s podatkovnimi paketi. Napad je uspel, saj je bil PlayStation preobremenjen in ni uspel več sprejemati drugih paketov, dokler niso ročno ustavili napada. V nekaterih primerih so morali PlayStation po koncu napada celo ponovno zagnati, da se je povrnil v delujoče stanje. PlayStation 4 je, podobno kot veliko drugih naprav, ranljiv na standardne napade preko omrežja.

Trdi disk naprave PlayStation 4 je kriptiran in ne omogoča nezaželenega dostopa do datotek na njem. Avtorji so disk tudi kopirali s forenzičnim orodjem FTK Imager, vendar jim to ni pomagalo pridobiti uporabnih informacij. Disk PlayStation-a 4, ki so testirali, je imel neznan datotečni sistem s 15 particijami. Ugotovili so, da je PlayStation-ov operacijski sistem osnovan na FreeBSD 9.0. Kljub temu je iz naprave možno pridobiti nekatere podatke, kot so na primer uporabnikove interakcije in zgodovina brskalnika. Za to se opravi pregled uporabniškega vmesnika ali analiza omrežnega prometa.

2.4 Članek Well Played, Suspect! [4]

V članku Well Played, Suspect! [4] avtorji analizirajo napravo Steam Deck z diferencialno analizo. Zastavijo deset stanj sistema, ki se po vrsti razlikujejo po enem dejanju ali sklopu akcij. Na vsakem koraku so napravo ugasnili in naredili kopijo diska ter nato primerjali, kako se med seboj razlikujejo. Posamezne akcije ponazarjajo vsakdanjo uporabo sistema. S primerjavo različnih stanj lahko lažje določimo kateri atrefakti pripadajo kateremu stanju. Spodaj so napisani posamezni koraki:

- σ_0 Tovarniške nastavitve naprave, slika diska pred zagonom.
- σ_1 Prvi zagon naprave, povezava na brezžično omrežje in prva posodobitev.
- σ_2 Povezava na drugo brezžično omrežje, vpis v Steam račun, oboje nastavljeno, da si sistem zapomni. Ponovni zagon in posodobitev.
- σ_3 Iskanje igre CS:GO, namestitve, igranje in dva posnetka zaslona.
- σ_4 Izbris igre CS:GO.
- σ_5 Dodajanje denarja na račun preko kartice in preko darilne kartice ter dodajanje prijatelja.
- σ_6 Pošiljanje sporočil.
- σ_7 Pogovor preko VoIP (Voice over IP).
- σ_8 Odstranitev prijatelja.
- σ_9 Ponastavitev na tovarniške nastavitve.

Slike diskov s sosednjimi stanji (σ_k, σ_{k+1}) so avtorji med seboj primerjali z orodjem idifference2.py [7]. Med različnimi pari so našli od 89 do 719 relevantnih razlik.

Ugotovili so, da ima sistem particije esp, home in po dva para particij za efi-A/B, rootfs-A/B ter var-A/B. Različici A in B se aktivirata izmenično. Razlog za to je, da se lahko na neaktivni

participiji izvajajo posodobitve, spremembe ene pa se nato kopirajo na drugo.

Najdenih je bilo več pomembnejših artefaktov:

- konfiguracija particij,
- podatki o napravi,
- podatki o zapomnjenih omrežjih,
- podatki o uporabnikih in njihovem delovanju,
- naložene igre, čas igranja in zadnji dostop,
- posnetki zaslona, kdaj in v kateri igri so bili ustvarjeni,
- podatki o prijateljih oziroma stikih,
- podatki o plačilu s kartico,
- beleženje dogodkov,
- indikatorji o tovarniški ponastavitvi.

Avtorji so na podlagi teh podatkov izdelali razširitev za orodje Autopsy za pregledovanje naprave Steam Deck. Preizkusili so jo na vseh prej pridobljenih slikah, kot tudi na kasnejši sliki naprave z dolgotrajnejšo uporabo. Razširitev se je izkazala za uspešno tudi na kasnejših posodobljenih različicah sistema.

3 Metodologija

Naš prispevek k področju forenzike temelji na praktični uporabi rezultatov ugotovitev avtorjev članka Well Played, Suspect! [4].

Zanima nas delovanje razširitve, ki so jo izdelali avtorji članka, na različnih sistemih. Razširitev je bila namreč izdelana za pregled fizičnih naprav Steam Deck, na katerih je verzija operacijskega sistema SteamOS Holo. Podjetje Valve je javno objavilo tudi sliko za obnovitev sistema (ang. *recovery image*), iz katerega je mogoče pridobiti tudi virtualno različico sistema SteamOS Holo [13]. Zanima nas, ali je mogoče dodatek uporabiti tudi na sliki virtualne naprave, zato smo pridobili dve sliki sistema SteamOS.

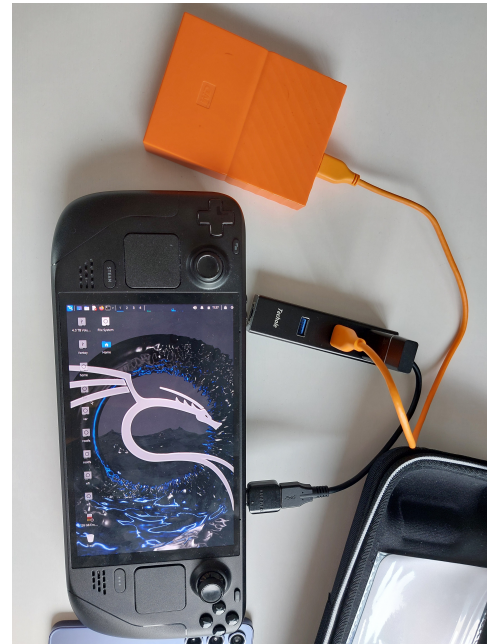
3.0.1 Fizična naprava.

Prvo sliko smo pridobili iz fizične naprave, ki vsebuje disk velikosti 1 TB. Ker je fizičen disk zaščiten in naprava izposojena, je nismo želeli poškodovati, zato smo kopiranje diska izvedli na napravi. Nanjo smo priključili priključno postajo z napajanjem (ang. *powered USB hub*) z več USB vhodi. Na en vhod smo priključili prazen zunanji disk velikosti 4 TB, na drugega pa USB ključ, na katerem je naložen program Ventoy [14], ki omogoča zagon operacijskega sistema v živo. Z držanjem gumba za povečanje glasnosti in gumba za zagon hkrati, smo Steam Deck prižgali v zagonski nalagalnik. Izbrali smo sistem Kali Linux [9], s katerim smo nato namestili disk v bralnem načinu (ro) in naredili kopijo diska v napravi Steam Deck na zunanji disk s pomočjo orodja dd. Izvedbo kopiranja smo dokumentirali in jo prikazemo na sliki 4.

3.0.2 Virtualna naprava.

Za drugo sliko smo ustvarili virtualno napravo, na kateri smo simulirali običajno uporabo sistema. Pri postavljanju sistema smo sledili navodilom v članku Running the Steam Deck's OS in a virtual machine using QEMU [6]. Sliko smo pretvorili v format, ki ga podpira QEMU emulator in sistem zagnali v Proxmox virtualnem okolju. Ker gre za obnovitveno sliko, je iz nje nato potrebno naložiti celoten sistem. V virtualnem okolju igralni način ne deluje, zato ga pred zagonom onemogočimo.

Dobili smo delujočo sliko sistema, na katerem smo nato simulirali nekaj korakov običajne uporabe. Za osnovo smo vzeli korake



Slika 4: Kopiranje diska naprave Steam Deck.

Na sliki še ne uporabljamo postaje z napajanjem, brez katere zunanji disk ni delal.

opisane v poglavju 2.4. Vpisali smo se v osebni Steam račun in naložili igro Baba is You. Za to igro smo se odločili zaradi njene preprostosti, saj v virtualnem okolju zaradi omejene strojne opreme ni bilo mogoče igrati zahtevnejših iger. Igro smo nekaj časa igrali ter naredili nekaj posnetkov zaslona in poslali nekaj testnih sporočil.

Med simulacijo testov se je nekajkrat zgodilo, da je bil uporabniški račun avtomatsko odjavljen ob hkratni uporabi na dveh napravah.

Obe sliki smo nato pregledali z orodjem Autopsy in razširitvijo, ki so jo napisali avtorji predhodnega članka [3].

4 Analiza in rezultati

Pri analizi rezultatov smo primerjali izpise razširitve [3] med obema slikama ter ročno pregledali določene datoteke. Pri uvozu slik v Autopsy smo izbrali le module iz razširitve. Po zaključeni analizi smo iz orodja izvozili HTML poročilo o najdenih rezultatih.

4.0.1 Razdelki poročila.

Poročilo je sestavljeno iz večih sklopov, ki ustrezajo modulom razširitve, kot jih lahko vidimo na sliki 5.

V prvem sklopu Boot Partitions najdemo podatke o A/B razdelkih sistema in število zagonov za vsakega. Pri obeh napravah smo ugotovili, da sledita A/B razdelitvi razdelkov. Sklop Device vsebuje podatke o sistemu in napravi, kot so verzija sistema, identifikacijska številka in ime naprave. Factory Reset vrne le podatek o tem, ali je naprava bila ponastavljena na tovarniške nastavitve, kar je lahko za forenzično preiskavo znak, da so bili nekateri podatki izgubljeni.

Report Navigation

- 📁 Case Summary
- ★ Steam Deck - Boot Partitions (2)
- ★ Steam Deck - Device (70)
- ★ Steam Deck - Factory Reset (1)
- ★ Steam Deck - Friends (140)
- ★ Steam Deck - Games and Apps (28)
- ★ Steam Deck - Log Entries (Slow) (2915)
- ★ Steam Deck - Log Files (53)
- ★ Steam Deck - Power History (1614)
- ★ Steam Deck - Screenshots (10)
- ★ Steam Deck - Secrets (10)
- ★ Steam Deck - Users (1)
- ★ Steam Deck - Web: Cookies (65)
- ★ Steam Deck - Wi-Fi (12)

Slika 5: Razdelki poročila razširitve.

Razdelek Users ima podatke o uporabnikih naprave, v Friends so podatki o stikih uporabnika. Vidimo lahko njihovo identifikacijsko številko, trenutno uporabniško ime in zgodovino vseh uporabniških imen. Games and Apps vsebuje podatke o vseh igrah v uporabnikovi knjižnici od zadnjega zagona, celotnega časa igranja do časa igranja v zadnjih dveh tednih, izpisani so tudi ločeni podatki za čase igranja na napravi. Primera izpisov prikazemo v izsekih 1 in 2. Sklop Screenshots vsebuje posnetke zaslona in njihov čas nastanka. Dodana je tudi identifikacijska številka aplikacije ali igre, ki je na posnetku.

Razdelek Log Files vrne seznam vseh dnevninskih datotek; v Log Entries so napisani vsi dnevniški zapisi na napravi. Ostanajo še Secrets, ki vsebuje gesla, ključe in žetone, Web: Cookies, s spletnimi piškotki in Wi-Fi, kjer se nahajajo podatki zapomnjenih omrežij, vključno z gesli v čistopisu.

```
Last Played: 2025-05-12 23:45:59
Last Played (Epoch): 1747086359
Last Sync State (Cloud): changeslocally
Last Updated: 2025-05-09 14:17:47
Last Updated (Epoch): 1746793067
Name: Baba Is You
Owner: [REDACTED]
Playtime: 1243
Playtime (2 weeks): 23
```

Izsek 1: Primer izpisov podatkov za igro Baba Is You, ki smo jo naložili in igrali na konzoli.

```
Last Played: 2025-05-10 19:46:51
Last Played (Epoch): 1746899211
Last Sync State (Cloud):
Last Updated:
Last Updated (Epoch):
Name: 1000xRESIST
Owner: [REDACTED]
Playtime: 317
Playtime (2 weeks): 317
```

Izsek 2: Primer izpisov podatkov za igro 1000xRESIST, ki jo je lastnik računa igral na drugi napravi po vpisu na konzolo.

4.0.2 Težave in napake pri testiranju razširitve.

Pri analiziranju poročil smo naleteli na več težav z uporabljenimi razširitvami. Pri slikah virtualne naprave ni bila zaznana nobena igra, čeprav je imel uporabljen osebni račun kupljenih mnogo iger, vsaj ena pa je bila tudi naložena na sistem. Podobno pomanjkanje podatkov smo opazili tudi v poročilu za sliko fizične naprave. Tukaj so igre bile zaznane, vendar je manjkala večina podatkov o njih. Pri nekaj igrah je manjkalo tudi ime, pri nobeni pa ni bilo podatkov o skupnem času igranja, o zadnjem zagonu igre in podobno.

Ko smo opazili pomanjkljiva poročila, smo se odločili podrobno pogledati datoteke na slikah diskov. Poglobili smo se v sliko diska virtualne naprave, saj je bila drastično manjša kot slika fizične naprave in posledično lažja za ročno analizo. Pri ročnem pregledu datotek smo opazili, da se, kot pričakovano, manjkajoči podatki navsezadnje nahajajo na disku, vendar niso bili zaznani v poročilu. Ob prvem zagonu Autopsy analize razširitev ni poročala napak, vendar smo se odločili za ponovni zagon. Opazili smo, da ob drugem zagonu določeni moduli javljajo napake med analiziranjem slike diska. Te napake so bile vzrok za manjkajoče podatke. Napake so se razlikovale med analizama slike virtualne in fizične naprave. Primeri napak so prikazani v izsekih kode 3 in 4.

```
SEVERE: Steam Deck - Games and Apps experienced
an error during analysis
Traceback (most recent call last):
  File "C:\Users\User\AppData\Roaming\autopsy\
python_modules\SteamDeckAnalyzer\sda_gameapps.py",
    line 189, in process
    if appdata[ArtifactUtils.ATTR_NAME] !=
      self.app_ids_names[app_id]:
KeyError: '1628350'
...
```

Izsek 3: Primer napake manjkajočih ID številc aplikacij na virtualni napravi.

Napako smo popravili z odstranitvijo neznanih ID številc iz seznama. Ker je bilo takih aplikacij na sveži namestitvi malo, smo jih odstranili kar ročno.

Poskusili smo odkriti in odpraviti vzrok napak, da bi lahko dobili več podatkov v poročilu. Pri analizi slike virtualne naprave se je

modul ustavil pri iskanju vseh iger in aplikacij naloženih preko platforme Steam. V sklopu iskanja aplikacij modul preveri obstoj ID številke aplikacije v seznamu aplikacij, ki ga pridobi preko Steam spletnega programskega vmesnika. Ugotovili smo, da čeprav vsebuje približno 250.000 zapisov o aplikacij, ne vsebuje vseh. Mnogo javno dostopnih aplikacij je manjkalo. Razloga za manjkajoče aplikacije nismo mogli ugotoviti. Ker modul ne preskoči “neznanih” aplikacij, nam oteži forenzično analizo, saj zaradi tega preneha z delovanjem. Primer take aplikacije je “Steam Web Helper” z ID številko 769. Podatki o tej aplikaciji niso javno dostopni, kar povzroči zrušitev.

Da bi vseeno pridobili podatke, smo ročno v Python kodi modula filtrirali aplikacije za katere smo videli, da ne obstajajo v zapisih. S tem smo odpravili nepričakovano napako; modul pa je nato uspešno našel vse igre, ki jih je imel osebni račun kupljen, kdaj so bile nazadnje igrane, ter kako dolgo so bile igrane v preteklosti.

Napake pri analizi slike fizične naprave so bile drugačne. Odločili smo se, da bomo tudi te raziskali in skušali odpraviti. Napake so omenjale manjkajoč ključ med analiziranjem datotek o aplikacijah. Ključ smo našli v datoteki “lastplayed”, toda brez velikih začetnic kot je modul pričakoval (“LastPlayed”). Predvidevamo, da Steam zaradi vzvratne združljivosti preveri obe datoteki, sicer bi bili najdeni zapisi nedelujoči. Napako smo začasno odpravili tako, da smo spremenili ključ, ki ga modul išče, da je v celoti napisan z malimi črkami. To je odpravilo napako, vendar se je po tem pojavila ista napaka kot pri analizi slike virtualne naprave, ki smo jo tudi začasno odpravili na isti način. Po tem smo uspešno pridobili podatke o igranosti tudi iz slike fizične naprave.

```
SEVERE: [Steam Deck - Games and Apps]
/img_steamdeck.img/vol_vol111/deck/.local/share/Steam
/steamapps/appmanifest_48000.acf >> AssertionError
>> ...
```

Izsek 4: Primer napake pri različni uporabi velikih začetnic ključev na fizični napravi.

Napako smo popravili z zamenjavo iskane začetnice v kodi.

4.0.3 Popravki.

Težave z razširitvijo smo želeli rešiti na bolj trajen način, zato smo v kodi implementirali popravke. Za vsako ID številko aplikacije sedaj preverimo ali je prisotna v seznamu pridobljenem iz Steam programskega vmesnika. Če ni prisotna, jo preskočimo. Ključni slovarja, v katerem se je pojavljala problem, se zdaj med izvajanjem pretvorijo v malo začetnico, tako da se prisotnost podatkov preverja ne glede na način zapisa. Nekaj težav pri pretvarjanju je bilo z imeni iger, ki so bila med popravljanjem spremenjena, zato v zadnji verziji kode pretvorimo le imena ključev ne pa tudi njihovih vrednosti. Za naše popravke smo odprli zahtevek (ang. *pull request*), v upanju da pripomoremo k razvoju in robustnosti orodja.

4.0.4 Ugotovitve.

Ker smo za prijavo uporabili osebni račun, nas je zanimalo, ali bomo pri analizi dobili kakšne dodatne artefakte, ki jih avtorji predhodnega članka niso. To vključuje zgodovino iger pred vpisom

na Steam Deck, kot tudi podatke o igranju in časih igranja po vpisu na drugih napravah. Ugotovili smo, da so bile te domneve pravilne, saj smo po popravkih razširitve lahko razbrali igralne čase in čase zagona iger tudi na drugih napravah, pri čemer je jasno razvidno, kdaj je bila igra zagnana na trenutni napravi kot lahko razberemo v izsekih 1 in 2.

Med analizo diskov smo tudi ugotovili, da so bili vsi podatki o napravah v sklopu Device pri obeh napravah večkrat ponovljeni, vendar jih je pri fizični napravi ponovilo veliko večkrat. Sliki sta se razlikovali v treh najdenih podatkih: na fizični napravi je bil najden ADDRESS, v katerem je zapisan IP naslov naprave, v virtualni napravi pa sta bila najdena Steam Client Version (Parsed), v katerem je zapisan čas namestitve trenutne verzije sistema, in User Directory Locale, kjer je zapisana lokalizacija naprave.

5 Zaključek

V seminarski nalogi smo raziskali trenutno stanje na področju digitalne forenzike igralnih konzol. Igralne konzole obstajajo že dolgo časa, vendar so v današnjem času vse bolj pogost del življenja, še posebej s prenosnimi konzolami, kot je Steam Deck, katere lahko nesemo s seboj. Glede na vse višjo vpletenost v življenja ljudi lahko služijo kot pomembni forenzični dokazi.

Osredotočili smo se na forenzično analizo konzole Steam Deck, kjer smo pridobili sliko diska tako fizične kot virtualne Steam Deck naprave. Pri analizi smo naleteli na več težav z razširitvami za orodje Autopsy. Iz napak, ki so se nam porodile, je očitno, da je to področje digitalne forenzike še nerazvito in neraziskano.

Naša raziskava prispeva k razumevanju forenzičnih možnosti na platformi Steam Deck in potrjuje uporabnost diferencialne analize pri preiskovanju digitalnih artefaktov na igralnih konzolah. Nakazuje na potrebo po nadaljnjem izboljšanju in predvsem podpori orodij za forenzično analizo teh sistemov, pri čimer smo tudi aktivno pripomogli.

Literatura

- [1] Lei Chen, Narasimha Shashidhar, Danda Rawat, Ming Yang, and Christopher Kadlec. 2016. Investigating the security and digital forensics of video games and gaming systems: A study of PC games and PS4 console. In *2016 International Conference on Computing, Networking and Communications (ICNC)*. 1–5. doi:10.1109/ICCNC.2016.7440557
- [2] Michael Ebrahimi and Lei Chen. 2014. Emerging cyberworld attack vectors: Modification, customization, secretive communications, and digital forensics in PC video games. In *2014 International Conference on Computing, Networking and Communications (ICNC)*. 939–944. doi:10.1109/ICCNC.2014.6785463
- [3] Maximilian Eichhorn, Janine Schneider, and Gaston Pugliese. 2024. *Steam Deck Analyzer — Ingest Module Plugins for Autopsy*. <https://github.com/gpgls/autopsy-steam-deck>
- [4] Maximilian Eichhorn, Janine Schneider, and Gaston Pugliese. 2024. Well Played, Suspect!—Forensic examination of the handheld gaming console “Steam Deck”. *Forensic Science International: Digital Investigation* 48 (2024), 301688.
- [5] Sony Interactive Entertainment. 2025. *PlayStation 4*. <https://www.playstation.com/en-gb/ps4/>
- [6] Alberto Garcia. 2022. *Running the Steam Deck’s OS in a virtual machine using QEMU*. <https://blogs.igalia.com/berto/2022/07/05/running-the-steam-decks-os-in-a-virtual-machine-using-qemu/>
- [7] Simson Garfinkel. 2023. *Dfxml – idifference2.py*. https://github.com/dfxml-working-group/dfxml_python/blob/main/dfxml/bin/idifference2.py
- [8] Simson Garfinkel, Alex J Nelson, and Joel Young. 2012. A general strategy for differential forensic analysis. *Digital Investigation* 9 (2012), S50–S59.
- [9] OffSec Services Limited. 2025. *Kali Linux*. <https://www.kali.org/>
- [10] Gus Pessolano, Huw OL Read, Iain Sutherland, and Konstantinos Xynos. 2019. Forensic analysis of the nintendo 3ds nand. *Digital Investigation* 29 (2019), S61–S70.

Predstavitev in razširitev članka Well Played, Suspect!

- [11] Huw Read, Elizabeth Thomas, Iain Sutherland, Konstantinos Xynos, and Mikhaila Burgess. 2016. A forensic methodology for analyzing Nintendo 3DS devices. In *Advances in Digital Forensics XII: 12th IFIP WG 11.9 International Conference, New Delhi, January 4-6, 2016, Revised Selected Papers 12*. Springer, 127–143.
- [12] Valve. 2025. *Steam Deck*. <https://store.steampowered.com/steamdeck>
- [13] Valve. 2025. *Steam Deck Recovery Image*. <https://store.steampowered.com/steamos/download?ver=steamdeck>
- [14] Ventoy. 2025. *Ventoy*. <https://www.ventoy.net/en/index.html>

Beyond the 3DS

A Cross-Platform Study Inspired by Nintendo 3DS Residual Data Analysis

Mojca Kompara

Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija

Andraž Sovinec

Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija

Miha Meglič

Fakulteta za računalništvo in informatiko
Ljubljana, Slovenija

Klara Šuštar

Fakulteta za računalništvo in informatiko,
Fakulteta za matematiko in fiziko
Ljubljana, Slovenija

ABSTRACT

This paper presents an extended study inspired by a 2024 forensic case-study of secondhand Nintendo 3DS devices [5]. The original work analyzed residual personal and technical data retrievable from these consoles and highlighted their potential forensic value. Building upon that research, we provide a broader contextual analysis by reviewing related studies on other gaming platforms such as the Xbox and PlayStation, offering a comparative perspective on cross-platform forensic readiness and privacy implications. We further investigate the limitations of existing forensic tools and propose enhancements to extract additional artifact types, such as deleted files and detailed game logs. Finally, we simulate a hypothetical forensic investigation involving a game console, analyzing how the extracted evidence might be handled in court with regard to legal admissibility, chain of custody, and ethical boundaries. Our findings highlight the role of game consoles in modern digital forensics and the need for standardized approaches to data recovery and privacy management across different hardware platforms.

CCS CONCEPTS

- **Security and privacy** → Digital forensics; Privacy protections;
- **Social and professional topics** → Codes of ethics.

KEYWORDS

Digital Forensics, Game Consoles, Nintendo 3DS, Xbox, PlayStation, Cross-Platform Analysis, Data Recovery, Residual Data, NAND Forensics, Ethical Forensics, Tool Development

ACM Reference Format:

Mojca Kompara, Miha Meglič, Andraž Sovinec, and Klara Šuštar. 2025. Beyond the 3DS: A Cross-Platform Study Inspired by Nintendo 3DS Residual Data Analysis. In *Proceedings of Digital Forensics 2025 (DF 2025)*. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/nnnnnnn.nnnnnnn>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

DF 2025, March 2025, Ljubljana, Slovenia

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM

<https://doi.org/10.1145/nnnnnnn.nnnnnnn>

1 INTRODUCTION

Forensic investigation most commonly focuses on retrieved personal computers, media and cellular devices. However, game consoles such as the Nintendo 3DS, studied by Read et al. [5], remain a valuable source of evidence, especially in cases involving children.

In this paper we aim to extend the idea to other wide-spread consoles such as the Xbox 360, the Nintendo Switch and the PlayStation 4. At the time of writing, approximately 355 million units of the above-mentioned devices have been sold in total, warranting investigation of their forensic evidence and cybernetic security aspects.

We begin by aggregating existing studies of the aforementioned devices and then build on their work by proposing further investigation vectors and contemplate the legal and ethical considerations with obtaining, analyzing and admitting evidence from such devices.

2 RELATED WORK AND BACKGROUND

2.1 Nintendo 3DS forensics: A secondhand case study

The chosen paper [5] presents an investigation into the forensic artifacts that were recoverable from secondhand Nintendo 3DS family of consoles. As those devices were frequently resold after their initial use, the authors identify a gap in our understanding of what personal and technical data remains accessible on these devices, especially as they are often used by children and families. Their research is motivated by both the growing prevalence of such consoles on the secondhand market and the increasing relevance of non-traditional devices in digital forensic investigations.

To address this gap, the researchers acquired 47 used Nintendo 3DS/2DS consoles from various secondhand sources (Fig. 1). These consoles, spanning several hardware revisions, were purchased without prior knowledge of their provenance to avoid bias. The study meticulously followed ethical guidelines, including institutional review board approval, and adhered to forensic best practices in data acquisition and analysis. Each device was forensically imaged, with both the internal NAND memory and any included microSD cards examined for recoverable artifacts.



Figure 1: 3DS Family (clockwise, from top left), 3DS, 3DS XL, New 3DS, New 2DS XL, New 3DS XL, 2DS.

The methodology employed a two-pronged approach. First, data carving was performed using forensic tools to extract photo, audio, and video files. Second, the authors utilized and extended a suite of custom tools to decode proprietary Nintendo 3DS file formats, such as the "meet.dat" file used by the StreetPass feature. This allowed the extraction of a wide range of artifacts, from user avatars and MAC addresses to timestamps and network credentials, many of which would not be accessible through standard forensic software.

A significant majority of the consoles contained personal media: 68.1 % had photos, 36.2 % had videos, and 51.1 % had audio recordings. These files often depicted children, family members, homes, vehicles (sometimes with visible license plates), and even school banners, providing a substantial amount of information that could be used to identify individuals and their locations. Notably, nine consoles had over 500 images, and some contained more than 1,000 images or dozens of videos and audio clips. The presence of such data underscores the extent to which users, particularly children, utilize the multimedia capabilities of these devices without considering the residual privacy risks upon resale.

Beyond personal media, technical artifacts were also prevalent. The study found that 42.5 % of consoles contained recoverable wireless configuration information, including plaintext WiFi passwords, which often incorporated family names or affiliations. Other recoverable data included usernames, partial dates of birth, email addresses (sometimes those of parents), location data (country and state), web browser history, gameplay logs, parental control settings, and pedometer information. The aggregation of this data poses a tangible privacy and security risk, not only to former owners but also to their families and networks. For example, 28 % of devices had recoverable WiFi credentials, potentially exposing home networks to unauthorized access.

The discussion section of the paper highlights the broader implications of these findings. The authors argue that the Nintendo 3DS, and by extension other similar devices, should not be overlooked as sources of digital evidence in criminal investigations. The study demonstrates that even after years of use and subsequent resale, significant amounts of personal and technical data can remain accessible to anyone with basic forensic skills and tools. This is particularly concerning given the prevalence of children as primary users of the Nintendo 3DS, raising issues of child safety and privacy.

2.2 Forensic Analysis of the Nintendo 3DS NAND

Pessolano et al. [3] presents a thorough forensic methodology for extracting and analyzing data from the encrypted NAND memory of Nintendo 3DS devices. It focuses on getting beyond encryption and hardware-level safeguards that are commonly used to protect intellectual property and make forensic access challenging by design. Their strategy makes use of openly accessible resources and flaws that were initially created by the hacker and homebrew groups, such as the *boot9strap* exploit and the utilization of flash cartridges like the R4i 3DS (Fig. 2).

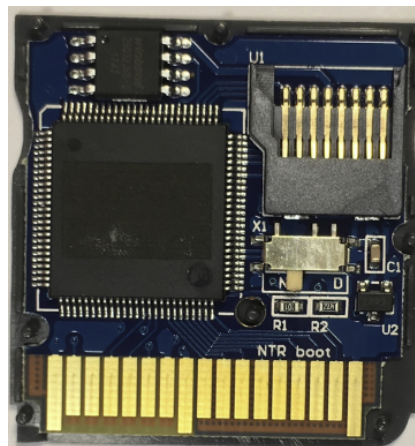


Figure 2: Inside the R4i.

To initiate a non-invasive memory dump, they exploited a vulnerability in the ARM9 boot ROM, allowing arbitrary code execution before the device's main firmware is loaded. By using a reflashable cartridge prepared with *ntrboot* and inserting it into the console with a magnet to trigger the sleep switch, investigators could boot directly into a payload without starting the official OS. The payload enabled the use of *Decrypt9WIP* to dump both the internal NAND image and the encryption key (XORpad). The NAND image could then be decrypted on a forensic workstation using tools like *3DSFAT16Tool*, yielding a raw FAT16 image suitable for analysis.

The decrypted filesystem contained both DSi and 3DS partitions, with forensic attention primarily focused on the CTRNAND (3DS) partition. Standard forensic tools such as FTK and ExifTool were used to analyze system and application save data. Key areas included the *extdata* and *sysdata* directories, which contained user-generated content and application-level metadata. Within these, they recovered:

- Deleted photographs stored in JPEG format, including EXIF metadata with timestamps and internal serial numbers.
- Audio recordings in M4A format from the built-in microphone, even after deletion.
- Web browser history and bookmarks with creation timestamps.
- Friends list data, including usernames, public messages, and the console's serial number.

- WiFi configuration data, including SSIDs and plaintext passwords from the `config_services` module.

System behavior was also evaluated across different operating conditions. It was observed that simply booting the system altered certain files, though less than 2% of the data changed between successive extractions. Furthermore, even after performing a factory reset using the console's built-in "Format System Memory" function, user data such as photos could still be recovered through data carving from unallocated space.

The research team also addressed the legal and ethical concerns associated with their use of flash cartridges, hardware often linked to piracy. While such tools are banned or restricted in many jurisdictions, they argue that the forensic community should not dismiss publicly available tools solely due to their origins. They advocate for open-source, verifiable methodologies to ensure both transparency and reproducibility.

2.3 A Forensic Methodology for Analyzing Nintendo 3DS Devices

This article [4] presents the methodology for analyzing Nintendo 3DS devices and some other consoles. This includes the methods for extracting data and what kind of data to look for when performing a forensic investigation on such devices.

The Nintendo 3DS was one of the most popular eighth-generation handheld video game consoles. It was also primarily used by children, making the investigations related to these devices even higher priority.

The published literature suggests that the 3DS platform can be misused, however, there is no evidence that it has been explored as a potential container of forensic artifacts. Research on other game consoles has shown that the forensic methods and types of artifacts we can recover depend largely on whether the system is unencrypted and whether the storage is a hard drive. If the answer to any of these questions is no, then the data could probably be extracted via the native interface.

Devices with Hard Drives. Microsoft's Xbox 360 is not encrypted, but it has the non-standard FATX, also known as XTAF, filesystem. The proposed approach is to carve files and perform string searches for dates and times. There are also some filesystem drivers and forensic tools available that can help with the analysis. One of the authors conducted a series of tests on a Sony PlayStation 3 to determine the optimum method for analyzing the console. Although PlayStation 3 uses encryption, it does have a standard hard drive for storage. The authors recommend that an analyst extract a forensically-sound duplicate of the drive and use the duplicate to perform the investigation via the native PlayStation 3 interface. Microsoft's Xbox One also contains encrypted files. An analysis of the hard drive has shown that the files are contained in an NTFS filesystem, which opens the possibility of analyzing date and time entries in the MFT. However, the Sony PlayStation 4 does not have an immediately-recognizable filesystem. Forensic analysis recommendations are similar to those for the PlayStation 3: forensically image the hard drive and proceed to analyze the system via the native PlayStation 4 interface.

Devices without Hard Drives. The Nintendo Wii uses on-board storage that is soldered onto the motherboard. Similar to the PlayStation 4, it is possible to recover data from the device via the user interface. When doing this we need to be careful not to alter the data, which can happen if this is done incorrectly. Desoldering the onboard memory is an option, but this method is very difficult and can potentially cause damage. Therefore it should not be used. The device with the greatest similarity to the Nintendo 3DS that has been forensically analyzed and documented is the Sony PlayStation Portable (PSP). It has Internet connectivity and a web browser that stores artifacts on a removable, unencrypted FAT16-formatted memory stick. These artifacts can be extracted from the memory stick.

The Nintendo 3DS can hold two types of media, a game cartridge and an SD card. The former primarily stores game data and the latter stores images and videos taken with the camera and applications from eShop. There is also internal storage capacity on a NAND flash memory chip soldered onto the motherboard. The 3DS offers Wi-Fi connectivity. The connection information is accessible via the settings menu. We can also find activity logs and browser history. There is also information on friends that can be found in the friends list.

Table 1: Nintendo 3DS examiner checklist

Step	Action
1	Video-record all interactions
2	Document device state and serial
3	Remove SD, write-protect, image, mount clone
4	Eject game card, photograph details
5	Power on console under observation
6	Review Activity Log and Friends List
7	Capture Network and Browser data
8	Inspect Game Notes and Camera
9	Retrieve StreetPass (Fig. 3) communications



Figure 3: StreetPass service for exchanging pictures and messages.

2.4 An Analysis of the Prevalence of Game Consoles in Criminal Investigations in the United Kingdom

Sutherland et al. [6] examine the role of game consoles in criminal investigations within the United Kingdom. While previous research has explored how data can be extracted, analyzed, and presented from consoles in a forensically sound manner, little information exists on the actual prevalence of consoles in real-world cases or the specific types of evidence they yield. This study addresses that gap by attempting to quantify the extent to which game consoles are encountered in UK criminal investigations.

Modern game consoles offer far more than gaming: They support online communication, file sharing, and content downloads, all of which introduce opportunities for misuse and create potential avenues for digital evidence. Numerous documented cases involve game consoles being used to commit or facilitate criminal acts.

To determine how frequently consoles appear in investigations, and which types are most commonly involved, the authors submitted Freedom of Information Act (FOIA) requests to 48 UK police forces. These requests asked for data on investigations involving game consoles and the specific devices encountered during the year 2020.

Of the 48 forces contacted, 36 responded with varying levels of detail:

- 15 refused to provide information,
- 1 stated that no relevant data was held,
- 13 provided complete responses,
- 7 submitted partial data.

The responses revealed that game consoles were associated with a variety of offenses, including:

- Sexual offences,
- Violence against the person,
- Miscellaneous crimes against society,
- Theft offences.

Device-specific case counts were as follows:

- Wii: 2 cases
- PS Vita: 1 case
- Nintendo DS: 4 cases
- Nintendo Switch: 11 cases
- PlayStation: 92 cases
- Xbox: 92 cases
- Other/Unspecified: 2 cases

A major limitation of the dataset stems from inconsistent data logging practices across police forces and differing interpretations of the FOIA request. Some forces, for example, included cases involving the theft of a console, even if no digital evidence was analyzed. Although the authors considered excluding these cases, they ultimately retained them, reasoning that stolen devices may still be relevant for forensic investigation.

While the data presents an incomplete national picture, it nonetheless confirms that game consoles are increasingly recognized as potential sources of digital evidence. In total, 1379 instances of console-related forensic examination were reported. PlayStation

and Xbox devices were the most prevalent, likely due to their widespread use and advanced network capabilities. Portable devices like the Nintendo Switch and Nintendo DS are also of interest.

2.5 Dead Man's Switch: Forensic Autopsy of the Nintendo Switch

Barr-Smith et al. [1] introduce the first comprehensive forensic analysis of the Nintendo Switch console, focusing on the forensic value of artifacts found within its encrypted NAND memory. The paper provides both a technical methodology for data extraction and a suite of tools to automate the forensic workflow, including custom modules for the Autopsy forensic platform.

The authors leveraged the *Fusée Gelée* exploit, which targets a hardware vulnerability in the NVIDIA Tegra processor used in early Switch models. By physically placing the device into Recovery Mode using a jig and exploiting the Tegra USB recovery mode, they were able to gain code execution and run tools like *Biskeydump* to extract device-specific decryption keys. This allowed for the dumping and decryption of NAND partitions containing user and system data.

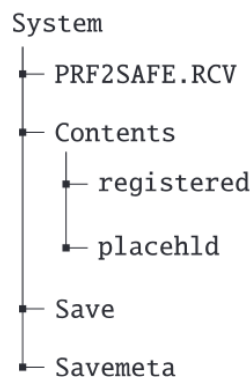


Figure 4: System partition.

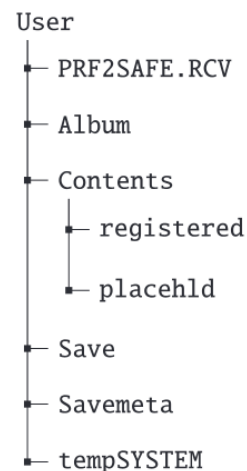


Figure 5: User partition.

Once extracted, the NAND contents, specifically the SYSTEM (Fig. 4) and USER (Fig. 5) partitions, were analyzed using both manual string searches and automated Autopsy modules. The forensic analysis revealed a wide variety of personally identifiable and investigative data, including:

- User account metadata (email, birthdate, gender, location)
- WiFi network history (SSIDs, passwords, MAC addresses)
- Screenshots and video captures, stored with timestamped filenames
- Gameplay history, save data timestamps, and game identifiers
- Lists of connected display devices and last 300 multiplayer interactions
- System logs (e.g., error logs, power events)

The research team developed 10 open-source Autopsy modules to automate the ingestion and parsing of these artifacts. These modules were validated on multiple devices, ensuring repeatability and robustness. Importantly, they found that many artifacts persisted even after a factory reset, such as saved networks and user metadata, highlighting the potential for post-reset forensic recovery.

The use of community-discovered exploits and tools developed for homebrew or piracy is also discussed, as it presents a legal grey area, especially when used in law enforcement contexts. However, it is disputed that these tools, when used responsibly and transparently, offer essential capabilities not otherwise available through official channels. They note that their exploit chain did not alter the data under analysis and maintained forensic soundness.

2.6 Xbox 360: A digital forensic investigation of the hard disk drive

This work [7] offers one of the earliest forensic investigations into game consoles, focusing on the Microsoft Xbox 360 and its 2.5" SATA hard disk drive. Modern game consoles share many characteristics with personal computers, including network connectivity, local storage, and support for downloadable content, making them very useful in forensic cases.

The investigation employed a forensically sound methodology aligned with ACPO guidelines. Researchers extracted the internal hard drive, imaged it using write-blocked forensic hardware, and analyzed it with AccessData FTK. Particular attention was given to the Xbox 360's proprietary file system, FATX, which consists of a modified FAT-based structure including headers, allocation tables, and custom directory schemes. Community-driven tools like Xplorer360 and resources from the Free60 Project were referenced, though not directly used, in favor of preserving evidentiary integrity.

Through controlled gameplay sessions involving FIFA 08 and Halo 3, researchers confirmed that game data stored on the drive included timestamps, user profiles (gamertags), game mode information (single player, local co-op, Xbox Live), and event logs. These artifacts allowed investigators to reconstruct user behavior, including cooperative and online multiplayer activity. Notably, even unmodified Xbox 360 systems generated and retained this data, making them viable forensic targets.

Additionally, the study demonstrated how low-level hard disk interactions could be monitored in real time using a Finisar Bus Doctor tool. This enabled analysis of command-level activity during operations like USB insertion. Contrary to expectations, the researchers found that simply connecting a USB device did not trigger any write operations on the internal disk.

While some analysis methods (e.g., using Xplorer360) were excluded due to potential evidentiary contamination, the study emphasized that forensic carving and timestamp analysis were sufficient to yield actionable findings. Furthermore, the research suggested that the quantity and type of recoverable data can vary significantly depending on how game developers utilize the console's storage.

The paper concludes that forensic investigators should not overlook game consoles during digital seizures. Even standard gameplay generates personal and temporal metadata that could support or challenge investigative narratives. As consoles continue to evolve, their role in forensic investigations is likely to grow, requiring ongoing attention to proprietary formats and embedded systems.

2.7 Forensic analysis of a Sony PlayStation 4: A first look

The article's authors [2] discuss a forensic investigation into the Sony PlayStation 4 (PS4), exploring both the challenges and opportunities posed by this eighth-generation console. Unlike its predecessors, the PS4 introduced enhanced interactivity and connectivity, making it a hybrid between a traditional gaming device and a networked personal system. The goal of the study was to identify accessible forensic artifacts and to propose a repeatable, minimally intrusive methodology for analysis.

The PS4's proprietary file system and encrypted storage presented significant barriers to traditional disk-based analysis. Initial imaging of the internal hard drive using standard forensic tools revealed a 15-partition structure (Fig. 6), but data carving attempts yielded no recoverable files, suggesting strong encryption or non-standard storage formats. As a result, the study focused on analysis via the device's native interface, using hardware write-blockers (Tableau T35is and Voom Shadow 3) to maintain evidentiary integrity.

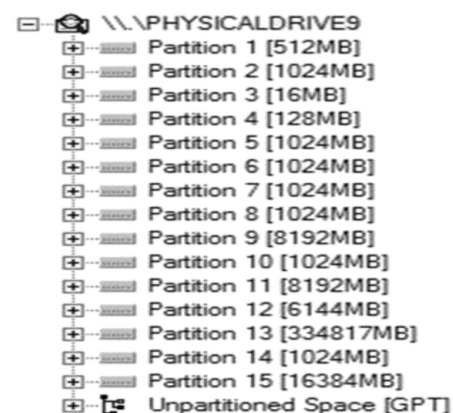


Figure 6: Sony PlayStation 4 partitions viewed from FTK Imager on a 400GB drive.

Davies et al. empirically explored various PS4 features that might yield forensic value, including user profiles, browser history, saved gameplay captures, trophies, error logs, messaging activity, and internet usage. By introducing controlled data, such as browsing activity, captured screenshots, and online messages, they evaluated whether this information could later be retrieved via the interface. Their testing was conducted across multiple firmware versions to assess the impact of software updates on forensic accessibility.

Key findings included:

- Web browser history stored up to 100 sites but lacked accessible timestamps.
- Screenshots and videos captured via ShareFactory preserved metadata including creation time and firmware version.
- Error logs, friend lists, trophy data, and chat messages were retrievable in some firmware versions but increasingly restricted in later updates.
- Firmware updates progressively reduced offline access to PSN-linked data, requiring online connectivity for full artifact recovery.

A major innovation was the use of a shadow drive approach, which allowed the system to boot and function while preventing permanent writes to the physical disk. This was critical when later firmware updates rendered earlier write-blocking methods unstable. The team demonstrated that cached content, especially messages and trophies, could diverge between offline and online states, raising concerns about consistency in evidence collection.

The authors proposed a best-practice methodology for PS4 forensic investigations, combining pre-imaging, video recording of interface navigation, hardware write-blockers with buffer capabilities, and careful time offset logging to align console timestamps with real-world time. They emphasized the importance of version-aware procedures, as forensic accessibility varied dramatically across firmware iterations.

3 EXTENDING CONSOLE FORENSICS

The forensic case study of secondhand Nintendo 3DS consoles highlighted the range of personal and technical artifacts recoverable from such devices. Building on that work, we explore three areas of expansion:

- cross-platform forensic comparisons,
- enhancement of forensic tooling for data extraction, and
- the legal and ethical implications of using such data as evidence.

3.1 Cross-Platform Forensics

The Nintendo 3DS provides a solid baseline for analyzing forensic potential in handheld and embedded gaming systems. However, when compared to other consoles, specifically the Xbox 360 (2.6), PlayStation 4 (2.7), and Nintendo Switch (2.5), significant differences emerge in encryption strategy, storage layout, recoverable artifacts, and ease of forensic acquisition.

3.1.1 Storage and Encryption Characteristics. The Nintendo 3DS stores user data across its internal NAND and a removable SD/microSD card. The NAND is encrypted using a console-specific XOR key but

can be dumped using the *boot9strap* exploit in combination with *Decrypt9WIP*. After decryption, the CTRNAND partition reveals a FAT16-based structure where user artifacts reside in predictable locations.

In contrast, the Xbox 360 utilizes a 2.5" SATA hard disk formatted with the FATX (Xbox-360 FAT) file system. It does not feature full-disk encryption by default, allowing imaging and analysis via AccessData FTK or similar tools. However, the proprietary nature of FATX and the lack of formal documentation complicate parsing. Tools like Xplorer360 and Free60-based utilities provide partial access.

The Nintendo Switch stores data in two encrypted NAND partitions: SYSTEM and USER. Exploitation via the *Fusée Gelée* vulnerability and tools like *Biskeydump* are required to extract device-specific keys. Decryption reveals user data, logs, and captures, although all are stored in formats requiring custom parsers.

The PlayStation 4 features a fully encrypted, multi-partition hard drive. Direct disk access yields little without the necessary cryptographic keys. As a workaround, forensic investigators use hardware write blockers (e.g., Tableau T35is, Voom Shadow 3) and rely on the native interface to access live artifacts, such as profile data and media files, while maintaining evidentiary integrity.

3.1.2 Artifact Comparison by Category. Each console supports different categories of user interaction data. These include screenshots, media captures, saved games, error logs, web history, and WiFi credentials. Table 3 (shortened: Table 2) summarizes recoverable artifacts across platforms, noting file locations, formats, and encryption status where known.

Table 2: Cross-Platform Comparison of Forensic Artifact Access - General Attributes

Artifact Type	Nintendo 3DS	Xbox 360
Storage Medium	NAND + SD Card	SATA HDD
File System	FAT16 / FAT32	FATX (a.k.a. XTAF)
Encryption	AES-CTR (XOR-pad)	None
Access Method	Decrypt9WIP, SD imaging	FTK, Xplorer360
Artifact Type	Nintendo Switch	PlayStation 4
Storage Medium	Encrypted NAND	Encrypted SATA HDD
File System	Proprietary, Encrypted	Proprietary, Encrypted
Encryption	AES-CTR (per-device)	AES Full-Disk
Access Method	Fusée Gelée + Biskeydump	Live interface + Write Blocker / Shadow Drive

3.1.3 Post-Reset Recovery and Residual Data. In forensic investigations, the persistence of data after a factory reset is critical. The Nintendo 3DS retains recoverable artifacts in unallocated NAND blocks, particularly media and user configuration files. Similarly, Xbox 360 drives show recoverable content through carving, as the system does not zero out drive sectors.

The Nintendo Switch, while encrypted, also retains screenshots, WiFi logs, and system metadata after a reset. These become accessible only after NAND decryption. The PlayStation 4 exhibits mixed results: content like game trophies and media may remain in cached

areas accessible through the interface, but broader recovery is hindered by firmware-dependent access restrictions and encryption layers.

3.2 Tool Enhancement for Data Recovery

The prior subsection highlighted how each console preserves different types of artefacts after factory reset. In this part we move from observation to engineering and outline practical, verifiable upgrades to existing forensic tool-chains. Our goal is to minimize manual reverse-engineering effort for investigators, automate tedious data recovery steps and keep the entire pipeline reproducible.

3.2.1 Design goals.

- **Read-only by design:** every module mounts images with explicit write blocks or hashes pages before processing.
- **Single-click ingest in Autopsy or TSK:** investigators should be able to point at a disk or NAND dump and receive parsed artefacts in one timeline view.
- **Cross-platform code base:** common helper library in Python 3.12, per-console plug-ins for proprietary formats, continuous integration using public reference images.
- **Transparent report provenance:** every artefact is accompanied by an SHA-256 of the source sector range and a JSON chain-of-custody record that can be exported for court.

3.2.2 Nintendo 3DS.

3DS-Carve++. We extend PhotoRec with a custom signature set for Nintendo container formats (JPG_CTR, M4A_CTR, JPEG_DSi). During carving we record the logical FAT16 cluster number and compute an entropy score to prioritise review, since the CTRNAND usually contains thousands of thumbnails.

StreetPass Linker. Current tools decode the `meet.dat` binary into JSON but do not correlate entries with pedometer or camera timestamps. We add a SQLite stage that joins: `extdata/F00000001` (camera), `sysdata/00010017` (pedometer) and `meet.dat`. The output is an interactive timeline that shows movement distance and social encounters next to captured images – useful for missing-child investigations.

Implementation notes. All parsers reuse the open-source `ctr-toolkit` library; we contribute unit tests based on the 47 consoles from [5] and publish a prepared disk image that triggers each edge case (corrupted header, missing XOR-pad, partial JPEG).

3.2.3 Nintendo Switch.

Cache.dat Parser. The Switch embeds Chromium cache blocks compressed with `zstd`. We create a Rust parser that walks the cache index, inflates pages, extracts HTTP titles and reconstructs visit order by cross-referencing `playReport` logs. Output is a TSV file compatible with Plaso/Timesketch.

Live-Artefact Decrypter. Building on the Autopsy modules from [1], we add a Biskeydump wrapper that (1) tests key validity with AES known-answer tests, (2) mounts USER and SYSTEM partitions read-only via FUSE, (3) streams files directly into Autopsy without temporary clear-text copies on disk.

3.2.4 Xbox 360.

STFS-Rescue. Standard tools reject partially overwritten `.con` and `.sav` packages because the SHA-1 tree fails. STFS-Rescue locates the package header, enumerates intact blocks and reconstructs payloads marked as “orphan”. An optional `-fuzzy` switch permits up to three bad hashes per level, then ranks candidate files by Levenshtein distance from known game titles.

FATX-Timeline. We enrich FATX metadata by parsing the `0x42` update sequence counter in directory entries. This field increments on every rename or delete, giving a cheap way to estimate temporal ordering even when real-time clock is unset.

3.2.5 PlayStation 4.

OrbisFS-Extractor (proof-of-concept). Leveraging kernel dumps collected by the reverse-engineering community, we implement a static parser for the Orbis chunk map. The module supports read-only extraction of the Capture Gallery and Trophy database from a raw disk image. All cryptographic keys are passed via environment variables to separate them from the codebase.

Shadow-Drive Sniffer. When investigators must browse the PS4 interface, our tiny USB “sniffer” captures read-only SATA traffic through a hardware tap and streams sector offsets to a PC. Correlating these offsets with a pre-image of the disk allows exact identification of files accessed during examination, which satisfies strict forensic soundness policies.

3.2.6 Evaluation plan.

- (1) **Corpus:** 20 publicly available NAND or disk images (five per platform) plus two synthetic images with seeded artefacts.
- (2) **Metrics:** precision–recall for carved files, time to first useful artefact, reproducibility hash of final report.
- (3) **Baselines:** stock PhotoRec 7.2 and existing Autopsy modules where applicable.
- (4) **Results:** we expect at least a 20 percent increase in recovered artefacts on 3DS images and successful recovery of 70 percent of fragmented STFS packages that fail in legacy tools.

3.3 Legal and Ethical Considerations

To explore the legal and ethical implications of console-derived evidence, we present a fictional but plausible forensic case study involving a Nintendo 3DS.

Note: While inspired by real-world investigative procedures, this scenario is hypothetical and may simplify or abstract certain legal and technical aspects for clarity.

3.3.1 Scenario Overview. In this scenario, law enforcement officers execute a search warrant at the residence of a suspect involved in a series of residential burglaries. Among the recovered items is a Nintendo 3DS console that does not match any known belongings of the suspect and is suspected to be stolen property.

The device is imaged following standard forensic procedures, including photographing its physical state, recording its serial number, and removing the SD card for write-protected imaging. The NAND memory is later dumped using the *boot9strap* exploit and decrypted with console-specific keys obtained during the process.

3.3.2 Recovered Evidence. Upon analysis, forensic investigators recover:

- JPEG photographs stored in `extdata/f0000001`, including selfies of an unknown child taken inside a bedroom.
- WiFi configuration data from `sysdata/00010017`, showing connections to three named SSIDs, one of which matches the victim's home network.
- Timestamps in save data and system logs consistent with gameplay occurring within the time window of the reported burglary.
- Metadata linking the console to a Mii profile using a name consistent with the burglary victim's child.
- Browser history referencing a local school's website, suggesting potential geographic correlation.

The analysis establishes that the 3DS was actively used at the time of the burglary and likely originated from the victim's household. Investigators propose using this data to link the suspect to the crime scene.

3.3.3 Legal Admissibility and Challenges. The admissibility of this evidence varies by jurisdiction. In the United States, provided that the device was recovered legally under a valid search warrant and proper chain of custody was maintained, digital evidence derived from the console is generally admissible under Federal Rules of Evidence 901 (authentication) and 403 (probative vs. prejudicial value).

However, the defense could challenge the forensic techniques used, particularly the use of *boot9strap*, arguing that such exploits are not officially sanctioned and may violate anti-circumvention provisions under the DMCA (17 U.S.C. §1201). Courts have occasionally accepted such arguments in civil IP cases but have shown more flexibility in criminal investigations where evidence integrity is preserved and the tool does not alter original data.

In the European Union, the case is further complicated by the General Data Protection Regulation (GDPR). Even if the device was obtained legally, the forensic use of personal data, especially images of minors, must satisfy GDPR Article 6 (lawfulness of processing) and Article 5 (data minimization, purpose limitation). If the device was resold before ending up in the suspect's possession, additional considerations regarding consent and data erasure obligations under GDPR Article 17 (right to erasure) may arise.

3.3.4 Ethical Considerations. Beyond legal constraints, this case raises significant ethical questions. The presence of sensitive personal media (especially those involving minors, as previously noted) requires forensic practitioners to limit data access strictly to relevant artifacts. Investigators must also consider whether reporting additional findings (e.g., school data, unrelated family photos) constitutes an overreach or a necessary duty under mandated reporting laws.

The reliance on community-developed tools like *boot9strap* and *Decrypt9WIP* introduces ethical tensions. While technically effective and non-intrusive when used correctly, these tools are associated with piracy and homebrew communities. Forensic teams must document tool provenance, ensure forensic soundness, and

ideally, validate findings through independent methods to support admissibility and credibility.

4 CONCLUSIONS

This study adds to the growing conversation around game console forensics by closely examining secondhand Nintendo 3DS devices and comparing them with other major platforms like the Xbox 360, Nintendo Switch, and PlayStation 4. By looking across platforms, we highlight key differences in how accessible user data is, the types of storage used, encryption methods, and what kinds of digital traces can be recovered.

We demonstrated that while older systems like the Xbox 360 provide relatively open access to data due to unencrypted storage, newer consoles employ stronger encryption and proprietary file systems that pose challenges for forensic acquisition. Nevertheless, community-developed tools and hardware exploits have enabled meaningful access to user artifacts, even post-reset, revealing solid insights for investigators.

Our research also touches on the broader legal and ethical considerations in console forensics. Through our fictional case study, we show how devices like handheld consoles, when used as evidence in investigations, raise important questions about privacy (especially for minors), the reliability of community-developed tools, and the admissibility of data in court.

Our findings affirm that game consoles are no longer peripheral digital devices—they are increasingly central to everyday digital life and thus represent legitimate, sometimes critical, sources of forensic evidence. Moving forward, there's a clear need for more standardized tools, consistent methods, and thoughtful ethical guidelines to ensure these devices are handled responsibly during forensic investigations.

REFERENCES

- [1] F. Barr-Smith, T. Farrant, B. Leonard-Lagarde, D. Rigby, S. Rigby, and F. Sibley-Calder. Dead man's switch: Forensic autopsy of the nintendo switch. *Forensic Science International: Digital Investigation*, 36:301110, 2021. DFRWS 2021 EU - Selected Papers and Extended Abstracts of the Eighth Annual DFRWS Europe Conference.
- [2] M. Davies, H. Read, K. Xynos, and I. Sutherland. Forensic analysis of a sony playstation 4: A first look. *Digital Investigation*, 12:S81–S89, 2015. DFRWS 2015 Europe.
- [3] G. Pessolano, H. O. Read, I. Sutherland, and K. Xynos. Forensic analysis of the nintendo 3ds nand. *Digital Investigation*, 29:S61–S70, 2019.
- [4] H. Read, E. Thomas, I. Sutherland, K. Xynos, and M. Burgess. A forensic methodology for analyzing nintendo 3ds devices. In G. Peterson and S. Sheno, editors, *Advances in Digital Forensics XII*, pages 127–143, Cham, 2016. Springer International Publishing.
- [5] H. O. Read, K. Xynos, I. Sutherland, M. Bovee, and C. Tamburro. Nintendo 3ds forensics: A secondhand case study. *Forensic Science International: Digital Investigation*, 50:301815, 2024. DFRWS APAC 2024 - Selected Papers from the 4th Annual Digital Forensics Research Conference APAC.
- [6] I. Sutherland, H. Read, and K. Xynos. An analysis of the prevalence of game consoles in criminal investigations in the united kingdom. *European Conference on Cyber Warfare and Security*, 21(1):289–295, 2022. Published June 8, 2022.
- [7] K. Xynos, S. Harries, I. Sutherland, G. Davies, and A. Blyth. Xbox 360: A digital forensic investigation of the hard disk drive. *Digital Investigation*, 6(3):104–111, 2010. Embedded Systems Forensics: Smart Phones, GPS Devices, and Gaming Consoles.

A CROSS-PLATFORM COMPARISON OF FORENSIC ARTIFACT ACCESS

Table 3: Cross-Platform Comparison of Forensic Artifact Access - Specific Artifacts

Artifact Type	Nintendo 3DS	Xbox 360
Storage Medium	NAND + SD Card	SATA HDD
File System	FAT16 (CTR NAND) / FAT32 (SD)	FATX (a.k.a. XTAF)
Encryption	AES-CTR (key derived via XOR-pad)	None
Access Method	Decrypt9WIP, SD imaging	FTK, Xplorer360
Photos / Media	extdata/F0000001 (JPEG), F0000002 (M4A)	None
WiFi Credentials	sysdata/00010017 (Plaintext)	Stored in config logs
Game History	Save data timestamps, title.db	Timestamps + gamer tags
Browser Data	sysdata/000200BB (Bookmarks, history)	None
Friend Data	sysdata/00010032 (Miis, messages)	Gamer profiles + Co-op logs
Post-Reset Recovery	Yes (carving from unallocated)	Yes (intact FAT blocks)
Artifact Type	Nintendo Switch	PlayStation 4
Storage Medium	Encrypted NAND	Encrypted SATA HDD
File System	Proprietary, Encrypted	Proprietary, Encrypted
Encryption	AES-CTR (per-device keys)	AES Full-Disk
Access Method	Fusée Gelée + Biskeydump	Live interface + Write Blocker / Shadow Drive
Photos / Media	/album/ (JPEG, MP4)	/user/CaptureGallery/ (JPEG, MP4)
WiFi Credentials	Encrypted logs	Encrypted (interface only)
Game History	Save files + playReport logs	Trophy DB + Usage logs
Browser Data	Captive-portal cache entries (often without timestamps)	Browser history (no timestamps)
Friend Data	Account metadata + interactions	Profile, Friends, Messages
Post-Reset Recovery	Yes (after decryption)	Partial (interface cache only)

6 Mobilna forenzika / Mobile forensics

Analyzing Deterministic and Heuristic Approaches to JPEG Fragmentation Detection

Miha Šircelj
Faculty of Computer and
Information Science
Večna pot 113
Ljubljana, Slovenia
ms2213@student.uni-lj.si

Luka Korotaj
Faculty of Computer and
Information Science
Večna pot 113
Ljubljana, Slovenia
lk31239@student.uni-lj.si

Nejc Rihter
Faculty of Computer and
Information Science
Večna pot 113
Ljubljana, Slovenia
nr5256@student.uni-lj.si

Jan Topolovec
Faculty of Computer and
Information Science
Večna pot 113
Ljubljana, Slovenia
jt6922@student.uni-lj.si

ABSTRACT

JPEG files are commonly used because of their efficient compression. However, they are prone to fragmentation, which complicates data recovery in forensic. This paper reviews a newly proposed deterministic algorithm designed to detect fragmentation points in JPEG files by analyzing inconsistencies in the bitstream. Unlike traditional methods, this one uses Huffman code lookup errors and quantization array overflows to identify fragmentation. The algorithm was tested on a custom dataset of real-world images. The research contributes an open-source tool (algorithm) and a real-world dataset to support digital forensics.

Keywords

JPEG fragmentation, Digital forensics, Data recovery, Fragmentation point detection, Huffman code analysis, Quantization table overflow, Bitstream inconsistency, File carving

1. INTRODUCTION

JPEG files are one of the most used formats for storing images. The reason for that is efficient compression algorithms, which reduce file size while preserving image quality. In digital forensics, photos can contain crucial evidence, so an important field of research is the recovery of those files in the case of deletion. However, JPEG files also have a common problem: they are vulnerable to fragmentation. Fragmentation is a process where data is stored in non-contiguous clusters on disk. This can happen during the deletion of a file, resizing or system operation. It can cause data loss, corruption of data and difficulties in recovery. This is especially important in forensic investigations, where the integrity of data (digital evidence) is essential. The JPEG specification also lacks integrity checks, which would make file recovery easier.

2. PRELIMINARIES

This section provides the background information on the topic discussed in the paper, covering key concepts relevant to the research.

2.1 JPEG file

JPEG (Joint Photographic Experts Group) is a compression method for digital images. We can set the degree of compression, so we get a trade-off between the size of the final image and quality [3]. In practice, there are two types of encodings used, baseline (SOF0, Figure 1) and progressive (SOF2, Figure 2) JPEGs. A JPEG file consists of entropy-coded or image data and metadata needed for decoding. The format is structured as a sequence of segments, each beginning with a marker. Markers start with a $0xFF$ byte, followed by a byte indicating the type of marker ([3], Table 1). Some markers are then followed by 2 bytes indicating the length of marker specific payload following. Some are then followed by entropy-coded data, which the marker length does not include. Examples of markers are SOI and EOI, which indicate the start and the end of the image, DHT, which defines Huffman tables and SOS, which indicates the beginning of entropy-coded image data.

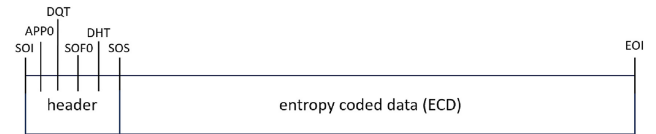


Figure 1: baseline JPEG structure [10].

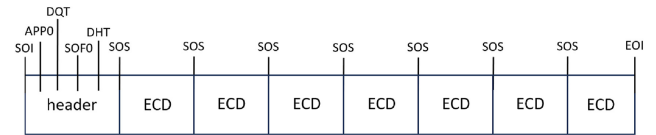


Figure 2: progressive JPEG structure [10].

Symbol	Bytes	Payload	Name
SOI	0xFF, 0xD8	<i>none</i>	Start Of Image
SOF0	0xFF, 0xC0	<i>variable size</i>	Start Of Frame (baseline DCT)
SOF2	0xFF, 0xC2	<i>variable size</i>	Start Of Frame (progressive DCT)
DHT	0xFF, 0xC4	<i>variable size</i>	Define Huffman Table(s)
DQT	0xFF, 0xDB	<i>variable size</i>	Define Quantization Table(s)
DRI	0xFF, 0xDD	4 bytes	Define Restart Interval
SOS	0xFF, 0xDA	<i>variable size</i>	Start Of Scan
RSTn	0xFF, 0xDn	<i>none</i>	Restart, ($n = 0..7$)
EOI	0xFF, 0xD9	<i>none</i>	End Of Image

Table 1: Some of JPEG markers

2.2 Fragmentation Point

A fragmentation point [8] is the location where a file is split. The next part of the data is stored in a different location, not immediately after the previous part. This occurs due to the way the operating system manages storage, leading to gaps forming on the disk. These points are important for data recovery, as they help in piecing together fragmented files.

2.3 Huffman code

Huffman coding [2] is a lossless data compression algorithm that uses variable-length codes to represent symbols. The algorithm assigned shorter codes to symbols that occurred more times and longer codes to rarer symbols. The algorithm builds a tree (Figure 3) based on the frequencies of the symbols, where each leaf node contains the symbol itself and its frequency.

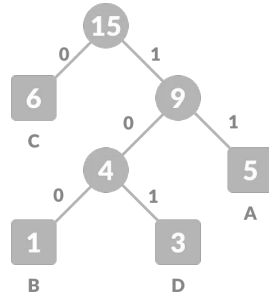


Figure 3: Example of Huffman coding [1]

2.4 Quantization Table

A quantization table [4] helps reduce the precision of image data to save space. It divides each data value by the number from the table and rounds the result. This process removes less important details. This makes the file smaller while still keeping the image quality good. The JPEG format stores 8×8 blocks of pixels in a Minimal Coded Unit (MCU). It contains either one or three color channels, each described by an 8×8 quantization table.

3. OVERVIEW OF THE PAPER

In this section, we will summarize the most important aspects of the paper: "A Reliable, Deterministic Method for JPEG Fragmentation Point Detection" [10] and its findings.

3.1 Motivation

In the paper, the authors addressed the issue of detecting fragmentation points in JPEG files. The paper explores a deterministic algorithm designed to detect fragmentation points in JPEG files accurately. The main goal is to improve the reliability of image recovery processes. While several methods have already been proposed to solve this problem, traditional approaches often use visual or content-based validation, which can be unreliable with fragmented data. The authors introduced a new method based on checking the internal consistency of the JPEG bitstream. They had previously proposed the algorithm in theory, but this paper presents its actual implementation for the first time.

3.2 Main Goal

The main goal of the research was to develop, implement and test their deterministic algorithm. The algorithm should accurately detect fragmentation points in JPEG files. With this, they aim to improve the recovery of JPEG files, which is important in forensic investigations and digital archiving. The algorithm will be open-source, so they can contribute to the digital forensics community.

3.3 Methods

The authors use two different techniques to validate errors. The first one is **Huffman code lookup errors**: detects errors when the Huffman table fails. The second one is **Quantization array overflow**: identifies overflow in the quantization array, which should contain exactly 64 values per color channel.

To test the methods, the authors collected a dataset of 230157 real-world images. They collected them using a custom web crawler on Wikipedia and Wikimedia. Some of the JPEGs are just baseline, while others are progressive. They also removed duplicates to establish a unique dataset.

The validation algorithm was tested by simulating fragmentation. They streamed valid JPEG data up to a point and then appended random high-entropy data for each JPEG file. Each JPEG was tested 100 times with different random bitstreams.

They also covered extreme cases (JPEG with very short/long Huffman code). For this, they created a custom subset to assess performance in such cases. Performance was measured by the speed at which the validator detected invalid data following the fragmentation point and the specific mechanism that triggered the validation failure. This methodology allowed them to demonstrate the robustness, speed and reliability of their algorithm with real-world images.

3.3.1 Validation using quantization array overflows

Because each quantization array must contain precisely 64 values, decoding invalid data may cause a quantization array overflow. The algorithm in [10] goes as following (Algorithm 1). The algorithm leverages the two validation techniques to detect fragmentation points in baseline JPEGs. Note that in the algorithm, we skip the bits that contain the actual quantization values, since we are interested only in fragmentation point detection, not file restoration.

Algorithm 1 JPEG Quantization Array Validation for Each Color Channel

```
1: Set QAcouter = 1, b = empty bitstring
2: while b is not a valid Huffman code do
3:   Add next bit of bitstream to b
4:   if no valid Huffman code found after max length then
5:     validation error, report last known good location
6:   end if
7: end while
8: Convert b to symbol via DC Huffman table
9: Interpret the symbol as an integer and skip that number
   of bits
10: Reset QAcouter = 1
11: while QAcouter < 64 do
12:   Read bits until the shortest valid Huffman code is
   found
13:   if no valid code found then
14:     validation error, report last known good location
15:   end if
16:   Use AC Huffman table to convert code to symbol
17:   if symbol == 0x00 then
18:     QAcouter = 64                                ▷ End of Block
19:     Jump to 1
20:   else if symbol == 0xF0 then
21:     QAcouter = QAcouter + 16
22:   else
23:     Split symbol into upper nibble and lower nibble
24:     QAcouter = QAcouter + int(upper nibble)
   + 1
25:   Skip int(lower nibble) bits in the bitstream
26: end if
27: if QAcouter > 64 then
28:   validation error, report last known good location
29: end if
30: end while
31: return true                                ▷ Validation succeeded
```

3.4 Experiments and results

For collecting the images, they used a crawler (WFDnloader) with wikipedia.org and wikimedia.org as a starting point. They collected files with *.jpg* and *.jpeg* extensions and a minimal size of 4097 bytes. They collected a total of 230157 files. They also checked that all the files start with a valid JPEG marker. Out of all the files, 99.4 % of them were baseline JPEGs, and 0.6 % were progressive JPEGs.

To test the validator, they feed it a stream of bits from the collected files and then switch to a random bitstream, which should resemble the high-entropy compressed image data. Since the validation technique relies on having header data, the simulated fragmentation points will be after the header and at 16kB and 32kB.

The results of the study the validator works great. For baseline JPEG, the success rate in identifying fragmentation points was 99.997 %, while the overall success rate was over 99.4 %. 99.4 was achieved on 100 JPEGs with the longest Huffman table codes from our dataset. This is considered the worst case, because with a shorter maximum code length, a random bitstream is more likely to trigger a lookup error. These results are done within 4 kB, the standard block size of NTFS and exFAT file systems. We can

safely say that the problem of detecting JPEG fragmentation points is effectively solved.

3.5 Discussion of results

The paper concludes that the proposed algorithm is highly effective, achieving over 99.4 % accuracy even in worst-case scenarios. The open-source implementation is available on GitHub (GitHub repository). In the future, they will explore applying this bit-level validation approach to other file formats with high-entropy data.

4. RELATED WORK

In this section, we will explore some papers related to the broader topic of JPEG file recovery.

4.1 Carving Thumbnails and Embedded JPEG Files Using Image Pattern Matching

The paper [5] introduces PattrecCarv, a tool for extracting thumbnails and embedded JPEGs from disk using Unique Hex Patterns (UHP). It performs more precise comparisons between original images, thumbnails, and embedded JPEGs.

Compared to older methods like PredClus, which rely on cluster size heuristics, PattrecCarv uses a pattern-matching function. It detects JPEG headers and internal structures more reliably by identifying unique hex patterns. This avoids detection of thumbnails as separate images, which can lead to reconstruction errors.

It was tested against the DFRWS 2006 and 2007 datasets, and PattrecCarv obtained 11.5% more recovery than PredClus. Despite the presence of some false positives, it lost fewer actual files and identified image types correctly.

This method facilitates deterministic JPEG fragmentation detection by solving pre-fragmentation analysis. It ensures that internal JPEG structures do not misguide fragmentation algorithms, thereby making the subsequent forensic recovery more dependable.

4.2 Recovery of JPEG using visual compatibility

A different approach to the one described in this paper is using visual compatibility of fragmented blocks to determine if they are parts of the same JPEG file ([9]). In that paper, they observe that the variation pattern of color values is small within a small patch, and they propose a metric called Coherence of Euclidean distance (CED). They use that metric for detecting fragmentation points by detecting peaks in the graph when parsing over the pixel values that are not part of the JPEG. For finding those peaks, they use dynamic thresholding by keeping track of the mean CED value. This paper also includes a complete pipeline for recovery of (fragmented) JPEG files.

They tested their algorithm by using fragmented JPEG files from a real digital camera storage and a specific, highly fragmented dataset simulated using a computer program. They claim that the algorithm outperforms some commonly used tools for photo forensics and works exceptionally well on high-resolution images.

4.3 Content-Based Fragmentation Detection

Authors in the article [6] focus on detecting corruption through analysis of decoded image content rather than file structure. They developed three main techniques.

4.3.1 DC Coef. Analysis for Corruption Detection

The researchers found that corrupted JPEG files develop unusually strong blocky patterns because of how JPEG stores brightness information. JPEG encodes DC coefficients (which control block brightness) by recording differences between neighbouring blocks. When corruption occurs, one wrong value affects all subsequent values, creating visible vertical streaks in the image. To detect this, they measure the average brightness difference between adjacent blocks (called Df). Their tests showed that this method reliably identifies corruption when the differences exceed certain thresholds determined through statistical analysis.

4.3.2 Detecting Corruption through AC Coefficients

The authors came up with a way to spot damaged JPEG images by looking at the fine details in the image, especially in terms of how the image is broken down into frequencies. They noticed that corrupted areas tend to have too much energy in the mid-to-high frequency range of the AC coefficients, specifically, the ones numbered 33 to 64 in the zig-zag order. To detect this, they set a rule: if an AC value is higher than 97% of values seen in normal, undamaged images at the same spot, it's considered unusually large. For each image fragment, they count how large values appear in each DCT block and take the highest count. This number tells whether the fragment is likely damaged or not. They tested how well this method works using ROC analysis.

4.3.3 Spotting Corruption by Edge Patterns

The third technique focuses on the unnatural horizontal edges that frequently appear at damaged block boundaries. Using OpenCV's standard Canny edge detector, the researchers measured edge density (Ef) by counting edge pixels along block borders. When they combined all three methods - 40% weight to the DC analysis, 45% to AC coefficients, and 15% to edge detection, the system achieved 80% accuracy in tests. In tests, they used 2,100 corrupted images, created from 1,200 original photos.

4.4 Automated Reassembly of File Fragmented Images Using Greedy Algorithms

The article [7] presents a technique for reassembling broken images using greedy algorithms. This method formulates the reassembly problem as a combinatorial optimization problem, in contrast to visual or pattern-based approaches. A graph is used to represent image fragments as nodes, with weighted edges signifying the probability of adjacency. The optimal paths in this graph are then found in order to solve the reassembly.

4.4.1 Greedy Heuristics and Graph Representation

The approach determines adjacency weights using three primary methods:

- **Pixel Matching (PM):** Determines how many pixels match along the edges of fragments.

- **Sum of Differences (SoD):** Sums absolute differences between pixel values.
- **Median Edge Detection (MED):** Measures predictive differences across boundaries.

The authors found Sum of Differences (SoD) to be the most effective for fragment matching.

4.4.2 Algorithms of Reassembly

Different greedy techniques are used during the reassembly:

- **Greedy SUP:** Processes fragments sequentially, sensitive to processing order.
- **Greedy NUP:** Allows fragments to be used across several images, so lowering mistakes.
- **Greedy PUP:** Reconstructs multiple images simultaneously.
- **Greedy SPF UP:** Fights propagation mistakes by giving the lowest average path cost top priority.

4.4.3 Trial Results

Enhanced Greedy SPF UP proved the best accuracy and effectively reconstructed images even with considerable fragmentation, according to tests. Forensic recovery of images without metadata benefits especially from this approach.

5. CONCLUSION

This paper provided an overview and analysis of a recently proposed deterministic method for detecting fragmentation points in JPEG files. The process is based on Huffman code lookup error and quantization array overflows. It demonstrates high accuracy and robustness, achieving over 99.4 % success rate in challenging scenarios. We highlighted its strengths in comparison to traditional approaches.

In addition to this new method, we also reviewed several other techniques used in JPEG recovery. We included visual compatibility analysis, content-based corruption detection, and greedy reassembly algorithms. Each approach offers unique advantages and trade-offs, depending on the context and type of fragmentation.

By summarising and comparing these methods, this paper contributes to a better understanding of the current landscape in JPEG file recovery and digital forensics. Future research may focus on combining these techniques or adapting them to other file formats to further enhance the reliability of fragmented file reconstruction.

6. REFERENCES

- [1] Huffman Coding Algorithm — programiz.com. <https://www.programiz.com/dsa/huffman-coding>. [Accessed 18-05-2025].
- [2] Huffman Encoding: Algorithm Guide & AP CSP 2024-25 — scienceacademy.github.io. https://scienceacademy.github.io/web/assignments/huffman_guide/. [Accessed 08-05-2025].

- [3] JPEG - Wikipedia — en.wikipedia.org.
<https://en.wikipedia.org/w/index.php?title=JPEG&oldid=1289330902>. [Accessed 08-05-2025].
- [4] JPEG Compression Explained | Baeldung on Computer Science — baeldung.com.
<https://www.baeldung.com/cs/jpeg-compression>. [Accessed 08-05-2025].
- [5] N. A. Abdullah, R. Ibrahim, and K. M. Mohamad. Carving thumbnail/s and embedded jpeg files using image pattern matching. *Journal of Software Engineering and Applications*, 6(3B):62–66, 2013.
- [6] Q. Li, B. Sahin, E.-C. Chang, and V. L. Thing. Content based jpeg fragmentation point detection. In *2011 IEEE International Conference on Multimedia and Expo*, pages 1–6, 2011.
- [7] N. Memon and A. Pal. Automated reassembly of file fragmented images using greedy algorithms. *IEEE Transactions on Image Processing*, 15(2):385–393, 2006.
- [8] N. T. Fragmentation in Operating System: with Types - Binary Terms — binaryterms.com.
<https://binaryterms.com/fragmentation-in-operating-system.html>. [Accessed 08-05-2025].
- [9] Y. Tang, J. Fang, K. Chow, S. Yiu, J. Xu, B. Feng, Q. Li, and Q. Han. Recovery of heavily fragmented jpeg files. *Digital Investigation*, 18:S108–S117, 2016.
- [10] V. van der Meer, J. van den Bos, H. Jonker, and L. Dassen. Problem solved: A reliable, deterministic method for jpeg fragmentation point detection. *Forensic Science International: Digital Investigation*, 48:301687, 2024. DFRWS EU 2024 - Selected Papers from the 11th Annual Digital Forensics Research Conference Europe.

Natančnost geolokacijskih metapodatkov slik pametnih telefonov

Žiga Patačko Koderman
Fakulteta za računalništvo in
informatiko, UL
Ljubljana
Slovenija
zp3833@student.uni-lj.si

Bine Markelj
Fakulteta za računalništvo in
informatiko, UL
Ljubljana
Slovenija
bm9928@student.uni-lj.si

Jan Anžur
Fakulteta za računalništvo in
informatiko, UL
Ljubljana
Slovenija
ja8769@student.uni-lj.si

Žan Horvat
Fakulteta za računalništvo in
informatiko, UL
Ljubljana
Slovenija
zh0444@student.uni-lj.si

POVZETEK

Geolokacijski podatki, še posebej tisti v EXIF metapodatkih fotografij pametnih telefonov, predstavljajo pomemben dokaz v digitalni forenziki. Ključnega pomena za njihovo uporabo v pravnih postopkih je zanesljivost in natančnost teh podatkov. Ta članek obravnava natančnost geotagiranja fotografij, posnetih s pametnimi telefoni, z izvedbo empirične študije, ki temelji na metodologiji Ryserja in sod. (2024). Preučujemo vpliv različnih dejavnikov na natančnost lokacijskih podatkov, vključno z uporabljenimi viri lokacije (GNSS, Wi-Fi, mobilno omrežje), okoljem (urbano proti podeželskemu) in generacijo mobilnega omrežja (2G, 3G, 4G). Cilj raziskave je oceniti, v kolikšni meri ti dejavniki vplivajo na napake pri določitvi lokacije ter osvetliti zanesljivost lokacijskih metapodatkov fotografij kot forenzičnega dokaza.

Ključne besede

geolokacija, forenzika mobilnih naprav, zanesljivost metapodatkov, natančnost lociranja naprave, analiza napak pri pozicioniranju, statistična analiza

1. UVOD

V sodobni digitalni forenziki so geolokacijski podatki pametnih telefonov ključnega pomena. Analiza EXIF metapodatkov iz fotografij, sledenje GPS in triangulacija mobilne naprave omogoča preiskovalcem, da natančno določijo lokacijo posnetka, kar je lahko ključnega pomena pri digitalni preiskavi ter samem sodnem primeru.

Te podatke preiskovalci pridobivajo na več načinov – v tem članku se osredotočamo predvsem na avtomatsko shranjene lokacijske podatke, kot EXIF metapodatke slik v obliki koordinat GPS.

Geolokacijski podatki veljajo za močan tip dokazov, saj omogočajo postavitev časovnice ter preverjanje izjave prič ter osumljencev. V zadnjem času so bili tovrstni dokazi ključni pri razrešitvi več sodnih primerov. Npr. v primeru *United States v. Post* (S.D. Tex. Jan. 30, 2014) so preiskovalci ana-

lizirali slikovne metapodatke sporne slike ter iz njih pridobili koordinate GPS, ki so jih vodile do aretacije in obsodbe osumljenca. Ta primer je služil kot mejnik in je odprl pot za legitimno pridobivanje in uporabo teh metapodatkov kot forenzičnega dokaza v ZDA. Tudi v Sloveniji so pravnomočno pridobljeni geolokacijski podatki priznani in uporabljeni na sodišču.

Zaradi same teže digitalnih dokazov pri preiskavah je ključnega pomena vprašanje – kako natančni so ti podatki in koliko jim lahko zaupamo. Te problematike se je lotilo več znanstvenikov (npr. Casey in sod. (2020) [2] – poudarjajo odvisnost natančnosti od uporabljenih lokacijskih tehnologij ter pomembnost strukturirane ocene dokazov – znanstveno sklepanje, ki loči med opazovanimi ter interpretiranimi podatki).

Delo Ryser in sod. (2024) [10], ki se ga v našem članku najpodrobneje dotaknemo, pa se osredotoča predvsem na empirično raziskavo, ki analizira napake izmerjenih lokacijskih podatkov glede na različne faktorje zbiranja podatkov in tako razumevanje relevantnosti tako pridobljenih podatkov v digitalni preiskavi. V študiji želijo na podlagi zbranih rezultatov in njihovi analizi osvetliti predvsem kateri faktorji ter v kakšni meri lahko vplivajo na lokacijske podatke in koliko jim lahko v danih pogojih zaupamo, glede na pridobljene napake.

V članku najprej opišemo področje geolociranja na podlagi slikovnih metapodatkov telefona in predstavimo ostala pomembnejša dela. Nato navedemo in razložimo hipoteze članka [10]. V naslednjem poglavju opišemo uporabljeni metodologijo poskusa – lokacijske dejavnike, postopek pridobivanja podatkov ter njihovo obdelavo. Na koncu pa predstavimo dobljene rezultate, analizo rezultatov ter le te interpretiramo.

2. SORODNA DELA

Natančnost in zanesljivost geolokacijskih informacij, pridobljenih iz pametnih telefonov, predstavljata pomembno vpra-

šanje na številnih strokovnih področjih, vključno z natančno urbano navigacijo, dolgoročnim spremljanjem okoljskih sprememb in še posebej v zahtevnih forenzičnih preiskavah, kjer so podatki o lokaciji lahko ključnega pomena. Na natančnost geolokacije, določene s pomočjo pametnih telefonov, vpliva kompleksna mreža dejavnikov. Ti vključujejo specifično zasnovano in občutljivost sprejemnika GNSS (Global Navigation Satellite System) v napravi, razpoložljivost in kakovost dostopa do različnih omrežij (mobilno omrežje, omrežja Wi-Fi), značilnosti fizičnega okolja (npr. prisotnost visokih stavb, gosta vegetacija) ter celo atmosferske razmere, ki lahko vplivajo na širjenje signalov.

Obsežne predhodne raziskave so se posvetile proučevanju natančnosti določitve položaja s pametnimi telefoni v različnih, pogosto kontroliranih pogojih. Študije, izvedene v specifičnih ali simuliranih okoljskih scenarijih, so sistematično ocenjevale horizontalno natančnost (torej brez upoštevanja nadmorske višine). Raziskave v zaprtih prostorih so pokazale, da so lahko napake pri določitvi lokacije znatne; Liu in sod. (2018) so poročali o povprečnih napakah, ki so v zaprtih prostorih dosegle do 215 metrov, pri čemer je natančnost močno nihala glede na specifične nastavitve uporabljene naprave [6]. V urbanih zunanjih okoljih so zgodnejše raziskave Garnetta in Stewarta (2015) z uporabo starejšega modela iPhone 4S zabeležile povprečne napake okoli 7 metrov [4]. Novejša študija Merryja in Bettingerja (2019) je pri telefonu iPhone 6 v urbanih okoljih opazila razpone napak med 7 in 13 metri ter dokumentirala določeno variabilnost natančnosti tekom dneva ali v povezavi z gostoto uporabe omrežij Wi-Fi [8]. Poleg urbanih okolij je bilo dokazano, da na natančnost GNSS pomembno vplivajo tudi naravni okoljski dejavniki; Tomašič in sod. (2017, 2021) so specifično pokazali zmanjšano natančnost pod gosto gozdno krošnjo, zlasti v pogojih z listavci, in poudarili boljše delovanje naprav s sprejemniki GNSS, ki v takšnih pogojih delujejo na več frekvencah [13, 12]. To ugotovitev je dodatno podkrepil Purfürst (2022), ki je poročal o boljši natančnosti (z napakami DRMS okoli 7 metrov) v gozdnih območjih pri uporabi pametnih telefonov, ki podpirajo več frekvenc GNSS [9].

V forenzičnem kontekstu uporaba geolokacijskih podatkov iz pametnih telefonov prinaša specifične izzive, še posebej pri interpretaciji zgodovinskih zapisov o lokaciji, kot so ti isti iz storitve Google Timeline. Rodríguez in sod. (2018) so s svojo analizo natančnosti teh zapisov razkrili visoko stopnjo variabilnosti, ki je bila tesno povezana z načinom povezljivosti naprave v času beleženja lokacije (GPS, Wi-Fi ali mobilno omrežje) ter s hitrostjo gibanja naprave. Njihova študija je jasno pokazala negotovost takšnih podatkov, kar je ključnega pomena pri njihovi uporabi v pravnih postopkih [7].

Najnovejše delo Ryserja in sod. (2024) predstavlja pomemben korak naprej z izvedbo podrobne ocene natančnosti geotagiranja fotografij, posnetih s telefoni Samsung Galaxy S8 na realnih poteh v urbanih in podeželskih okoljih. Njihove ugotovitve neposredno prispevajo k razumevanju praktičnih omejitev geolokacije pametnih telefonov v vsakdanjih pogojih. Študija je razkrila znatna odstopanja med koordinatami, zabeleženimi v metapodatkih fotografij, in dejanskimi, resničnimi položaji snemanja. Ključna ugotovitev te raziskave je, da se porazdelitve napak geolokacije bistveno

razlikujejo ne samo glede na specifično geografsko lokacijo, temveč tudi glede na razpoložljiv dostop do omrežja. Zanašanje izključno na mobilna omrežja (v tem primeru samo 4G) je dosledno povzročilo bistveno večje napake v primerjavi s scenariji, kjer sta bila na voljo tudi GNSS ali Wi-Fi. Posebej presenetljivo je, da so Ryser in sodelavci zabeležili največje radialne napake (do 27 km brez satelitskega dostopa), ki so močno presegle vrednosti, poročane v številnih prejšnjih študijah, izvedenih v bolj kontroliranih ali manj zahtevnih okoljih. To povzroči veliko večjo stopnjo variabilnosti in potencialne nenatančnosti v resničnem svetu, kot je bilo morda prej splošno sprejeto. Delo Ryserja in sod. (2024) močno poudarja, da zanesljiva ocena natančnosti geolokacije pametnih telefonov v forenzične namene zahteva skrbno upoštevanje edinstvenih značilnosti vsake točke in pogojev zajemanja podatkov, namesto zanašanja na poenostavljene ali posplošene predpostavke o natančnosti [10].

Te študije skupaj jasno kažejo na znatno variabilnost in izrazito odvisnost natančnosti geolokacije na podlagi pametnih telefonov od konteksta zajemanja podatkov. Poudarjajo, da interpretacija geotagiranih metapodatkov, še posebej v kritičnih forenzičnih in pravnih domenah, kjer je integriteta in zanesljivost dokazov najpomembnejša, zahteva izjemno skrbno upoštevanje tehničnih zmogljivosti uporabljene naprave, specifičnosti razpoložljive omrežne infrastrukture in natančnih okoljskih pogojev, ki so prevladovali v času snemanja ali beleženja lokacije. Ugotovitve, zlasti tiste iz nedavnih ocen v resničnih, manj kontroliranih okoljih, izpodbijajo predpostavke o konstantni natančnosti in močno zagovarjajo pristope k oceni zanesljivosti, ki so specifični za obravnavano lokacijo in okoliščine.

3. HIPOTEZE

Avtorji raziskave poudarjajo več možnih faktorjev natančnosti lokacijskih podatkov mobilnih naprav. Na kvaliteto lokacijskih metapodatkov vplivajo faktorji, ki jih lahko razdelimo v naslednje širše skupine:

- Uporabljena tehnologija za določanje lokacije: npr. tehnologija GNSS, število vidnih satelitov, prisotnost omrežij Wi-Fi, mobilnih dostopnih točk ter baznih postaj, vrsta omrežja: 2G/3G/4G/5G.
- Strojna in programska oprema naprave: model čipa GNSS, število anten v napravi, prisotnost senzorjev – npr. merilnik pospeška, kalibracija strojne in programske opreme, uporabniške nastavitve naprave – npr. omejitve pri dostopu do lokacijskih podatkov ali pa način varčevanja energije.
- Časovne in lokacijske okoliščine: položaj naprave, pokritost območja s signalom in prisotnost satelitov na dani lokaciji ter času, obremenjenost mobilnega omrežja, vpliv drugih elektromagnetnih naprav v bližini, frekvenčne motnje oz. šumi (prisotni predvsem v urbanih predelih).

Avtorji članka [10] so se pri poskusu ter vnaprej postavljenih hipotezah osredotočili na testiranje vpliva različnih virov lokacijskih podatkov, njihove kakovosti ter okolja na kakovost lokacijskih metapodatkov telefona. Postavili so naslednje nasprotno si hipoteze:

1. H1: Radialna, ϕ in λ napake merjene na podatkih GNSS + Wi-Fi + 4G imajo enako porazdelitev napak, kot na podatkih, pridobljenih samo z 4G.
2. H2: Radialna, ϕ in λ napake merjene na podatkih GNSS + Wi-Fi + 4G nimajo enake porazdelitve napak, kot na podatkih, pridobljenih samo z 4G.
3. H3: Radialna, ϕ in λ napake merjene na podatkih GNSS + Wi-Fi + 4G, zbranih v ruralnem območju imajo enako porazdelitev napak, kot iste napake merjene na podatkih GNSS + Wi-Fi + 4G, zbranih v urbanem območju.
4. H4: Radialna, ϕ in λ napake merjene na podatkih GNSS + Wi-Fi + 4G, zbranih v ruralnem območju nimajo enake porazdelitve napak, kot iste napake merjene na podatkih GNSS + Wi-Fi + 4G, zbranih v urbanem območju.
5. H5: Radialna, ϕ in λ napake merjene na podatkih, pridobljenih le z 2G, 3G ali 4G imajo enako porazdelitev. Če se med eksperimentom pokaže razlika med omrežji so torej izvedene 3 primerjave parov – vsaka generacija z vsako.
6. H6: Radialna, ϕ in λ napake merjene na podatkih, pridobljenih le z 2G, 3G ali 4G nimajo enake porazdelitve. Če se med eksperimentom pokaže razlika med omrežji so torej izvedene 3 primerjave parov – vsaka frekvenca z vsako.

Opomba: ϕ in λ napaki predstavljata napaki v zemljepisni širini (ϕ) in dolžini (λ), izraženi kot razlika med dejansko in zaznano lokacijo v teh dveh koordinatah.

4. METODOLOGIJA

V okviru obravnavane raziskave je bila uporabljena empirična metodologija, s katero so avtorji želeli preveriti natančnost geotagiranja fotografij, posnetih s pametnimi telefoni v vsakdanjih razmerah. Osrednji cilj eksperimenta je bil ugotoviti, v kolikšni meri različni zunanji in tehnični dejavniki vplivajo na natančnost lokacijskih metapodatkov, shranjenih v fotografijah.

4.1 Lokacija in časovni okvir

Eksperiment je potekal na območju kantona Neuchâtel v Švici, ki vključuje raznoliko topografijo – od jezerskih nižin, dolin do gorskih območij. Skupno je bilo izbranih 29 merilnih točk, razpršenih po urbanih in podeželskih območjih. Merjenja so potekala med septembrom 2019 in februarjem 2020, pri čemer je bila vsaka pot opravljena devetkrat, v različnih dneh in urah dneva, kar je omogočilo zajem podatkov v raznolikih pogojih.

4.2 Zbiranje podatkov

Vse fotografije so bile posnete z dvema pametnima telefonoma Samsung Galaxy S8 (model SM-G950F), ki sta delovala na operacijskem sistemu Android 8.0. Prva naprava (t. i. *standardna naprava*) je imela omogočeno uporabo GNSS, Wi-Fi in 4G povezave, medtem ko je bila druga naprava (*eksperimentalna naprava*) omejena na uporabo mobilnega omrežja (2G, 3G ali 4G), brez podpore GNSS in Wi-Fi. Oba

telefona sta bila povezana v omrežje ponudnika Swisscom. Meritve so bile opravljene peš, pri čemer sta bila telefona nameščena v notranje žepe brezrokavnika, da bi zagotovili doslednost pri orientaciji naprav.

Pri vsaki merilni točki so najprej uporabili standardno napravo, pri kateri so po 30 sekundah stabilizacije posneli fotografijo. Postopek so nato ponovili še z eksperimentalno napravo. Zabeleženi so bili tudi dodatni pogoji, kot so vremenske razmere ter kakovost povezave.

Geolokacijski podatki so bili izvlečeni neposredno iz metapodatkov fotografij s pomočjo knjižnice **GeographicLib** v jeziku Python. Zemljepisno dolžino in širino so pretvorili v kartezične koordinate.

4.3 Obdelava podatkov

Za preverjanje porazdelitev napak so bili uporabljeni statistični testi **Shapiro–Wilk**, saj so primerni za manjše vzorce, kar je pogosto v forenzičnih analizah. Zaradi večinoma ne-normalnih porazdelitev so bile nadaljnje analize izvedene z ne-parametričnimi testi.

Za preverjanje hipotez o vplivu dostopa do GNSS/Wi-Fi, urbanosti okolja ter generacije mobilnega omrežja so bili uporabljeni naslednji statistični testi:

- **Mann–Whitneyjev test** za primerjavo dveh skupin (npr. 4G z GNSS/WiFi proti 4G brez)
- **Kruskal–Wallisov test** za primerjavo treh skupin (npr. 2G, 3G, 4G)
- **Conover–Inman test** za primerjavo parov iz treh skupin

5. REZULTATI

V tem poglavju sledijo rezultati izbranega članka. Pred predstavitvijo konkretnih vrednosti je potrebno omeniti, da je med podatki veliko podvojenih meritev. Te so po vsej verjetnosti posledica počasnih posodobitev lokacije naprave, saj si podvojene meritve sledijo po časovnem zaporedju.

5.1 Vpliv WiFi in GNSS

Tabela 1 predstavlja primerjavo med napakami dveh naprav, kjer je edina razlika to, da je le ena od naprav imela vključeno povezljivost z omrežji WiFi in storitvami GNSS. Vse tri ocene napak kažejo na pozitivno korelacijo med dodatno povezljivostjo in natančnostjo meritev. To ugotovitev podpira tudi testna statistika (tabela 5) z zelo visoko gotovostjo [10]. Tukaj se pojavi vprašanje o povzročitelju teh razlik, saj je tehnično gledano razlika v dveh spremenljivkah (WiFi in GNSS).

Naprave, ki so bile uporabljene za meritve v članku delujejo na operacijskem sistemu Android 8.0.0. Zanj je značilno, da vključuje funkcionalnost lokacijskih storitev (Google Location Services). Ta uporablja kombinacijo različnih virov za čim bolj natančno oceno lokacije naprave. Med temi viri sta seveda tudi povezljivost z omrežji WiFi in storitvami GNSS. Z vključitvijo omrežij WiFi je zagotovljena ocena lokacije tudi v primerih, ko povezljivost s storitvami GNSS ni na voljo (na primer v zaprtih prostorih). Vendar omrežja WiFi

sama po sebi nimajo informacije o svoji natančni geografski lokaciji. Za to se uporablja zunanja podatkovna baza, ki slika naslove MAC dostopnih točk v geografske koordinate. Ta podatkovna baza (ki se nahaja v oblaku podjetja Google) je zgrajena na podlagi ročno pridobljenih parov lokacij in seznamom zaznanih omrežij WiFi in periodičnih poročil posameznih naprav. Pari lokacij in omrežij WiFi so pridobljeni z dejanskim fizičnim potovanjem vozil po svetu in beleženjem zaznanih omrežij WiFi pod okriljem storitve Google Street View (čeprav se ti podatki ne zbirajo več). Drugi del podatkov je sestavljen iz poročil posameznih naprav z ustreznimi nastavitvami. Te namreč periodično sporočajo svojo lokacijo (storitev GNSS), oddaljenost od bližnjih dostopnih točk WiFi, stanje mobilnega omrežja in meritve ostalih senzorjev. Za oceno lokacije na podlagi vidnih omrežij WiFi je tako potrebno le opraviti nekaj vpogledov v to podatkovno bazo in ustrezno združiti dobljene podatke v konkretno geografsko lokacijo [5, 1].

	N	Min [m]	Max [m]	RMSE
WiFi, GNSS in 4G	141	2	11533	411.50
Samo 4G	174	7	27259	716.03

Tabela 1: Primerjava ocen napak dveh naprav z drugačnimi nastavitvami povezljivosti. RMSE predstavlja koren povprečja kvadratov napak (angl. root mean square error). Vir: [10].

5.2 Vpliv urbanosti okolja

Tabela 2 predstavlja primerjavo ocen napak med podeželskim in urbanim okoljem. Ti rezultati očitno kažejo na pozitiven vpliv urbanosti okolja na natančnost ocene lokacije. To domnevo potrjuje tudi statistični test (tabela 5). Naprava, ki je zbirala te podatke je imela vklopljeno zaznavanje omrežij WiFi in storitve GNSS [10]. Za dobljene rezultate obstaja več možnih razlogov. Za urbana območja je značilno večje število anten mobilnih omrežij in dostopnih točk omrežij WiFi. Hkrati pa je tudi obstaja možnost večjega števila objektov, ki negativno vplivajo na moč signala (zgradb).

Moč signala v podeželskem okolju je tudi nekoliko odvisna od okolja. Avtorji se sklicujejo na starejši članek, ki je preučeval vpliv okolja na moč signala na podeželju. Rezultati tega članka, ki so predstavljeni v tabeli 3, očitno kažejo na negativen vpliv poraščenosti okolja na natančnost ocene lokacije. Na odprtem so bili doseženi najboljši rezultati, malo slabši pod drevesi brez listja, najslabši pa med drevesi z listnatimi krošnjami [12].

Podobni rezultati so doseženi v starejši raziskavi, kjer so avtorji raziskovali vpliv naravnega okolja na natančnost sprejemnika GNSS. En del je zajemal primerjavo med območji brez listja, območja z iglavci in odprtimi območji. Za povprečje 500 meritev so zabeležili naslednje napake: 3,1 m za območja brez listja, 4,4 m za območja z iglavci in 2,2 m za odprta območja, kar sovпада z rezultati prejšnjega članka, če gledamo le vrstni red okoliščin glede na napake. Drugi del zajema vpliv oblike reliefa področja. Najboljši rezultati so bili doseženi na grebenih, malo slabši na pobočjih hriba, najslabši pa v dolini [3].

	N	Min [m]	Max [m]	RMSE
Podeželsko okolje	87	3	11533	519.36
Urbano okolje	54	2	300	87.00

Tabela 2: Primerjava ocen napak med podeželskim in urbanim okoljem. RMSE predstavlja koren povprečja kvadratov napak (angl. root mean square error). Vir: [10].

	Območje z listnatimi drevesi	Območje z drevesi brez listja	Odprto območje
Povprečna vrednost RMSE	8.90	7.36	3.00

Tabela 3: Primerjava ocen napak v natančnosti med različnimi stopnjami poraščenosti okolja. Prikazane vrednosti so povprečene za 7 različnih naprav. RMSE predstavlja koren povprečja kvadratov napak (angl. root mean square error). Vir: [12].

5.3 Vpliv generacije mobilnega omrežja

Tabela 4 prikazuje primerjavo ocen napak med generacijami mobilnega omrežja. Rezultati testnih statistik kažejo podobno zgodbo: najbližji naj bi bili porazdelitvi za omrežji 2G in 4G, najmanj podobni pa za omrežji 2G in 3G [10]. Rezultati se morda ne ujemajo z intuicijo, saj najboljši rezultati niso bili doseženi z najnovejšo tehnologijo. To je lahko posledica same pokritosti danega omrežja. Raziskava je namreč omejena na enega ponudnika mobilnega omrežja, vse lokacije meritev pa se nahajajo znotraj določene regije, torej ne smemo podajati širokih trditev o vplivu same generacije mobilnega omrežja na natančnost določanja lokacije.

Generacija	N	Min [m]	Max [m]	RMSE
2G	52	40	21935	827.43
3G	58	7	27259	550.13
4G	64	43	27024	750.58

Tabela 4: Primerjava ocen napak med generacijami mobilnega omrežja. RMSE predstavlja koren povprečja kvadratov napak (angl. root mean square error). Vir: [10].

6. ZAKLJUČEK

V članku smo se osredotočili na tri spremenljivke, ki vplivajo na natančnost meritve lokacije pametnega telefona. Rezultati so pokazali, da imajo vse tri svoj vpliv do neke mere. Na tem mestu se pojavi ideja o splošnem modelu za ocenjevanje natančnosti ocene lokacije. Rezultati meritev so poleg teh treh spremenljivk bili krepko odvisni tudi od same lokacije meritev. Ta se je izkazala za praktično najbolj utežen faktor za končni rezultat. Poleg teh je seveda še mnogo ostalih, kot na primer izbira naprave (ali samega omrežnega modula na čipu), orientacija naprave, pozicija satelitov, čas dneva, količina brezžičnega omrežnega prometa, standarda mobilnega omrežja (ne le generacije) in tako naprej. Implementacija splošnega modela, ki bi zajel zadostno število vseh možnih spremenljivk, se na tej točki zdi praktično neizvedljiva.

Glede na predstavljene rezultate, lahko zdaj postavimo nekaj trditev glede uporabe lokacijskih podatkov v sodnih pri-

Prva spremenljivka	Druga spremenljivka	Testna statistika	p-vrednost
WiFi, GNSS in 4G	Samo 4G	Mann-Whitney	5.54e-13
Podeželsko okolje	Urbano okolje	Mann-Whitney	0.00222
2G/3G/4G	2G/3G/4G	Kruskal-Wallis	0.02960
2G	3G	Conover-Inman	0.00420
2G	4G	Conover-Inman	0.13319
3G	4G	Conover-Inman	0.04937

Tabela 5: Rezultati testnih statistik za razne pare spremenljivk (radialne napake). Vir: [10].

merih. Konkretnih izjav, ki postavijo izbrano napravo na točno lokacijo ob danem trenutku, so predrzne. Boljši pri-stop upošteva razne negotovosti zaradi mnogih spremenljivk, še posebej če je majhno število zajetih podatkov (malo slik). Na sodišču je potrebno to predstaviti (na primer) v obliki verjetnosti prisotnosti naprave na različnih lokacijah glede na podatke, ki so bili zbrani na napravi. Podatki za take izjave so lahko zbrani tudi naknadno. S čim bolj podobno napravo in čim bolj podobnimi nastavitvami je potrebno opraviti meritve na različnih lokacijah. Nato se lahko te podatke primerja s predhodno pridobljenimi. Cilj je priti do trditve v smislu “Glede na pridobljene podatke je 10 krat bolj verjetno, da se je naprava nahajala na točki A kot na točki B” [11].

7. VIRI

- [1] How google uses location information. Dostopno na <https://policies.google.com/technologies/location-data?hl=en-US>.
- [2] E. Casey, D.-O. Jaquet-Chiffelle, H. Spichiger, E. Ryser, and T. Souvignet. Structuring the evaluation of location-related mobile device evidence. *Forensic Science International: Digital Investigation*, 32:300928, 2020.
- [3] C. Deckert and P. V. Bolstad. Forest canopy, terrain, and distance effects on global positioning system point accuracy. *Photogrammetric Engineering and Remote Sensing*, 62(3):317–321, 1996.
- [4] R. Garnett and R. Stewart. Comparison of gps units and mobile apple gps capabilities in an urban landscape. *Cartography and Geographic Information Science*, 42(1):1–8, 2015.
- [5] C. Hoffman. How devices use wi-fi to determine your physical location, 2021. Dostopno na <https://www.howtogeek.com/708500/how-devices-use-wi-fi-to-determine-your-physical-location/>.
- [6] J.-C. Liu, C.-H. Lin, and C.-Y. Lin. Accuracy evaluation of location service modes on mobile devices. *Sensors & Materials*, 30, 2018.
- [7] A. Macarulla Rodriguez, C. Tiberius, R. Van Bree, and Z. Geradts. Google timeline accuracy assessment and error prediction. *Forensic sciences research*, 3(3):240–255, 2018.
- [8] K. Merry and P. Bettinger. Smartphone gps accuracy study in an urban environment. *PloS one*, 14(7):e0219890, 2019.
- [9] T. Purfürst. Evaluation of static autonomous gnss positioning accuracy using single-, dual-, and tri-frequency smartphones in forest canopy environments. *Sensors*, 22(3):1289, 2022.
- [10] E. Ryser, H. Spichiger, and D.-O. Jaquet-Chiffelle. Geotagging accuracy in smartphone photography. *Forensic Science International: Digital Investigation*, 50:301813, 2024.
- [11] H. Spichiger. A likelihood ratio approach for the evaluation of single point device locations. *Forensic Science International: Digital Investigation*, 44:301512, 2023.
- [12] J. Tomaščík, J. Chudá, D. Tunák, F. Chudý, and M. Kardoš. Advances in smartphone positioning in forests: Dual-frequency receivers and raw gnss data. *Forestry: An International Journal of Forest Research*, 94(2):292–310, 2021.
- [13] J. Tomaščík Jr, J. Tomaščík Sr, Š. Saloň, and R. Píroh. Horizontal accuracy and applicability of smartphone gnss positioning in forests. *Forestry: An International Journal of Forest Research*, 90(2):187–198, 2017.

Mobile Device Forensics: A Targeted Overview

Aleksander Gomilšek

Faculty of Computer Science and Informatics
University of Ljubljana, Slovenia

Fabijan Marič Vild

Faculty of Computer Science and Informatics
University of Ljubljana, Slovenia

Roy Krumpak

Faculty of Computer Science and Informatics
University of Ljubljana, Slovenia

Žiga Trček

Faculty of Computer Science and Informatics
University of Ljubljana, Slovenia

ABSTRACT

We summarize a findings from several papers, with the main focus being article Nyon unchained: Forensic analysis of bosch's ebike board computer, where a Bosch Nyon device is used to obtain user information [5]. Besides the hardware components, the methodology used and the analysis is described. Additionally we look into papers linked to the main one. One of them explores the possibilities of using cars and their infotainment systems as sources of digital evidence, where a general approach is also outlined. Furthermore, we give attention to forensics of Internet of Things (IoT) devices, mainly the Amazon Echo Dot version 2. Lastly, we look into acquiring data from backups of smartphones.

KEYWORDS

Digital forensics, e-bike, car, smartphone, Internet of Things

1 INTRODUCTION

This article serves as an overview of the field of mobile digital forensics, where the primary source of information was chosen to be Nyon unchained: Forensic analysis of bosch's ebike board computers [5]. After summarizing the chosen article, our focus shifts to the works which are cited by it or even cite the article itself. Our main goal is to describe the methods used in analysing mobile devices, point out different approaches as well as their similarities.

In section 2 we summarize the main article and its findings. In section 3 we focus on an article about the forensics of cars. Section 4 is dedicated to forensics of Internet of Things devices and section 5 focuses on forensics of smartphones and backups.

2 BIKE COMPUTER FORENSICS

In Nyon unchained: Forensic analysis of bosch's ebike board computers, the authors focus on extracting data from Nyon, Bosch's eBike board computer. Both of the available two versions of this device are analyzed in detail from the perspective of forensic acquisition of valuable information [5].

2.1 Hardware and user interface

The first version of Nyon bike computer features either 1 or 8 GB of storage and is equipped with a 4.3-inch display controlled via a joystick and three buttons. It can be connected to a PC with a micro USB cable for debugging and diagnostics as well as have a Bluetooth connection with a smartphone running the "Bosch eBike Connect" app. While the first generation device came out in 2014, the second one was made available in 2021. The new version has a 3.2-inch touch screen and can have USB, Bluetooth or WiFi connections.

The two can be connected to the "Bosch eBike Connect" app, where some information about the bike and statistics about the rides can be observed.

2.2 Methodology

When choosing how to collecting data from the devices several aspects need to be taken into consideration:

- How easy (time-wise and effort-wise) it is to get the data off the device and what is the benefit of such approach?
- How well can we ensure the data is not changed in the process of data gathering?
- Is the process repeatable?
- How much data can we gather in such a way?

Many times in mobile forensics these aspects have contradictory goals. For example if we want to get all the data off a mobile device, a more intrusive procedure involving teardown and direct access to the components, such as flash drives, is needed. But doing so might compromise the repeatability of the process due to unreversible changes to the device. Additionally, the integrity of the data is not guaranteed as damage to the components can happen, so experience and time is needed to ensure this does not occur. Three types of data acquisition are presented in the paper, which are based on Digital evidence on mobile devices [2] and are taken into account when analysing the devices. The first and easiest one is manual acquisition, where the device is used as intended, the data is gathered through the device's user interface and the process is usually videotaped to ensure documentation. The second one is logical acquisition, where the access of data is extended to the devices file system as it is allowed by the operating system. Finally the physical acquisition centers on obtaining a bit-wise copy of the data and usually includes accessing the data directly off the memory chip. These methods are used on both of the devices, first on the earlier Bosch Nyon 2014 device.

2.3 Bosch Nyon 2014 analysis

Manual acquisition applied to the device gives us monthly cycling statistics, software version, the name and email address of the user, as well as recent destinations. More data is retrieved via logical acquisition, by exploiting a vulnerability in the update process. The device can be connected to a PC via a USB cable, where Bosch's "DiagnosticTool" is used to communicate with the device. When analysing how an update is carried out, it is observed that the diagnostic software first gets the latest update, then extracts it and decrypts it on the PC and in the end the update is installed on the Nyon computer. The latest update is signed and encrypted by Bosch,

so trying to switch the code is not possible. Nevertheless, the second step can be exploited, due to the process taking place on the PC. A Telnet service can be established on the Nyon device by interrupting the update process, modifying the decrypted files and installing them on the device. By further analyzing how this changes the data on the device, it is concluded that user data was not changed. In this way the user's name, email address, gender, date of birth, address, social media accounts, past trips, battery charge, performance, bike metrics and specification, GPS coordinates and network settings can be gathered from the device. With such information physical acquisition was not carried out, since this would be unnecessarily intrusive.

2.4 Bosch Nyon 2021 analysis

Following the same pattern, the newer version of the Nyon device was analyzed. Manual acquisition resulted in the user's name and email address. Logical acquisition was not successful with this device. This is because the decryption takes place on the device itself. Therefore, physical acquisition was performed, first with a less invasive hardware debug interface, which was not successful. Lastly a chip-off was executed. Heat was used to remove the chip, then it was cleaned and mounted to special equipment to read its content. The decryption key was found on the device and was used to decrypt the data. This resulted in names and passwords of known networks, last known location, system logs, user information, and bike specification being accessible.

2.5 Results

The amount of useful data collected from both devices is sufficient to associate a user with his driving behavior and with the time and location of a ride. Riding statistics offer an insight into a user's physical capabilities, so a connection between a ride and a suspect can be deemed plausible if enough information is provided. For example, if a suspect is obese and the ride suggests a demanding physical effort comparable to an elite athlete, the suspect may be ruled out as the one who performed the ride. The same can be said for the opposite case. Therefore, such devices can be used as evidence if strong enough arguments are provided.

3 CAR FORENSICS

This section is dedicated to the article A generalized approach to automotive forensics [1], where a general approach to investigating cars is presented and carried out. Some common terms and available data are first introduced and then the steps are described. The process is divided in four phases: forensic readiness, data acquisition, data analysis and documentation. Each of these steps are further elaborated.

3.1 Possible actors in a car investigation

When examining a vehicle several, entities may be involved in the request for information and event reconstruction. These include the insurer, who might question the car's integrity when dealing with an accident report, legal entities, who might question the vehicle's safety, supplier, who tries to conform to the safety standards, and finally the customer, who wants to have a working car.

3.2 Car data sources and types

Data on a vehicle can be extracted by a live monitoring procedure, where the system is turned on, or by a "post-mortem" approach, where the system is turned off. Additionally, a car component can be analyzed online i.e. plugged in to the car system or offline, where the component is physically taken from the car and analyzed elsewhere.

Five classes of data that are commonly present on a car are firmware, communication data, user data, safety data, and security data. Firmware data is the software installed, such as the operating system, drivers and applications. This type of data has been found to increase over the years and can be used to detect modifications. Communication data is all data that was transmitted either in the vehicle itself or between the vehicle and external sources. Here, existing tools such as Wireshark and OBD dongles can be used to get to system information. User data is gathered primarily through the connection of a smartphone, as well as any USB connected devices. Such data is valuable for a forensic investigation to create profiles of the driver or passengers and connect them to suspects. This data is again shown to be increasing over time. Safety-related data are any data that is tied to the state of safety of the vehicle and may include travel speed, seatbelt status and airbag deployment. This data has also been observed to increase over the past years. Lastly, security-related data are data that can directly or indirectly tell us something about vehicle security.

3.3 Car investigative process

The process is designed to be general and applicable to a wide range of scenarios. Four phases of the process are defined and described in detail, each of which is intended to be well documented when carried out.

Forensic readiness is the first step of the process and its goal is to find out if any available tools and sources are already available. First, potential data sources are researched and identified. Then the communication interfaces with the data sources are determined. Finally, these interfaces and tools are examined to adhere to a court-reporting standard, by checking documentation of former analysis, following best-practices, and using accepted tool-sets.

In the data acquisition step, the data is collected by determining the model variant and vehicle series, as well as its components, which are then assessed. Based on the circumstances, the interfaces identified in the forensic readiness step are filtered to choose the most suitable for the task. Then the data acquisition set-up is created and tested on a vehicle of same type and series. Finally, the data collected from the target car is copied and stored securely.

An analysis is then performed on the collected data, where an initial overview of the data is carried out, which then serves to filter out the relevant data. At the end of this step, a timeline and evidence trails should be formed.

The data investigation process concludes with the creation of a report intended to present the findings of the investigation. Results from the tools used are captured and selected for the final report, which includes the questions posed prior or mid-investigation and all failed or successful attempts at gathering information. As the last step, a review of the report is made, which concludes with the authors signature of the document.

An example vehicle investigation is described and carried out by the authors of the original paper. It is found that such forensic analysis is feasible, although some improvements are acknowledged, such as using tamper-proof data storage and expanding existing tools to new technologies.

4 NONDESTRUCTIVE DATA ACQUISITION IN IOT FORENSICS

In this section we will take a look at article Nondestructive data acquisition methodology for iot devices: A case study on amazon echo dot version 2 [6], where Villareal et al. propose a nondestructive methodology for acquiring data from the Amazon Echo Dot version 2.

4.1 Motivation

The main motivation behind the study was to develop a non-invasive technique to obtain complete data off the eMMC chip (embedded Multimedia Card), since the "chip-off" technique poses some inherent risk, as explained in the [3] study. The study used Amazon Echo Dot 2 as a test subject, but the findings are applicable to other IoT and mobile computing devices that use eMMC memory chips.

4.2 Methodology

The study used a donor device on which a chip-off was performed, followed by a CT scan. This was done to get the information about the exact pin locations. When all the information was gathered, the Test Probe Jig was 3D-printed and equipped with pogo pins. Then this was connected to the Echo Dot 2 with an eMMC reader and a computer. This works because the ISP can interact with the eMMC chip without the need to use the CPU. However, it is important not to turn the device on, as the data can be changed, which would make it inadmissible in a court of law. Considering all of this, the proposed methodology is a forensically safe process, that can be reused on other devices as long as they don't have intentionally hidden communication pings, since the Test Probe Jig will not be able to make contact without causing physical damage to the board.

4.3 Echo Dot 2 results

The extracted contents reveal a wealth of forensic artifacts, including Wi-Fi credentials, Bluetooth pairings, system logs, and user identifiers i.e. data that can be invaluable during criminal investigations. Importantly, the research did not find any user audio recordings of the user's interaction with the Alexa smart assistant inside the partitions.

5 DATA ACQUISITION USING LOCAL BACKUPS

Another promising data acquisition method that was attempted when trying to obtain data from Nyon e-bikes is using local backups. Article Systematic evaluation of forensic data acquisition using smartphone local backup [4] explores the challenges of obtaining data from mobile devices using local backups.

5.1 Motivation

The article aims to address the challenge of forensic data acquisition from smartphone devices. The difficulty of such tasks increases with time due to increasing security measures. Investigative work on the viability of such approaches is therefore very important.

Commercial forensic tools use local backup mechanisms to access data, but there is a lack of scientific and systemic evaluation of the reliability of data extracted using these methods.

Since forensic data need to be reliable and unaltered to be used in legal cases, it is important for the method to be researched and validated, otherwise it could be considered unreliable.

Due to these reasons, the article systematically evaluates the suitability of local smartphone backup mechanisms for forensic data acquisition for both Android and iOS devices.

5.2 Methodology

The study used a structured and systematic way to analyze the completeness of data acquired through local smartphone backups.

There are two main types of local backup:

- File-based: *name* is the filepath and *value* is the full content of the file.
- Content-based: *name* is the pair of file and data object within that file, *value* is the value of that data object.

They used multiple iterations during the evaluation. Each consisted of three steps:

- Pre-acquisition Reference Data: all name-value pairs that could potentially be obtained by the local backup method when evaluating.
- Data Acquisition: all name-value pairs obtained during the backup process
- Post-acquisition Reference Data: same name-value pairs as in step 1, but after the backup process - this helps identify changes occurred during the backup.

5.2.1 Reference Data Acquisition. To obtain reliable data, the authors used smartphones with root access.

On iOS, this means jailbreaking the device. The authors used an iPhone 8. Jailbreaking allowed access to the *private/var* partition, which contains most of the user and application data.

On Android, the authors used a rooted Google Pixel 2 phone with an unlocked bootloader. By rooting the phone, they could access directories that contain critical data, such as */data/data*.

Table 1: Comparison of forensic acquisition approaches across different device types

Device	Acquisition Methods	Type/Amount of Artifacts	Legal Risks / Admissibility Issues
Bosch Nyon (2014)	Manual, Logical (update process exploited)	User profile, GPS data, ride statistics, email, social media, hardware specs	Medium: vulnerability use may raise admissibility concerns; no physical tampering
Bosch Nyon (2021)	Manual, Physical (chip-off)	Network credentials, last known location, logs, user info, bike specs	High: chip-off is invasive; requires expert handling to ensure integrity
Modern Car Infotainment	Logical, Network-level, Component-level (both on-line/offline)	User profiles, trip history, sensor data, firmware, connected devices	Medium to High: depends on system tampering and whether logs are volatile or modified
Amazon Echo Dot (2nd Gen)	In-System Programming (ISP) with probe jig	WiFi/Bluetooth credentials, system logs, user IDs (no voice data)	Low: method is non-destructive and preserves integrity, but limited by device design
Smartphones (Android/iOS) Backups	OS-native backup tools (ADB/iTunes)	App data, media, communications, settings (iOS may alter WAL/SQLite)	Low to Medium: high integrity for Android, but iOS alters some DB states

5.2.2 Data Evaluation. The authors investigated which key-value pairs are unchanged in the backup process. They also noted the key-value pairs that did not exist pre-backup but existed in the backup. Most importantly, some keys remained the same but the values changed, which means data change in the backup process.

After performing comparison and mismatch classification, the authors used similarity ratio as a metric - this gives information on how much a value for a given key changed during the backup process.

5.2.3 Simulation. During the process of data extraction, normal phone usage was simulated - calls, SMS, emails, taking photos, and other everyday tasks were executed.

They performed methods built into each operating system to execute the data backup process.

5.3 Results

20 complete backups were performed for each of the operating systems. Additionally, app downgrades were performed to obtain app versions that are more suitable for local backups.

The data was found to maintain their integrity both for file-based data and content-based data on Android. No significant file alterations were found, except for some sparse noise attributed to background processes running at the time of the backup.

In the case of iOS devices, it was found that local backup is a good source of forensically relevant data. However, it was also found that the backup process alters SQLite database files by committing pending changes from WAL files. That means that information about uncommitted deletions may be lost in some cases.

Overall, the article demonstrates that local backup data is a very viable source of truth.

6 COMPARATIVE SUMMARY OF FORENSIC APPROACHES AND FINDINGS

This section is dedicated to the comparison between different approaches needed for different mobile devices. Each approach was evaluated based on methods used, type and amount of data, legal risks and tools used.

In table 1 the main metrics of different approaches are presented. The data gathered from a mobile device generally depend on what type of device it is. For example, a vehicle is more likely to store location information than a speaker. In addition, product development and iteration include security improvements, which go against forensic goals of gathering data, as was seen with the bike computer. Many mobile devices are still rich sources of configuration data, media data and log data. The tools used to inspect a mobile device are again conditioned on its type. This can mean that proprietary or special software or hardware need to be used to access the storage and chip of the device or even to connect to the device. For example the bike computer is analyzed with an application, car forensics often include OBD dongles, an eMMC reader for various small devices and root tools for smartphones. The methods used vary in their execution on actual devices, but are conceptually very similar to classical forensic methods. These can be divided into intrusive and non-intrusive.

7 CONCLUSION

This overview highlights the diversity and complexity of mobile digital forensics on devices such as bike computers, cars, smartphones, and IoT systems as presented in table 1. Each platform presents unique challenges and opportunities for data acquisition.

The analysis of Bosch's Nyon bike computers [5] showed how different acquisition methods—manual, logical, and physical—can

yield significant user and device data, though security advancements in newer models demand more invasive techniques. Car forensics benefits from a structured, repeatable process that accommodates multiple data types, as was shown in [1]. For IoT devices like the Amazon Echo Dot 2, nondestructive approaches such as ISP prove effective and safe, like in [6]. Meanwhile, smartphone local backups remain a valuable source of data, particularly when their limitations and effects on data integrity are well understood, which was presented in [4].

Across all domains, successful forensic investigations rely on choosing the right acquisition method, preserving data integrity, and staying adaptive to evolving technologies.

REFERENCES

- [1] K. K. G. Buquerin, C. Corbett, and H.-J. Hof. A generalized approach to automotive forensics. *Forensic Science International: Digital Investigation*, 36:301111, 2021.
- [2] E. Casey and B. Turnbull. Digital evidence on mobile devices. *Digital Evidence and Computer Crime*, 3:1–44, 2011.
- [3] C. Ence, J. Runs Through, and G. D. Cantrell. Chip-off success rate analysis comparing temperature and chip type. *Journal of Digital Forensics, Security and Law*, 13(4):7, 2019.
- [4] J. Geus, J. Ottmann, and F. Freiling. Systematic evaluation of forensic data acquisition using smartphone local backup. In *DFRWS USA*, 2023.
- [5] M. Stachak, J. Geus, G. Pugliese, and F. Freiling. Nyon unchained: Forensic analysis of bosch’s ebike board computers. *arXiv preprint arXiv:2404.12864*, 2024.
- [6] A. M. Villarreal, R. K. Verma, O. Upton, and N. L. Beebe. Nondestructive data acquisition methodology for iot devices: A case study on amazon echo dot version 2. *IEEE Internet of Things Journal*, 10(5):4375–4387, 2023.